# Chapter 1

# Toward a Unified Foundation for Simulation-Based Acquisition

H.S. Sarjoughian and F.E. Cellier

*Simulation-Based Acquisition (SBA) has become an important framework for the development of engineering systems of high complexity. It offers a rapid prototyping capability for the design and/or evaluation of engineering systems, the components of which are by themselves complex systems that may be manufactured by different vendors. Using SBA, the designers of such Systems of Systems can verify that the interplay between the component systems functions correctly and reliably. The paper stipulates that SBA is enabled by the synergism of three technologies, namely Modeling & Simulation (M&S), Artificial Intelligence (AI), and Software Engineering (SE).*

## 1.1 Introduction

The average complexity of engineered systems has grown over the years at a phenomenal rate. Let us consider advances in the car industry. Modern cars are full of electronics that control everything, from the behavior of the brakes to the operation of the windshield wiper that turns automatically on and off depending on the current weather conditions. Even the servicing of the car is partly automated. When the car is brought to the garage, its on-board computers communicate via the Internet with the master computer of the car manufacturer to check whether the car operates as intended.

Engineered systems must be produced on an increasingly tight schedule. Let us consider advances in the building industry. A modern mobile home is delivered to the customer within weeks from the day it was ordered. Yet the house cannot truly be prefabricated. No two houses are ordered exactly the same way. There are lots of options, including the color of the carpets, the specs of the bathrooms, and the location of the electrical outlets. Depending on the state the house is to be delivered to and also the altitude where the house is to be used, building codes change, requiring a modification in the insulation, the

inclination of the roof, and the length of the stacks. Furthermore, in line with modern just-in-time delivery, manufacturers no longer carry any spare parts. They no longer have warehouses to store them. Thus, even the parts are ordered exactly when they are needed, and it is expected that they will be delivered within days if not hours.

Such demands could not be met by the manufacturing industry if it were not for Simulation-Based Acquisition (SBA). There simply is no margin for trial and error in this process. When a manufacturer orders a part, he must know in advance that the part will function as desired, and that it will arrive at the time when he needs it. To this end, all facets of the manufacturing process are being simulated ahead of time, using models of parts that are provided by the part manufacturer, that can be downloaded across the Internet from the website of the part manufacturer, and that can be readily plugged into the overall models simulating the manufacturing process.

The scenario described in the previous paragraph is taken from a futuristic world. In the current reality, part manufacturers rarely offer models of their products, and if they do, these models are rarely in a form that they could be integrated easily and rapidly into the simulation model of a manufacturing process. Exact simulation results must more often than not be replaced by rough estimates that are based on engineering experience; overly optimistic assessments often lead to unforeseen delays and cost overruns; and many a company has already folded due to such errors.

It is the purpose of this chapter to outline some of the requirements for a successful application of SBA. It is the conviction of the authors that only a seamless synergy between three technologies: Modeling & Simulation (M&S), Artificial Intelligence (AI), and Software Engineering (SE) can create an environment in which SBA can be successfully employed.

Advances in areas such as Artificial Intelligence (AI), Modeling & Simulation (M&S), and Software Engineering (SE), have been instrumental in creating the technological era we live in. Significant recent developments in the theories and practices of AI (e.g., [15]), M&S (e.g., [6,14,32]), and SE ([13,22,26]) have already supported the development of numerous engineered systems such as distributed training simulators and multi-agent systems. Unprecedented technological advances necessitated by economic and geo-political globalization attest to the fact that systems will continue to grow in their complexities and interdependencies on one another. Let us consider training simulators in such diverse fields as medicine and warfare, where simulators demand realistic models for the engineered systems themselves, their environment, as well as interactions with the human users of these systems [3,27]. Sophistication of such systems, in part, can be attributed to two fundamentally distinct yet interdependent demands. First, systems are expected to interoperate, in a heterogeneous setting, in order to achieve shared objectives. Second, systems are required to exhibit sophisticated degrees of autonomy (or more realistically modest degrees of semi-autonomy). The capability of systems to behave collectively and autonomously in distributed heterogeneous settings

promises to be one of the most difficult challenges posed to the scientific and engineering communities at large.

It is the purpose of this book to offer a kaleidoscopic view of the set of tools that will be needed to bring SBA to maturity. The vantages of the collection of articles included in this volume are manifold. Indeed this is to be expected given the breadth and depth of AI, M&S, SE, and in particular their interactions. Examinations of the articles reveal the essential roles that AI, M&S, and SE play not only individually, but also more importantly, synergistically. We note that perspectives portrayed by each article may be borne out of the reader's own interpretations or due to the authors' proclaimed perceptions and beliefs. These perspectives may, as matter of course and in their own unique ways, reveal some historical views, suggest the current state of knowledge, and/or foretell some elements of a near-term, and in some cases longer-term, futuristic information age landscape. These articles offer novel approaches –knowledge representation, reasoning, and/or simulation to name a few– suitable for addressing the needs of "Systems of Systems" based on the foundations and applications of AI, M&S, and SE.

We describe, in an abstract setting, the tapestry of the contributed articles from two specific, complementary viewpoints. The first presents a view of the landscape that has evolved from AI, M&S, and Software Engineering[1]. This treatment is centered on the discrete event worldview for reasons described in Section 1.2.1. The second suggests an approach toward systematic use of concepts, theories, and practices –rooted in a variety of disciplines and most notably AI, M&S, and SE– in building advanced systems as mentioned earlier. The approach has been called Concurrent Engineering (e.g., [8]), and more recently, Simulation-Based Acquisition [25].

## 1.2 AI, M&S, and SE: A Unified Perspective

From a conceptual point of view, the realization of distributed systems hinges on appropriate knowledge representation schemes, reasoning mechanisms, computational techniques, and evaluation strategies. A particularly elusive enigma for realization of engineered systems is their ability to evolve/adapt in dynamic environments. Challenges in modeling (e.g., model abstraction and validation), efficient computational techniques (e.g., parallel/distributed simulation), and realization of such systems (e.g., software design methodologies) are generally well known within AI, M&S, and SE. There are numerous overlaps in the research inquiries that are being pursued in each of these fields independently. In spite of these overlaps, the inquiries are not generally redundant, as they are based on the individual viewpoints and

[1]Our discussion is not founded on a rigorous treatment of a shared worldview on AI, M&S, and SE, as this is an open research inquiry.

idiosyncrasies characteristic of the three research areas. Yet, it is our conviction that these developments alone are insufficient to tackle the complex demands of SBA, and that only a consequent collaboration between researchers in these three areas, exploiting their synergism to the fullest extent, can bring SBA technologies to maturity. Stated differently, a unified approach is necessary to deal with the multitude of challenging issues such as model (software) composability, distributed execution, and elevating existing technologies to future higher grounds. Even under the best of circumstances, tackling the SBA problem will be a formidable undertaking!

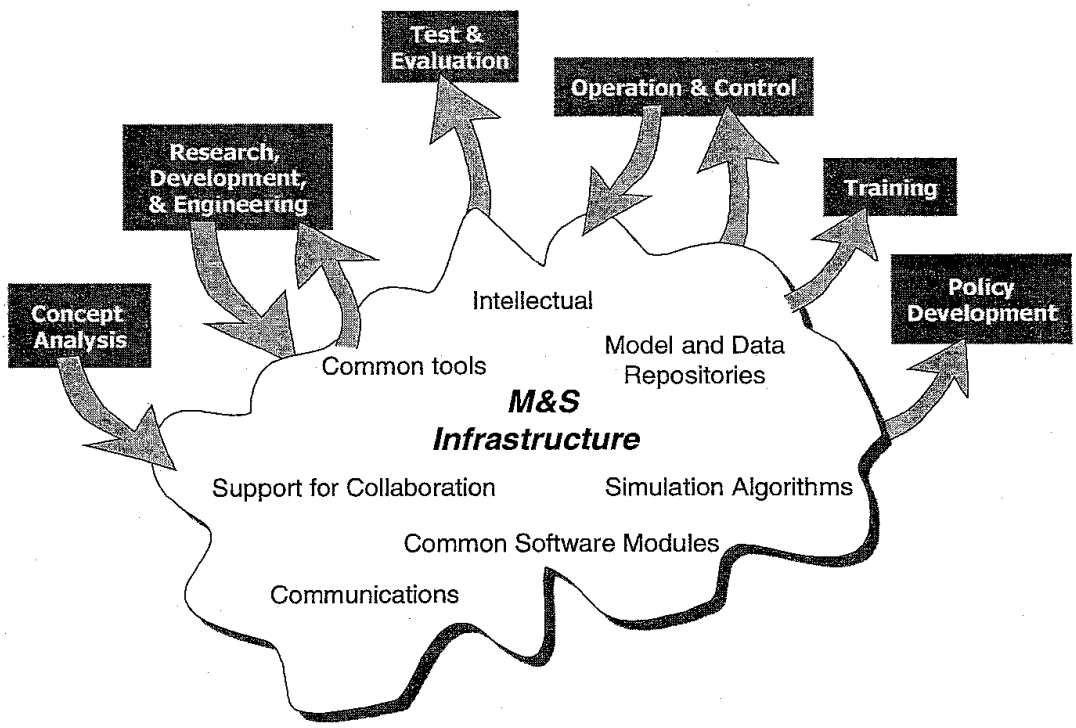


Figure 1.1: M&S infrastructure in support of SBA

Figure 1 illustrates various facets of modeling and simulation. It is easy to observe the need for capabilities offered by AI and SE in addition to those of M&S. Consider training systems where artificial intelligence techniques must be employed to represent a "virtual" patient being operated on in a synthetic operating room. During surgery, unforeseen events may happen, such as when the trainee accidentally perforates a blood vessel. The AI controlling the patient model needs to react to this situation, and drive the simulation such that it responds realistically to the incident by changing the patient's blood pressure and heart rate. Clearly, advanced Software Engineering practices are required to develop such extraordinary virtual training environments.

To bring together AI, M&S, and SE, it is imperative to adopt a general architectural framework and methodology. A candidate high-level architecture has been suggested in [31,24]. This architecture consists of seven layers (1) Network (the lowest layer), (2) middleware, (3) simulation, (4) modeling, (5) search, (6) decision, and (7) collaboration (the highest layer). Lower layers provide services for the upper layers, and higher layers can use and possibly subsume the roles of the lower layers. This architecture can be used for both simulation and system development. Adoption of a common architectural framework enables designers to study competing/alternative designs by employing a mixture of "simulated" and "performing" systems and therefore support migration from simulation design to operational design.

Based on the viability of such a layered architecture, the claim is put forth that AI, M&S, and SE are three distinct pillars upon which Systems of Systems can be built[2]. Indeed, our view stems from the fact that it is increasingly crucial for research in these areas and others (e.g., psychology and natural language) to unify in such a way that the whole is significantly more than the sum of its parts. In particular, a *modular*, *layered*, and *hierarchical* architectural framework is adopted as the scaffolding to unify the elements of these pillars. Clearly, each field must be valued and recognized in its own right, yet it is equally crucial to recognize that it is only through the strength of collective advances of the trio that there is any hope of developing systems that may withstand the demands described earlier.

The articles contained in this book offer a sample of initial, suggestive building blocks towards erecting the pillars of a unified framework and its associated technologies to support creation of Systems of Systems. The contributed articles are organized primarily from the point of view of their generality *vs.* specificity. Chapters 2 through 5 discuss concepts and theories primarily from the point of view of modeling and simulation. Chapters 6-8 present modeling and simulation frameworks and methodologies founded on a variety of system-theoretic, software engineering, and artificial intelligence principles. The remaining chapters present modeling and simulation approaches and techniques, such as discrete-event cellular automata, intelligent systems (e.g., agent modeling and planning, and neuron-based learning), and systems-based software analysis.

### 1.2.1 Discrete-Event Worldview and Distributed Computation

Distributed interoperability and autonomy are key attributes of Systems of Systems. The bottleneck in performance of a simulation of such a complex environment is the ability of the systems to communicate effectively and efficiently with each other across multiple hardware and software platforms.

---

[2] This view is reminiscent of the interplay of art, music, and mathematics described so eloquently by Hofstadter [17].

The demand for efficient interoperability can only be met by minimizing the need for communication between different systems. This can only be accomplished if different systems communicate with each other by means of discrete events. To this end, the authors postulate that the *Discrete-Event System Specification (DEVS)* [30,32] is the only meaningful approach to designing Systems of Systems.

Evidently, this does not preclude individual systems to embrace different M&S methodologies for internal communication. For example, one such tightly coupled system might represent a physical plant that, by itself, is composed of many different parts. It may be appropriate to represent that plant using a differential equation model, whereby each part is encoded as a separate software object. At compile time, the hierarchy of software objects representing the plant will be flattened, leading to a monolithic simulation code representing the entire plant. This system, which is loosely coupled with the other systems of the environment, will communicate with those other systems using a discrete-event protocol.

### 1.2.2 A Brief Account of Artificial Intelligence & Software Engineering

It is important to inquire about the role of AI in the proposed framework, and particularly its relationship with the discrete-event worldview. For example, are the knowledge representation schemes sought by the AI community fundamentally distinct from those of the M&S or SE communities? Does SE pose any constraints on AI and M&S, and vice-versa? Such questions, perhaps philosophical, are plentiful with expectedly a variety of different opinions and answers.

A major branch of AI research has focused on the knowledge representation problem in general, and the qualification and frame problems in particular. In addition to the wide interest and research activities focused on concepts and theories of intelligence and knowledge representation, the AI community has also had a keen appetite to address distributed computation and intelligence modeling. Yet, topics such as scalability, performance, and heterogeneity have not received as much attention. These and other closely related topics – openness, resource sharing, fault-tolerance, and architectural frameworks – have been under investigation primarily from the software engineering community. Indeed, software engineering has introduced a plethora of techniques and paradigms in support of software development (e.g., [1,4,9,19,21,26]) benefiting the AI and M&S communities.

Not surprisingly, the discoveries/inventions made in one area are generally founded on advances achieved in others. For example, advances in automated reasoning, genetic algorithms, fuzzy logic, and neural networks are incorporated into a variety of advanced modeling and simulation environments. Similarly, software engineering principles, practices, and environments have become

indispensable in the development of M&S environments. On the other hand, software engineering tools employ a repertoire of modeling paradigms (Unified Modeling Language [28]) and AI-based techniques to support analysis, design, and realization of distributed heterogeneous systems.

### 1.2.3 Interoperability, Composability, and Distributed Computing

Many different modeling and simulation theories, methodologies, and practices exist for a variety of purposes and domains (cf. [7] for a sample of approaches and application areas). A simple way to differentiate among various approaches is to consider whether a system's structure is characterized as continuous or discrete. From a system-theoretic view, systems can be modeled using continuous-time, discrete-time, or discrete-event formalisms. Each approach offers features that are most suitable for addressing a set of demands. Discrete-event approaches are essential for a variety of systems that are event-oriented (e.g., enterprise resource planning, e-commerce, and semi-autonomous systems). Continuous approaches are most appropriate for the description of many physical plants, such as distillation columns or vehicles, the behavior of which changes continuously over time.

Use of these different modeling approaches, often in isolation, has served the needs of the M&S community well, as long as researchers were content to model and simulate homogeneous systems separately and in isolation. Only recently, the demand has risen for simulating heterogeneous complex Systems of Systems, often using a distributed computing environment. Surely, the previously developed modeling paradigms will continue to serve their domain-specific purposes. However, concentration on specific modeling methodologies has had a tendency to influence the modelers' mind, preventing them from considering facets of their systems that did not lend themselves to being described conveniently using the envisaged modeling approach. Consequently, these approaches often limited the scope of an M&S effort, and thereby created obstacles to dealing with Systems of Systems. It is important to note that, while each of these modeling approaches can be *augmented/extended* to enable heterogeneous, scaleable, composable simulation, the resulting tools will likely be ad hoc and unable to provide the broad encompassing foundation required for SBA.

With the emergence of large-scale simulations demanding a speed-up of several orders of magnitude, distributed computational techniques have become a necessity. Moreover, in recent years it has been realized that there are other motivations demanding distributed simulations. Specifically, due to anytime/anyplace access to (simulation-based) information, enterprise system modeling, knowledge (model) proprietary issues, and increasing need for combined logical and real-time simulations, the discrete-event M&S paradigm has become an attractive candidate for SBA [25].

Based on current and envisaged future simulation needs, the discrete-event framework promises to be the primary choice for unifying continuous and discrete modeling and simulation needs. Among the many approaches to discrete-event modeling and simulation, only the DEVS framework has been shown to embed all classes of dynamic systems. The DEVS framework lends itself naturally to computational efficiency, distributed computing, and component-based software realization such as DEVSJAVA, DEVS/HLA, and DEVS/CORBA) [2, 32].

The ability to support efficient distributed computation based on system-theoretic mathematical underpinnings is particularly promising in achieving model validation, simulation verification, and accreditation (VV&A). Due to the increasing complexity and time-critical nature of systems to be simulated, support for VV&A has become indispensable to modeling and simulation and more generally to SBA.

### 1.2.4 Continuous and Discrete-Event Modeling and Simulation

Since the 1970s, many approaches to combined continuous and discrete modeling and simulation frameworks have been proposed and developed (e.g., [5,11,12,14,16,17,23,29,30,33]). DEV&DESS [23] provides a uniform framework to model continuous and discrete-event models. With the discrete-event system specification, the dynamics of a system can be represented in terms of atomic and/or hierarchical coupled models. Input, output, and state sets as well as functions operating on these sets constitute atomic models. Functions are used to account for how the models are to respond to inputs, change states, and generate outputs. Hierarchical coupled models are used to synthesize complex models from simpler ones. These models communicate with one another through their input/output couplings. Both atomic and coupled models must have their input, output, and state trajectories being piecewise constant. To support larger classes of trajectories, two extensions of the DEVS formalism (Quantized DEVS and GDEVS) have recently been introduced.

These approaches further extend hybrid modeling and simulation of continuous and discrete-event systems. With the Quantized DEVS approach [33], continuous models, mapped into discrete-event descriptions at any chosen level of granularity, can be composed with ordinary discrete-event models. The Generalized Discrete Event Specification [16] has been developed to support the characterization of hybrid dynamical systems having piecewise linear input/output segments. With this approach, input, output, and state trajectories can be represented as piecewise polynomial segments.

Yet, in spite of the sheer generality of the DEVS approach to modeling, also the DEVS modeler is not immune to watching the world through a tinted pair of glasses, and interpreting what he or she observes using a limited, namely DEVS-indoctrinated, world view. The fact that a DEVS model can embed a subsystem that is described by differential equations does not adequately account for the

needs of a modeler asked to provide such a differential equation model. The derivation of a set of differential equations describing a large electronic circuit or a complex multi-body system is a formidable task in its own right, a task that is not addressed by the DEVS formalism.

To this end, an object-oriented continuous system modeling methodology based on system theory was developed [11] and recently standardized [20]. Modelica [20] embraces many of the same principles that govern the design of DEVS. It also distinguishes between atomic and coupled models. Just like DEVS models, Modelica models are homomorphic, i.e., the interface of a coupled model is indistinguishable from that of an atomic model.

The most important difference between the two modeling methodologies is in the handling of inputs and outputs. Whereas DEVS models clearly distinguish between the two types of terminal variables, Modelica models do not. Terminal (interface) variables of a Modelica model are declared as terminal variables, not as either inputs or outputs. For example, an electrical resistor model may have two terminal variables: voltage and current. However, whether the current is input ($U = R \cdot I$) or output ($I = U/R$) depends on the environment in which the model is embedded, and therefore, the modeler should not be forced to make this choice ahead of time. The same model should be usable in both situations. DEVS models are based on the principle of cause and effect. An input event causes an output event to happen either immediately or later. Modelica models are based on the essentially acausal nature of physics (there is no physical experiment that can distinguish between a drunk driver driving his car into a tree and a tree driving itself into the car of the drunk driver). All variables are simultaneously dependent on each other.

Few M&S methodologies make a clear distinction between the underlying principles of *modeling* and *simulation*. Both DEVS and Modelica make this distinction. The purpose of the *modeling methodology* is to support the modeler in organizing his or her knowledge about the system to be described. The purpose of the *simulation methodology* is to apply input behavior to a given model for the purpose of generating output behavior.

Modelica is a pure *modeling methodology*. There is nothing in the Modelica specification that indicates how the model, once specified, is to be used in a simulation, or even, how the model ought to be translated into (interpreted by) a simulation code. Consequently, there is nothing in the Modelica specification that would prevent a Modelica model from being translated into a DEVS atomic model to be used as one monolithic node within a discrete-event simulation.

## 1.3 Simulation Based Acquisition

Simulation Based Acquisition (SBA) [25] is a relatively new and evolving initiative that has gained considerable attention within various United States

federal agencies and business communities since its announcement in 1995 [10]. SBA was defined and adopted in 1997 by the Acquisition Council of the DoD EXCIMS as "an acquisition process in which U.S. Department of Defense and industry are enabled by robust, collaborative use of simulation technology that is integrated across acquisition phases and programs". Other initiatives similar to SBA have been promoting processes capable of minimizing cost, just-in-time delivery, supporting system evolution through reuse, and adhering to the principles of system engineering (e.g., [8]).

The expectations for SBA are to substantially improve the acquisition process through making early informed decisions, reducing risk, optimizing system performance *vs.* ownership cost, timely delivery, improving modularity/reuse, and supporting unprecedented information sharing among its stakeholders. It is important to note that the need for anytime/anyplace access to information is, arguably, the most significant driving force behind SBA. Based on this premise, SBA may be viewed as a means to seek a new breed of *what-to*, *why-to*, *know-to*, and *how-to* knowledge supporting the creation of Systems of Systems. Indeed, based on the need for anytime/anyplace information, a suitable SBA framework must explicitly enable creation, use, and transformation of *heterogeneous layered knowledge*.

Therefore, the success of SBA, similar to that of its earlier contemporaries, will hinge upon its ability to promote scalability, heterogeneity, openness, resource sharing, and fault-tolerance. Whereas such traits in their abstract forms are meanwhile universally accepted [4,13], there does not exist any accepted architectural framework to enable their realizations for SBA. Hence, based on the proposed unified approach and the suggested seven-layer architecture (cf. Section 1.2), a partial candidate architecture can be outlined as specified in Figure 1.2.

Obviously, it is beyond the scope of this article to undertake a complete exposition of SBA specification and architecture, and to provide a blue print for its realization. Nonetheless, the authors find it instructive to exemplify the proposed unified approach to SBA within the periphery of three layers of the seven-layer architecture (see Section 1.2). In the following paragraphs, a collective set of elements is identified and discussed for each of the three layers and their relationships among each other.

To describe Figure 2, let us begin with the modeling layer. In the previous section, the needs/requirements for continuous and discrete-event modeling were discussed. The application of SBA across the information technology spectrum demands modeling from the enterprise level (e.g., engineering resource planning) down to the elementary level (e.g., energy transportation). The simulation layer responsibility is to bring about seamlessly the overall behavior of a suite of models that may be characterized in different forms (continuous, discrete-event, and discrete-time). The simulation layer is to support execution of candidate composition of continuous and discrete models in suitable alternative modes using a variety of simulation protocols (e.g., message passing *vs.* publish and subscribe). Behavioral manifestations of

models can be achieved via specialized simulation techniques executing in possibly both centralized and distributed modes. Adopting distinct layering of modeling and simulation promotes separation of knowledge incorporation from the demands of software engineering. Such separation plays a central role in achieving SBA, since it provides choice and flexibility in the selection of the elements of the modeling and simulation layers.

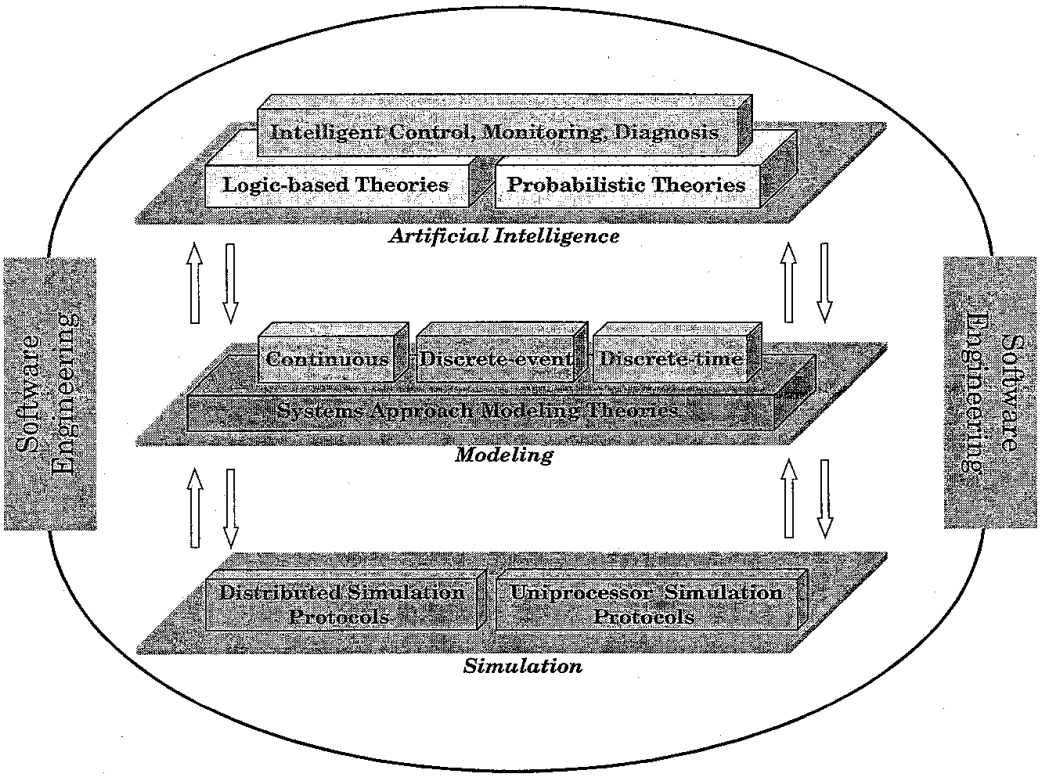


Figure 1.2: A Synergistic view of AI, M&S, and SE

The layer designated as Artificial Intelligence accounts for separation of modeling concerns. Rather than directly incorporating "intelligence" into the models represented at the modeling layer, intelligence features are to be represented distinctly. Intelligence behavior, such as monitoring, is modeled in a modular fashion in such a way as to be composed with the models from the modeling layer. It should be noted that the advocated separation of concerns, while seemingly rather simple, is difficult to accomplish in practice. The only known promising approach to facilitate the realization of separation of concerns is through the principles of software engineering.

The role of software engineering is emphasized through the ring that surrounds the three layers. The ring suggests the essential role that software engineering plays in the specification of the SBA architectural framework and its realization. As suggested earlier, scalability, heterogeneity, openness, resource sharing, and fault-tolerance traits are to be achieved by adhering

closely to sound software engineering principles. Indeed, without exercising software engineering principles systematically and vigorously, the realization of SBA is likely to remain an unfulfilled dream.

It is our hope that the proposed unified approach and the seven-layer architecture lives up to its expectation and advances a united, broader, planned use of simulation assets (e.g., models, technologies, and methodologies) in accordance with the SBA vision.

## 1.4 Conclusions

Scientific and engineering communities alike are facing unprecedented challenges to bring about systems capable of supporting globalization of economies, cultures, etc. Simulation Based Acquisition promises to be the primary framework for realization of a wide range of futuristic engineered systems. Indeed, SBA is expected to be the catalyst in realization of distributed, complex, highly information rich systems by fostering integration of similar and disparate science and engineering technologies. However, existence of totally simulation-based engineered systems will require a new breed of engineering and scientific paradigms governed under an extensible, resilient scientific and engineering framework. Technologies developed in a synergistic fashion based on a unified framework will bring about a "world without borders". The authors argued that without a unified approach founded on artificial intelligence, modeling & simulation, and software engineering, SBA's highly regarded promises are unlikely to be realized. In this setting, the suitability of systems theory was discussed as the unifying approach to represent dynamical systems. Furthermore, the paper illustrates how Discrete Event System Specification and Modelica can support model interoperability and composability while supporting distributed modeling (different model entities can be created and maintained easily by different modelers) and distributed simulation (distributed computing across multiple heterogeneous platforms). Both modeling approaches also lend themselves naturally to the incorporation of a controlling artificial intelligence layer.

## References

[1] Andrews, G.R., *Concurrent Programming: Principles and Practices*, 1991, Redwood City, CA, Benjamin/Cummings, xvii+637.

[2] Arizona Center for Integrative Modeling and Simulation, Software Tools, http://www.acims.arizona.edu/SOFTWARE/software.html, 2000.

[3] Braham, R. and R. Comerford, *Sharing Virtual Wolds.* IEEE Spectrum, 1997, **34**(3), p. 18-19.

[4] Booch, G., *Object-Oriented Design with Applications*, 1994, Redwood City, CA, Benjamin/Cummings.
[5] Cellier, F.E., *Combined Continuous/Discrete System Simulation by Use of Digital Computers: Techniques and Tools*, 1979, ETH Zurich.
[6] Cellier, F.E., *Continuous System Modeling*, 1991, Springer Verlag, xxvii + 755.
[7] Cloud, D.J. and L.B. Rainey, *Applied Modeling and Simulation: An Integrated Approach to Development and Operation*, Space Technology Series, 1998, McGraw Hill. xviii+712.
[8] Cutkosky, M.R., et al., *PACT: An Experiment in Integrating Concurrent Engineering Systems,* IEEE Computer, 1993, **26**(1), p. 28-37.
[9] DMSO, *Runtime Infrastructure*, http://hla.dmso.mil/hla/rti, 1997.
[10] DoD-5000.1, *DoD 5000.1, Defense Acquisition*, 1996, U.S. Department of Defense
[11] Elmqvist, H., *Structured Model Language for Large Continuous Systems*, 1978, Lund Institute of Technology.
[12] Elmqvist, H., F.E. Cellier, and M. Otter. *Object-oriented modeling of hybrid systems*, in *Proceedings of European Simulation Symposium*, 1993, Society for Computer Simulation.
[13] Emmerich, W., *Engineering Distributed Objects*, 2000, John Wiley & Sons. xv+371.
[14] Fishwick, P.A., *Simulation Model Design and Execution: Building Digital Worlds*, 1995, Prentice Hall.
[15] Genesereth, M., N.J., Nilsson, *Logical Foundation of Artificial Intelligence*, 1987, San Mateo, CA, Morgan Kufmann, xviii+405.
[16] Giambiasi, N., B. Escude, and S. Ghosh, *GDEVS: A Generalized Discrete Event Specification for Accurate Modeling of Dynamic Systems,* SCS Transactions, 2000.
[17] Ho, Y.C., *Special Issue on Discrete Event Dynamic System,* IEEE Proceedings, 1989.
[18] Hofstadter, D.R., *Godel, Escher, Bach: An Ethernal Golden Braid*, 1979, Vintage Books Edition, May 1989, xxi+777.
[19] Javasoft, *Java Technology*, http://java.sun.com, 2000.
[20] Modelica, http://www.modelica.org/, 2000.
[21] OMG, *CORBA/IIOP 2.2 Specification*, http://www.omg.org/corba/corbaiiop.html, 1998.
[22] Orfali, R., et al., *The Essential Distributed Objects Survival Guide*, 1995, John Wiley & Sons.
[23] Praehofer, *Systems Theoretic Foundations for Combined Discrete Continuous System Simulation*, in *Institute of Systems Science, Department of Systems Theory and Information Engineering*, 1991, Johannes Kepler University.
[24] Sarjoughian, H.S. and B.P. Zeigler, *DEVS and HLA: Complementary Paradigms for M&S.* Transactions of Society for Computer Simulation, vol. 17, no. 4, pp. 187-197, 2000.
[25] SBA, *Simulation Based Acquisition: A New Approach*, 1998, Defense Systems Management College, Report of the Military Research Fellows DSMC.
[26] SEI, *Software Engineering Institute*, http://www.sei.cmu.edu, 2000.
[27] Sorid, D. and S.K. Moore, *The Virtual Surgeon,* IEEE Spectrum, 2000, **37**(7), p. 26-31.
[28] UML, *Unified Modeling Language,* http://www.rational.com/uml/index.jtmpl, 2000.

[29] Wymore, W., *Model-based Systems Engineering: An Introduction to the Mathematical Theory of Discrete Systems and to the Tricotyledon Theory of System Design*, 1993, Boca Raton, CRC.

[30] Zeigler, B.P., *Toward A Formal Theory of Modeling and Simulation: Structure Preserving Morphisms,* Communications of the ACM, 1972, **19**(4), p. 742-746.

[31] Zeigler, B.P., H.S. Sarjoughian, and S. Vahie, *An Architecture For Collaborative Modeling and Simulation*, in *11th European Simulation Multiconference*, 1997, Istanbul, Turkey.

[32] Zeigler, B.P., H. Praehofer, and T.G. Kim, *Theory of Modeling and Simulation.* Second Edition, 2000, Academic Press.

[33] Zeigler, B.P., H.S. Sarjoughian, and H. Praehofer, *Theory of Quantized Systems: DEVS Simulation of Perceiving Agents,* Cybernetics and Systems, 2000, **31**(6), p. 611-647.