Investigating the Application of Web-Based Simulation Principles within the Architecture for a Next-Generation Computer Generated Forces Model

Ernest H. Page and Jeffrey M. Opper

The MITRE Corporation, 1820 Dolley Madison Blvd., McLean, VA 22102

Abstract

With a heavy emphasis on distribution and reuse, web-based simulation portends a dramatic shift in the application of simulation as a problem-solving technique and decision-support tool. Next-generation simulation systems of all kinds should be evaluated and constructed with an appreciation of the potential paradigm shift that web-based simulation represents. Fiscal constraints indicate that next-generation computer generated forces (CGF) models will be used to support a wide range of missions, unifying—and replacing—a variety of CGF systems currently in existence. We describe the evolution of web-based simulation and derive a collection of modeling principles that characterize our vision of the web-based simulation future. We examine these principles in terms of their implications for next-generation CGF systems.

Keywords: Computer-generated forces, High Level Architecture, Semi-automated forces, Web-based simulation

1 Introduction

The emergence of the world-wide web (WWW) has produced an environment within which many disciplines are being re-evaluated in terms of their inherent approaches, techniques and philosophies. The disciplines concerned with computer simulation are no exception to this phenomenon—the concept of "web-based simulation" has been introduced and is currently the subject of significant interest to both simulation researchers and practitioners. While much of the discipline of web-based simulation seems to be little more than a technology push, some researchers in the field have ascribed to it the potential to effect significant, fundamental change in simulation modeling methodology. A web populated with digital objects—models of physical counterparts—is envisioned where modeling objectives are provided to search engines that, in

turn, identify the appropriate digital object(s) from which to construct an experimental model [11]. Common interfaces permit the objects to interoperate at runtime [17]. The physical locations of the objects involved in the computation are not relevant to the modeler. In this envisioned future, simulation becomes ubiquitous. Model conceptualization, construction, execution and analysis is distributed, collaborative, and interactive. Levels of automated support for the modeling process significantly increase, and the pace of modeling is rapid [18,19].

Computer generated forces (CGF) is a term used to describe simulations that provide representations of military forces. Also referred to as *semi-automated forces* (SAF), computer generated forces typically include models of behavior and decision making that enable the modeled forces to exhibit a degree of autonomy. Generally, though, CGF require a level of human-directed control during runtime. CGF have traditionally been used to populate "synthetic" battlefields in support of training with human-in-the-loop simulators. For example, a collection of tank simulators may be networked together using a common environment. CGF may be used to represent, or augment, the opposing forces in the training scenario.

Significant investments in CGF have been made over the past fifteen years. Within the U.S. Army, for example, numerous CGF systems are currently being supported, including the Modular Semi-automated Forces Model (ModSAF) and the Close Combat Tactical Trainer SAF (CCTT-SAF). Recently, plans for the next-generation U.S. Army CGF system have been formulated. This CGF is proposed to unify the capabilities of the existing Army CGFs and replace many of these systems. It will also be employed to support the full range of Army domains, namely: Advanced Concepts and Requirements (ACR), Research, Development and Acquisition (RDA), and Training, Exercise and Military Operations (TEMO) [26].

It seems reasonable to consider the design of next-generation simulation systems terms of the application of advanced and web-based technologies. In this paper, we briefly examine some of the possibilities for next-generation CGF systems. The remainder of the paper is organized as follows. In Section 2, we review the area of web-based simulation and attempt to characterize the impact it will have on future simulation practice. Six principles of web-based simulation are identified and discussed. Section 3 briefly describes the origins and nature of CGF modeling. The implications of the emergence of web-based simulation on next-generation CGF architectures are presented in Section 4. Our presentation considers not only the possible application of the WWW and web-based technologies to fulfill CGF system requirements, but also includes a consideration of the broader implications of the underlying principles of web-based simulation identified in Section 2. Conclusions appear in Section 5.

2 Web-Based Simulation

Web-based simulation is an emerging theme in simulation research and practice. Driven largely by the phenomenal growth in the World Wide Web (WWW) and its attendant technologies, it is tempting to view web-based simulation as nothing more than a *technology push.* To a certain extent, it is just that. A significant portion of the literature surrounding web-based simulation involves a re-dressing of the same old emperor in new technological garb. However, a few researchers in the field have described the possibility for the web to *fundamentally* alter the practice of simulation modeling (see [18,19]). We briefly characterize the scope of web-based simulation in the following section. In Section 2.2, we consider the question of the fundamental nature—and impact—of web-based simulation. Section 2.3 introduces six principles of web-based simulation, and discusses their relative merits. Some potential drawbacks of realizing these principles are discussed in Section 2.4.

2.1 What is web-based simulation?

The term *web-based simulation* emerged in the mid 1990s, although the exact pedigree of its coinage is unknown to this paper's authors. A 3-paper session within the modeling methodology track of the 1996 Winter Simulation Conference served to formally introduce the topic of web-based simulation to the simulation community at large. Two papers in the session describe simulation packages based on the Java programming language [4,15]. In the third paper [9], Fishwick offers his view of an emerging sub-area of simulation. In so doing, Fishwick provides a framework for much of the subsequent development of concepts in web-based simulation, and [9] is therefore widely regarded as the seminal work in the area.

Fishwick describes several potential impacts of web technologies on simulation, giving particular attention to three areas: (1) education and training, (2) publications, and (3) simulation programs. He describes the proliferation of web content as a "kind of twenty-first century gold rush" and admonishes simulation researchers and practitioners to be proactive in defining the relationship of the web and simulation.

Page [17] elaborates on Fishwick's observations, classifying web-based simulation as falling into five primary areas:

– *Simulation as hypermedia.* Text, images, audio, video ... simulation—the nature of the WWW design enables the production, storage and retrieval of "documents" containing any or all of these (and other kinds of) elements. The availability of simulation as a desktop, browser-based commodity has

the potential to significantly alter current teaching and training methodologies, both for simulation as a technique, and for disciplines that apply simulation, like engineering, physics, and biology. Paradigms that focus on distance learning and interactive, simulation-based education and training are emerging.
- *Simulation research methodology.* The ability to rapidly disseminate models, results and publications on the web permits new approaches to the conduct of simulation research, and scientific research in general. The practical, economic and legal issues associated with the electronic publication of documents, for example, are numerous (e.g. see [22]). The electronic publication of simulation models raises additional considerations. How will intellectual property rights be protected? Are model authors subject to liability if their models fail?
- *Web-based access to simulation programs.* Most commonly associated with the term web-based simulation, this area includes both the remote execution of existing (so-called "legacy") simulations from a web browser through HTML forms and CGI scripts, and the development of mobile-code simulations (e.g. applets) that run on the client side.
- *Distributed modeling and simulation.* This area includes activities that deal with the use of the WWW and web-oriented technologies (e.g. CORBA, Java RMI) as infrastructure to support distributed simulation execution [12,13,20,23,25]. Internet gaming issues are included here [5], as is research in tools, environments and frameworks that support the distributed (collaborative) design and development of simulation models [6,11].
- *Simulation of the WWW.* Modeling and analysis of the WWW for performance characterization and optimization.

As evidenced by the proceedings of the first two International Conferences on Web-Based Modeling and Simulation [3,10], web-based simulation is a diffuse topic. But it is a new area of investigation and perhaps it is a characteristic of any new area that a few years must pass before a core set of researchers and practitioners emerge and with them, a core focus.

2.2 Why is web-based simulation important?

A cornerstone of computing philosophy for much of the past 30 years is Edsger Dijkstra's principle of the *separation of concerns* [7]. Essentially, this principle argues that *correctness* is the fundamental objective of most software systems, and that correctness is fostered by first considering the design aspects of software without being encumbered by the peculiarities of the computing infrastructure that will serve to execute the software system. This separation of *specification* from *implementation* is thematic of much of software engineering, instrumental to the evolution of high level programming languages, and

widely supported in the simulation community as indicated by the proliferation of simulation-support environments.

Within the simulation modeling community—and arising largely from the automation-based paradigm [2]—a dominant view is that model specification is the domain of the modeler and model implementation is the domain of automation. That is, not only should implementation details be *delayed* until late in the software design process, as Dijkstra's principle asserts, but further that the modeler should be *insulated* (to as great a degree possible) from implementation details. It is the responsibility of the simulation-support environment to accept the modeler's descriptions and effect a suitable model implementation [14].

With this perspective in mind, the execution of models in a web-based setting seems essentially an implementation detail—a detail that should therefore be insulated from the modeler. On the other hand, there is the notion of *revolutionary* change: the emergence of new techniques and technologies that admit radical, fundamental change in approaches to a given problem. The assembly line is an example of such a revolution. Its invention radically altered the methodology for automobile production. Prior to the assembly line, technicians moved around factory floors to service stationary components. Subsequent to its invention, the components became mobile and the technicians stationary.

The question of whether web-based simulation represents an evolutionary or revolutionary change is addressed by Page et al. in [18,19]. The debate captured in these papers does not produce a consensus view, however compelling arguments are offered by Fishwick and Paul regarding the potential of the web as a revolutionary catalyst. Fishwick describes the web as an enabler of the digital object marketplace, enhancing the ubiquity of simulation. Paul describes the web as an enabler of rapid exploration, fundamentally altering traditional approaches to systems analysis and development.

Indeed, it is arguable that web-based simulation is *only* important to the degree that it *is* an impetus for revolutionary change. The possibilities for this type of change are characterized in the principles of web-based simulation introduced below.

2.3 Principles of web-based simulation

We believe that the concept of web-based simulation describes a future simulation practice that differs from current and "traditional" approaches in many ways. We suggest six *principles* of web-based simulation to illustrate these differences:

– Digital object proliferation.
– Software standards proliferation.
– Model construction by composition.
– Increased use of "trial and error" approaches.
– Proliferation of simulation use by non-experts.
– Multi-tier architectures and multi-language systems.

Any such list of principles is unlikely to be exhaustive, and not all of the principles noted fall directly, nor solely, from the concept of web-based simulation. However, taken together these principles capture an interesting vision of the future for simulation practice. We briefly address each of these principles in turn.

Digital object proliferation. In the web-based simulation future, the web will become less oriented toward a document metaphor and more oriented toward a digital object metaphor (where a document is simply one type of digital object) [11,19]. It will become increasingly commonplace for manufacturers of physical objects to employ simulation models of these physical objects during the life cycle of the object. These models will be viewed as the digital object equivalent of the physical object and will be made available for use through publication on the WWW. Fishwick [11] notes that publication of digital objects raises a number of significant questions. Who maintains the digital object? What are the object developers rights? How are they protected? What are the object user's rights? How are they protected? Fishwick suggests that answers to many of the questions in the digital object domain are simply analogues to those questions in the physical domain. If publication of digital objects becomes profitable, a marketplace of digital objects will emerge and the forces associated with free market economies will also drive the digital object marketplace. The linchpin of this vision, however, is the profitability of digital objects. The degree of such profitability is currently unclear.

Software standards proliferation. Software standards are not an outgrowth, by any means, of web-based simulation. However, their evolution is critical to enable interoperation and composition (see below) that form the hallmark of web-based simulation. Standard notations for digital object description must emerge that facilitate exploitation by search engines. Standard interfaces are required for "plug-replaceable" functionality. Existing standards such the Unified Modeling Language (UML), the Extensible Markup Language (XML), the Common Object Request Broker Architecture (CORBA), Object Linking and Embedding/Component Object Model (OLE/COM) and the U.S. Department of Defense's High Level Architecture (HLA) are each potentially valuable standards in this regard.

Model construction by composition. Given a web populated with thousands—perhaps even tens of thousands—of digital objects, the modeling process will naturally tend to begin with a *search* to find one or more digital objects suitable to the modeling objective. The rise of software standards—particularly interface standards like CORBA and the HLA and their concomitant implementations in middleware (see below)—will yield an environment where solutions will be sought through the composition of existing digital objects. In the world of digital objects, new code will be written only as a last resort.

Increased use of "trial and error" approaches. In [19] Paul argues that the web engenders a rapid, interactive approach to problem solving. In the web-based simulation age, with massive repositories of digital objects to choose from, model development will tend to be much more rapid. Does this combination of objects work? No. How about this combination? Paul asserts that the analysis process is improved since, "the search space has been dramatically reduced not by accuracy (the old way), but by massive and rapid search conducted by an empowered analyst (the new way)."

Proliferation of simulation use by non-experts. As simulation becomes a desktop commodity, it will therefore be available to masses. Such ubiquity is a mixed blessing. Having access to such a powerful problem-solving technique is potentially quite valuable. On the other hand, to the untrained user—a user with a what-you-see-is-what-you-get perspective—the potential to misapply the technique is great. As responsible engineers of the future, those enabling the web-based simulation revolution, should shepherd the safety of the technique to a reasonable degree. Education and methodology need to advance in close relationship with technology. Intelligent decision support agents would seem to be an essential safety mechanism.

Multi-tier architectures and multi-language systems. Despite the desires of Java Evangelists, it is unlikely that a singular programming language will emerge to replace all others. Certainly, Java may come to dominate the programming language landscape, but the diversity of programming tasks dictate a diversity of programming styles—imperative, functional, sequential, parallel—and languages that naturally support those styles. Therefore the digital object marketplace of the future will exhibit a heterogeneity of source language. The composition process will naturally produce multi-language systems. Multi-tier architectures are already becoming a dominant approach. It is quite likely this trend will continue. Stratification of functionality permits both specialization and optimization. In the context of simulation, for example, it may be desirable in one case to use middleware optimized to support

repeatable model execution (for analysis) and in other cases to use middleware optimized to support realtime execution (for training).

2.4 Is there a dark side?

One potential pitfall of the web-based simulation future described here has already been alluded to: making simulation available as a desktop commodity exposes the technique to a community of users not necessarily trained to use it properly. When gazing into our crystal ball we see a few other possibilities as well. Could digital objects proliferate beyond our ability to make effective use of them? Even today, finding information on the web is increasingly difficult due to the stupendous volume of data it contains. Search queries must be increasingly restricted, and even after careful restriction thousands of "hits" are often the result. But even if the search problem can be accommodated, the composition problem may be more insidious. Here, a user, or a fully- or semi-automated support system, must decide the best composition from among the available possibilities. We are confronted with combinatorial explosion. It seems intuitive that determining if the combination of component a, component b and component c is better than combination of component d and component e is analogous to determining the truth value of the Boolean expression $(a \wedge b \wedge c) \vee (d \wedge e)$. The general satisfiability problem is, of course, intractable and for expressions containing as few as 100 elements (in arbitrary clauses) a solution cannot be guaranteed to be producible in a reasonable time [8]. From this observation one might draw the rather dire conclusion that reusability in-the-large cannot succeed unless $P = NP$. It may, in fact, be more expedient in some (many?) cases to construct a system from scratch rather than attempt to compose one when the number of available candidates for composition is too large! In all likelihood, though, components can be organized and compartmentalized to avoid confronting this type of intractability. Still, we believe thoughtful consideration of both the positive and negative consequences of achieving the principles of web-based simulation is warranted.

3 Computer Generated Forces

The genesis of CGF and SAF simulations occurred through the efforts of the Defense Advanced Research Projects Agency (DARPA) SIMNET program of the mid to late 1980s. SIMNET consisted of a series of tank simulators whose viewports were coupled to 3-D image generators. These image generators rendered a somewhat realistic representation of the environment or *battlespace* from a digitized representation of the terrain elevations, soil types,

and features in the area of interest. The tank simulators were interconnected via Ethernet and used a common protocol (*the SIMNET protocol*) to share state data [27]. In each tank simulator, crew positions (gunner/loader, driver and commander) were represented using a concept referred to as "selective fidelity" [28, p. 17]. This principle follows from a minimalist philosophy—only those artifacts necessary to produce the desired behavior in the trainee should be captured within the simulator. Rather than real vision blocks or viewports, special image generator hardware was used to project an artificially generated (or synthetic) external environment to the crew member. As the tank simulator moved across the synthetic terrain, its position was periodically updated in the image generator to provide the illusion of movement. This data was also broadcast to the other simulators participating in the exercise to enable representations of the tank within their local viewports.

In early uses of the SIMNET simulators, this arrangement was sufficient for small team training events. However, to stimulate simulator crews with tactically significant opposing forces, it became readily apparent that augmenting the synthetic battlespace required a more scaleable solution than simply adding more manned simulators. To satisfy this need, special-purpose computer programs were developed that allowed a user to create a group of simulated battlefield objects (tanks, trucks, aircraft and other vehicles) and control their behavior. This control mechanism was through a scripted set of tasks that were executed using special templates providing environmental context (routes, points, positions, and so forth). The simplistic nature of these behavioral scripts resulted in the need for operators to monitor model execution to correct doctrinally aberrant behaviors that could arise when the model was operating in a highly dynamic and unpredictable environment. This requirement for operator monitoring and control gave rise to the moniker semi-automated forces.

Over the intervening years, the number and complexity of CGF systems has steadily increased. Modular Semi-Automated Forces (ModSAF) is perhaps the best known contemporary example of this type of system. During the 1990s, ModSAF has grown from its roots as a relatively simple provider of ground-based armor platforms to a complex simulation capable of injecting a large variety of ground, air, and sea-borne platforms, obstacles, and environmental features into the synthetic environment. This growth comes with a price—the size of the C language codebase has dramatically risen from 100,000 lines to over 1,000,000 in the latest release. A number of other CGF or CGF-like simulations have also been developed to satisfy particular user needs. The includes the Army's Close Combat Tactical Trainer (CCTT) SAF and the Synthetic Theater of War (STOW) JointSAF both of which originated from the basic ModSAF design, and distant cousins such as JANUS and CASTFOREM which have been developed to support analytical uses.

4 Web-Based Simulation Principles and CGF Architectures

Next-generation CGFs will exist in a web-enabled world. To ignore this reality and proceed to develop such systems in the traditional monolithic fashion would risk immediate obsolescence and rejection by a user community rapidly accepting the web's ubiquity. Instead, we might choose to initiate the synthesis of a new CGF architecture through the adoption of four basic architectural metaphors [21]. These metaphors embrace the essence of the principles of web-based simulation noted in Section 2.3.

4.1 The Extranet Metaphor

Future CGFs will be globally distributed to support myriad training and analysis uses. What better way exists to support collaboration and information sharing between sites than to cultivate the development of a CGF extranet? It can be easily envisioned that an installation process automatically configures a site web server with a default set of hypertext documents, applets, agents, and other infrastructure components that immediately support interaction via the browser on the user's desktop. In this context, the user can invoke CGF life cycle applications, monitor simulation executions, perform data analysis, and search remote site repositories using an environment that is familiar and intuitively navigable.

Figure 1 schematically illustrates a fragment of that extranet where several user sites and a software configuration management activity (SCA) are interconnected. Automated web-based software release and update distribution can be handled through the use of proposed standards such as the joint Microsoft/Marimba Open Software Description (OSD) specification and the Distribution and Replication Protocol (DRP) which have been submitted to the World Wide Web Consortium (W3C) for consideration. OSD supports the specification of the composition of a software package (contents, dependencies, prerequisites) using XML. DRP allows software entities to examine OSD specification files, access packages, and download software distributions to remote hosts.

Figure 1 also depicts "web-aware" applications that allow CGF users to canvas the CGF extranet for digital objects such as platforms (tanks) or platform components (hulls), execution scenario files, or simulation output data that may be required to support a local simulation requirement. A variety of technological solutions exist or are under development to support the integration of application software with web-based software agents that support distributed information query and retrieval. Active Software's Active Web and Talarian's

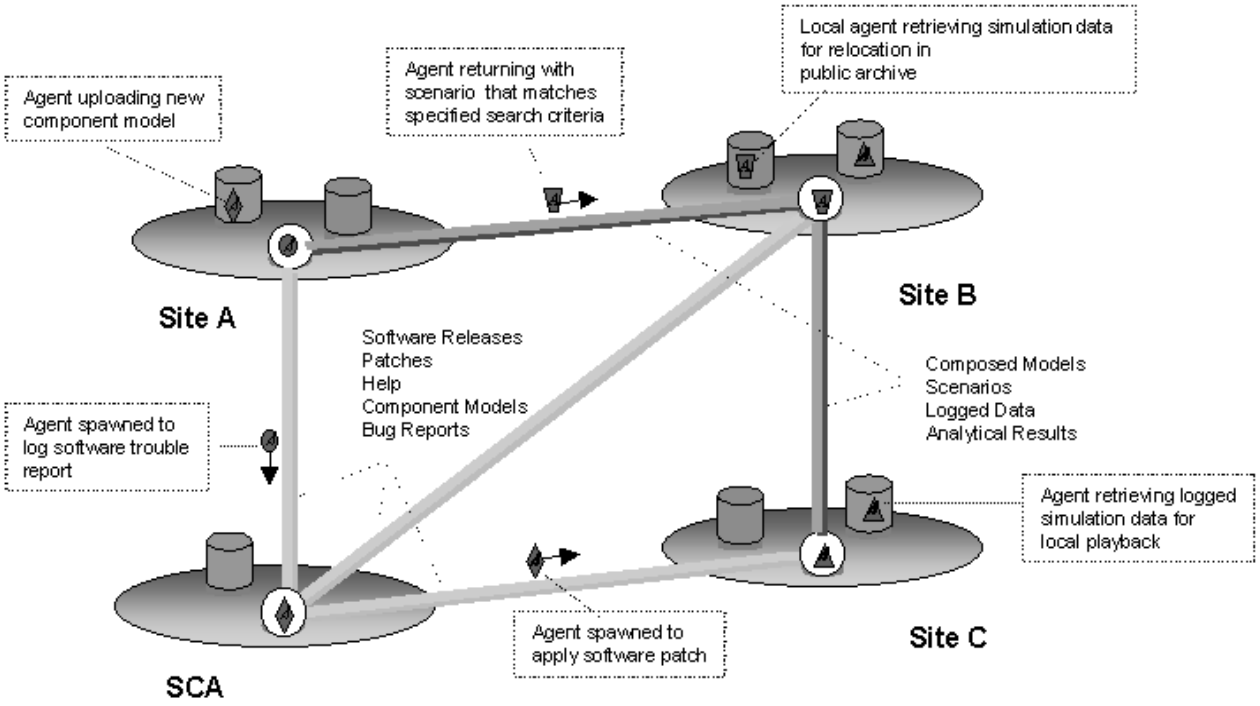

Fig. 1. A CGF Extranet.

Smart Sockets are two examples of current vendor approaches to this type of integration. Fundamentally, the web makes it much easier to borrow than to build.

4.2 The Melting Pot Metaphor

Evolving operational requirements indicate that the intended future use of CGFs generally conforms with the kind of "cradle-to-grave" model development life cycle expressed by Balci [1]. Software life cycles such as this call for a number of tools that perform specific tasks:

- Requirements decomposition and conceptual model development
- Model composition, analysis, and verification
- Code generation, compilation, and linkage
- Simulation configuration, initialization, execution, and control
- Data reduction, analysis, presentation, and archival
- Model maintenance

A number of architectural styles (Figure 2) exist that can be used to describe software systems such as *pipes and filters, communicating processes, virtual machines, and event systems* [24]. In practice, large software systems tend not

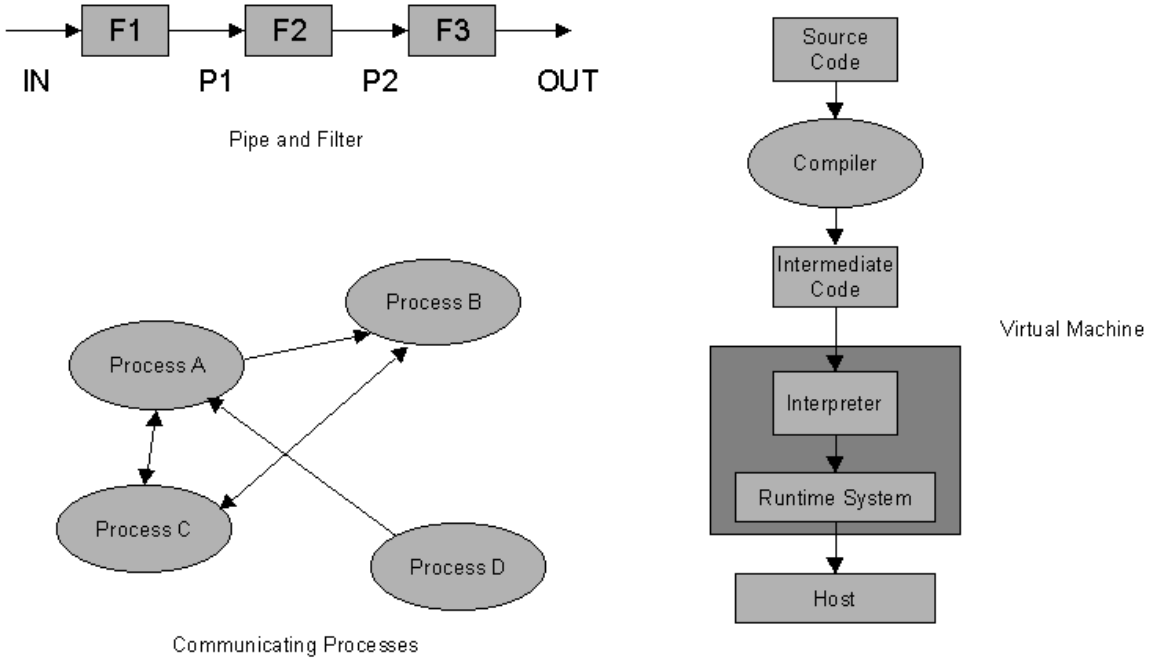

Fig. 2. Architectural Styles.

to be architecturally "pure," but instead adopt a heterogeneous mix of styles reflecting the optimization of form to function. Future CGF should provide no exception—database-centric processes such as model composition could best be characterized as client-server micro-architectures, while the simulation kernel itself could be viewed as an event system or a collection of communicating processes depending upon implementation specifics.

Selection of a particular micro-architecture also tends to bias the selection of languages and tools to those best supportive of the computational model implied by the style. Consider the requirement to build a model composition tool that allows a user to construct a complex object from a set of component parts. The selection of a client-server style where the tool accesses a central parts repository may lead to a decision to adopt a bean-based approach implemented in Java and using the JDBC interface specification to support connection to a traditional relational or object database. Web interfaces will naturally call for the use of HTML, XML, and Perl to support static and dynamic web page generation. Performance requirements of the simulation kernel may preclude the use of Java and instead call for the use of languages that efficiently compile to native code. Extending this exercise through the remainder of the system leads to a quick realization that future CGFs will be a polyglot where Java, C, C++, HTML, XML, Perl are leading contenders.

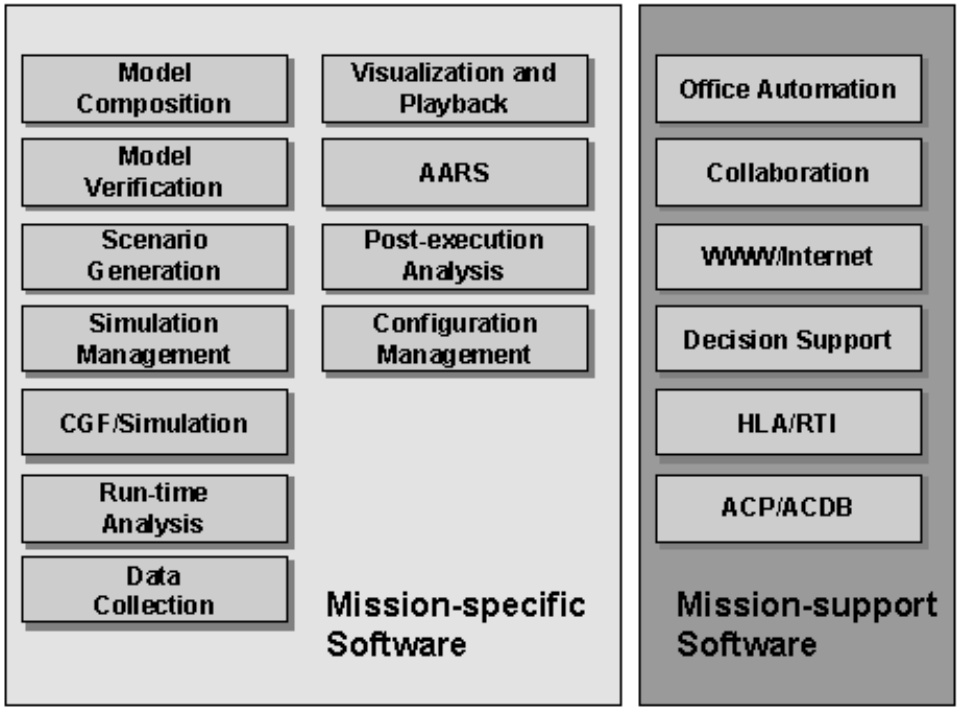

Fig. 3. A CGF Product Line.

4.3 The Product Line Metaphor

The idea that future CGFs will not be architectural monoliths can cause concern to management and oversight activities responsible for providing reasonable guarantees that such CGFs will be built on schedule and within budget allowances. Adopting a product line approach to the development of these systems is natural risk mitigation strategy. In [16], Northrop presents three key terms that define the core elements of the approach:

- *Product Line:* a group of products sharing a common, managed set of features that satisfy specific needs of a selected market.
- *Product Family:* a set of related systems that are built from a common set of core assets.
- *Domain:* a specialized body of knowledge; an area of expertise.

In our case, the CGF product line (Figure 3) will contain a product family consisting of model composers, model verifiers, scenario generators, simulations, execution managers, analysis tools, and other products that support the full life cycle. Interestingly, the domain of consideration here is somewhat novel as we are considering the CGF to be a web-based simulation. Our fundamental challenge, as we continue specification of such a CGF architecture, is to identify those core assets that enable our product line to use the web in a con-

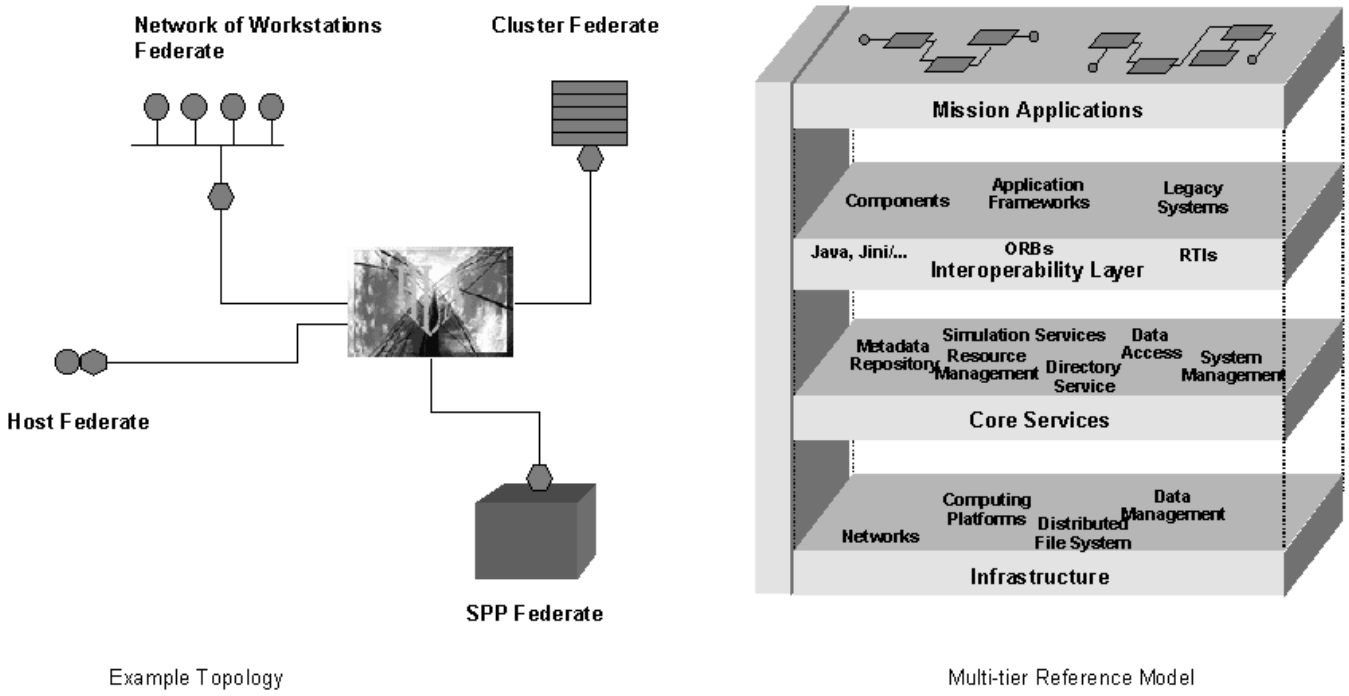

Fig. 4. CGF in an HLA Federation.

sistent and reusable manner. While, the proliferation of *de facto* and *de jure* interoperability standards (HTML, XML, RMI, CORBA, and JDBC/ODBC to name a handful) will help guide the specification process, the dynamicism of the marketplace reinforces the need for the encapsulation of software functionality and the standardization of interfaces to achieve a level of insulation and "plug-replaceability."

4.4 The CGF/HLA Turing Test Metaphor

In a spin on the classical Turing Test, this metaphor considers the requirement levied by DOD that CGFs be HLA-compliant simulations. Compliance requires that a CGF (*a federate*) interoperate with other simulations (*a federation*) using the services provided by the HLA. The HLA Run-Time Infrastructure (RTI) is a collection of software libraries that provide a number of services to simulations wishing to interact with other software systems in a coordinated manner. Coupling of the RTI with any simulation can occur in one of three ways: (1) via native code integration, (2) via middleware (brokered invocation), and (3) via gateways (protocol translation).

Several factors must be considered in the determination of how future CGFs will couple with the HLA RTI. First, future CGFs will likely be distributed

simulations fielded on systems including conventional personal computers, networks of workstations, cluster computers, and scalable parallel processors depending upon the nature and scale of the simulation requirement. Additionally, CGFs will be required to support multi-modal simulation time flow mechanisms capable of executing both in real-time and faster than real-time. Such requirements place numerous technical challenges on the system architecture. The level of technical uncertainty levied by the totality of the CGF requirements may lead us to discount native code coupling as the preferred approach and instead focus on a multi-tiered architectural approach. Stratified isolation of the simulation from the RTI services layer (Figure 4) permits us to achieve a measure of flexibility in the implementation. Compliance is assured if the CGF (in whatever form) stimulates, and is stimulated by, the federation in the appropriate manner. Again, this approach emphasizes the need for well-established interoperability standards.

5 Conclusions

The era of the web has certainly arrived. Fishwick's characterization of a "twenty-first century gold rush" is well underway. In this paper, we have tried to speculate on the magnitude—both positive and negative—of the forces for change that the web and web-based technologies bring to bear on the venerable technique of simulation. A World Wide Web populated with digital objects will give rise to a predominate approach to modeling based on composition. Model conceptualization, construction, execution and analysis will be distributed, collaborative and highly interactive. Use of the technique by "non-experts" will demand that levels of automated support for modeling and simulation activities significantly increase. We note also, by way of caution, that the proliferation of digital objects may give rise to an environment so cluttered and so voluminous that many modeling problems become unmanageable, or worse, intractable or undecidable.

In the second part of the paper we describe a candidate architecture for next-generation CGF systems, and characterize it with respect to the principles of web-based simulation. As this architecture evolves, we will continue to evaluate it with respect to these principles—with an underlying objective of ensuring that future CGFs will live harmoniously in the web-based simulation world.

References

[1] Balci, O. (1986). "Requirements for Model Development Environments," *Computing and Operations Research,* **13**(1), pp. 53-67.

[2] Balzer, R., Cheatham, T.E. and Green, C. (1983). "Software Technology in the 1990's: Using a New Paradigm," *IEEE Computer,* **16**(11), pp. 39-45, November.

[3] Bruzzone, A.G., Uhrmacher, A. and Page, E.H., Eds. (1999). *Proceedings of the 1999 International Conference on Web-Based Modeling and Simulation,* SCS Simulation Series **31**(3), 255 pages.

[4] Buss, A.H. and Stork, K.A. (1996). "Discrete Event Simulation on the World Wide Web Using Java," In: *Proceedings of the 1996 Winter Simulation Conference,* pp. 780-785, Coronado, CA, 8-11 December.

[5] Crews, T.M. and Louis, K.F. (1999). "Investigating Interoperability in a Commercial Gaming Engine," In: *Proceedings of the 1999 International Conference on Web-Based Modeling and Simulation,* pp. 149-155, San Francisco, CA, 17-20 January.

[6] Cubert, R.M. and Fishwick, P.A. (1997). "A Framework for Distributed Object-Oriented Multimodeling and Simulation," In: *Proceedings of the 1997 Winter Simulation Conference,* pp. 1315-1322, Atlanta, GA, 7-10 December.

[7] Dijkstra, E.W. (1976). *A Discipline of Programming,* Prentice-Hall, Englewood Cliffs, NJ.

[8] Garey, M.R. and Johnson, D.S. (1979). *Computers and Intractability: A Guide to the Theory of NP-Completeness,* W.H. Freeman and Company, New York, NY.

[9] Fishwick, P.A. (1996). "Web-Based Simulation: Some Personal Observations," In: *Proceedings of the 1996 Winter Simulation Conference,* pp. 772-779, Coronado, CA, 8-11 December.

[10] Fishwick, P.A., Hill, D.R.C. and Smith, R., Eds. (1998). *Proceedings of the 1998 International Conference on Web-Based Modeling and Simulation,* SCS Simulation Series **30**(1), 203 pages.

[11] Fishwick, P.A. (1998). "Issues with Web-Publishable Digital Objects," In: *Proceedings of SPIE: Enabling Technologies for Simulation Science II,* pp. 136-142, Orlando, FL, 14-16 April.

[12] Fox, G.C., Furmanski, W., Nair, S., Ozdemir, H.T., Ozdemir, Z.O. and Pulikal, T.A. (1999). "WebHLA – An Interactive Multiplayer Environment for High Performance Distributed Modeling and Simulation," In: *Proceedings of the 1999 International Conference on Web-Based Modeling and Simulation,* pp. 163-168, San Francisco, CA, 17-20 January.

[13] Klein, U. S. Straßburger, and J. Beikrich. (1998). "Distributed Simulation with JavaGPSS based on the High Level Architecture," In: *Proceedings of the 1998 SCS International Conference on Web-Based Modeling and Simulation,* pp. 85-90, San Diego, CA, 11-14 January.

[14] Nance, R.E. (1994). "The Conical Methodology and the Evolution of Simulation Model Development," *Annals of Operations Research,* **53**, Special Volume on Simulation and Modeling, O. Balci (Ed.), pp. 1-45.

[15] Nair, R.S, Miller, J.A. and Zhang, Z. (1996). "Java-Based Query Driven Simulation Environment," In: *Proceedings of the 1996 Winter Simulation Conference,* pp. 786-793, Coronado, CA, 8-11 December.

[16] Northrop, L. (1997). *Essentials of Product Line Practice.* URL: `http://www.sei.cmu.edu/plp/essentials_slides` [June 17].

[17] Page, E.H. (1998) "The Rise of Web-Based Simulation: Implications for the High Level Architecture," In: *Proceedings of the 1998 Winter Simulation Conference,* pp. 1663-1668, Washington, DC, 13-16 December 1998.

[18] Page, E.H., Buss, A., Fishwick, P.A., Healy, K.J., Nance, R.E. and Paul, R.J. (1998). "The Modeling Methodological Impacts of Web-Based Simulation," In: *Proceedings of the 1998 SCS International Conference on Web-Based Modeling and Simulation,* pp. 123-128, San Diego, CA, 11-14 January.

[19] Page, E.H., Buss, A., Fishwick, P.A., Healy, K.J., Nance, R.E. and Paul, R.J. (1999). "Web-Based Simulation: Revolution or Evolution?" submitted to: *ACM Transactions on Modeling and Computer Simulation,* February.

[20] Page, E.H., Moose, R.L. and Griffin, S.P. (1997). "Web-Based Simulation in Simjava using Remote Method Invocation," In: *Proceedings of the 1997 Winter Simulation Conference,* pp. 468-474, Atlanta, GA, 7-10 December.

[21] Rechtin, E. and Meier, M. (1997). *The Art of Systems Architecting,* CRC Press, Boca Raton, FL.

[22] Samuelson, P. (1996). "Regulation of Technologies to Protect Copyrighted Works." *Communications of the ACM,* **39**(7), pp. 17-22, July.

[23] Sarjoughian, H.S., and B.P. Zeigler. (1998). "DEVSJAVA: Basis for a DEVS-based Collaborative M&S Environment," In: *Proceedings of the 1998 SCS International Conference on Web-Based Modeling and Simulation,* pp. 29-36, San Diego, CA, 11-14 January.

[24] Shaw, M. and Garlan, D. (1996). *Software Architecture: An Emerging Discipline,* Prentice-Hall, Upper Saddle River, NJ.

[25] Shen, C.-C. (1998). "Discrete-Event Simulation on the Internet and the Web," In: *Proceedings of the 1998 SCS International Conference on Web-Based Modeling and Simulation,* pp. 57-62, San Diego, CA, 11-14 January.

[26] U.S. Army Simulation, Training and Instrumentation Command. (1998). OneSAF Operational Requirements Document. Version 1.0, Orlando, FL, June.

[27] U.S. Defense Modeling and Simulation Office. (1993). 1993 DMSO Survey of Semi-Automated Forces, DMSO, Alexandria, VA, July.

[28] Voss, L.D. (1993). *A Revolution in Simulation: Distributed Interaction in the '90s and Beyond,* Pasha Publications, Inc., Arlington, VA.