A Swarm Ontology for Complex Systems Modeling

Roger Burkhart
Deere & Company
One John Deere Place
Moline, IL 61265
burkhartrogerm@johndeere.com

Draft Working Paper

Symposium on Complex Systems Engineering
RAND Corporation, Santa Monica, CA
January 11-12, 2007

ABSTRACT

Modeling and simulation will provide crucial capabilities for complex systems engineering, but existing languages, frameworks, and tools fall short of both fundamental and practical needs. This paper describes work-in-progress to fill these needs, based on a foundation of logic-based description and a system of concepts to describe complex dynamic systems. The modeling framework can include mappings to visual forms of model representation such as the OMG Systems Modeling Language (OMG SysML™), and to executable forms of multi-agent simulation such as those implemented in the Swarm simulation system.

1. Introduction

Modeling and simulation will be fundamental to the practice of complex systems engineering, since the inherent nature of complex systems makes them difficult to analyze by other means. Large numbers of elements that interact in decentralized ways, giving rise to emergence across multiple levels of causation, make a complex system difficult to understand and analyze, much less engineer, using traditional techniques of closed-form analysis or prediction. Agent-based modeling is an alternative technique that has grown up across many disciplines. It relies on computational models and experiments to explore the possible behavior of a complex system. The Swarm simulation system [3] was an early toolkit to help researchers build such computational experiments. The user communities of Swarm and related toolkits have shown the dramatic growth of agent-based modeling as a research technique across many fields, from ecology to economics and social systems to optimization and search.

Just because a system is complex, however, does not mean that it is incapable of being engineered, or that traditional methods of systems engineering management are not required to define and trace the realization of a system and its relationships to stakeholder needs and requirements. A range of modeling and documentation forms have been built up across many decades to support the multi-disciplinary practice of systems engineering, from requirements management to system structure and interfaces to system behavior models. A collection of systems engineering models, along with visual diagram forms adapted from the Unified Modeling Language (UML), has recently been standardardized by the Object Management Group (OMG) as the OMG Systems Modeling Language (OMG SysML^{TM}) [8].

This paper will outline some work-in-progress to build modeling frameworks that can combine models for system specification with models that express the dynamic structure and behavior of a complex system. Such frameworks can be useful for development of systems to be deployed in a complex and evolving environment, or which incorporate distributed and decentralized control, problem-solving, adaptation, or optimization as integral parts of an engineered system.

2. Outline of approach

Some motivations and elements of the design approach are provided in the form of presentation slides at [5] and [6]. Briefly, the ability to represent multiple levels of system structure in a complex systems modeling framework such as Swarm can be a good match for the multi-level description of system structure in OMG SysML.

Both these modeling frameworks, however, currently lack the formality and abstraction above the level of implementation which is needed for them to scale across multiple communities and development phases as needed for development of real systems. Swarm and related agent-modeling toolkits are implemented at the level of programming language libraries (the Objective C language in the case of Swarm, Java for many of the more recent toolkits), as driven by their primary goal to drive executable simulations. SysML is derived from the Unified Modeling Language (UML), also standardized by OMG, and combines visual diagrams for human communication with metamodels that can be exchanged in digital form across modeling tools. UML and SysML, however, provide only a limited set of standardized behavior models (procedural operations, activity flow diagrams, or finite-state machines), and their newly standardized abilities to describe hierarchical system structure and interconnection of system elements are still incomplete and in need of further specification and formalization.

The core of system structure description in UML and SysML, however, can be mapped to formal semantic models under various forms of logic-based languages, such as the description logics of the Resource Description Framework (RDF) and the Web Ontology Language (OWL) being standardized by semantic web initiatives, or full first-order logic in the Common Logic language being standardized by the International Organization for Standardization (ISO). The OMG Ontology Definition Metamodel [9] contains metamodels for these and other languages for logic-based description, along with the beginnings of mappings across them and with UML.

A notable characteristic of logic-based languages is their neutrality with respect to ontology. An ontology is a system of concepts suitable for describing some domain of interest. In languages such as UML and SysML, such concepts are defined in terms of basic types for the elements in some domain (classes in UML, or blocks in SysML), along with properties that relate these elements to each other.

A modeling framework for complex systems can be based on a commitment to basic concepts for the description of dynamic systems. The Swarm system [2][3][4] contained a beginning structure of concepts for the description of complex dynamic systems, with dynamic populations of interacting elements organized into multiple levels of potentially emergent structure. These basic concepts can be taken out of their original context in a language toolkit implementation, and defined as a formal ontology in a logic-based language.

A Swarm ontology for Dynamic System Description (SwarmDSD) can supply the definitions of core concepts to meet the many difficult challenges in complex systems modeling. A position paper for an earlier workshop [1] outlined some of these challenges, many of which are still not fully addressed by simulation frameworks such as Swarm. In the Swarm toolkit, a “swarm” is a structural element of a model that contains both a collection of interacting elements and a schedule of activity over these elements. A generalization of a swarm to include an internally regulating schema, itself produced as a dynamic structure during the lifetime of the swarm, can greatly extend the expressibility of a complex system model [5]. The schema can include declarative statements of constraints to be enforced by the model, thus integrating sophisticated forms of functional and logic programming as in recent multi-paradigm programming environments [7].

A complex system description based on a Swarm ontology could be used either to specify an engineered system and its environment at initial levels of completeness or abstraction, or to define its working mechanisms in sufficient detail that the model could be executed as a simulation. With sufficient detail, Swarm-based models could also become an executable model of computation in its own right. This was one of the original design goals for the Swarm system [2], but never realized in the current implementation. With this level of specification, a Swarm system could become an abstract virtual machine for its own execution, without reliance on any other language except to provide a substrate for implementation.

The current design of an ontology that could also drive an executable computation is based on a progression of bootstrap layers. This progression starts with a simple form of local, graph-based representation, but moves up to swarms of distributed Swarm models interacting across networks. The underlying representation is a semantic network roughly similar to RDF [9], but stripped to an essential core for internal representation of an abstract machine. The specific concepts of a Swarm ontology can then be defined in a self-contained logical system expressed on top of this underlying representation.

Many engineering issues remain to be worked out within such a framework, as well as the formal definitions that comprise its conceptual plan. Future versions of this or other papers will be needed to outline both conceptual and implementation aspects of an executable Swarm ontology. Once realized, however, a Swarm description of itself, in various forms suited to communicate to both humans and computing machines, could become a primary means to specify and document its own design.

REFERENCES

[1] Burkhart, R., "Effective Description of Enterprise Processes," in *OOPSLA '93 Workshop on Behavioral Semantics in Object-Oriented Information Modeling*, Kilov, H., Harvey, W., eds., Institute for Information Management, Robert Morris College, 1993 (available at http://citeseer.ist.psu.edu/burkhart93effective.html, but pages are in reverse order)

[2] Burkhart, R., "The Swarm Multi-Agent Simulation System," Position Paper for *OOPSLA '94 Workshop on "The Object Engine*, 1994 (available at http://www.swarm.org/wiki/Papers_on_Swarm)

[3] Minar, N., Burkhart, R., Langton, C., Askenazi, M., *The Swarm Simulation System: A Toolkit for Building Multi-Agent Systems*, Santa Fe Institute Working Paper 96-06-042, Santa Fe, NM, 1996 (available at http://www.santafe.edu/research/publications/wplist/1996)

[4] Burkhart, R., Schedules *of Activity in the Swarm Simulation System*, Position Paper for OOPSLA '97 Workshop on OO Behavioral Semantics, 1997 (available at http://www.swarm.org/wiki/Papers_on_Swarm)

[5] Burkhart R., "Standardizing an Agent Life-cycle Model," *Agent 2004 Conference on Social Dynamics: Interaction, Reflexivity and Emergence*, (slides in proceedings at http://agent2004.anl.gov/proc.html)

[6] Burkhart R., "Modeling System Structure and Dynamics with SysML Blocks," *Frontiers in Design & Simulation Research 2006*, Georgia Institute of Technology, 2006 (slides at http://www.pslm.gatech.edu/events/frontiers2006/proceedings/2006-03-16-Frontiers2006-Burkhart-2.pdf)

[7] Van Roy, P., Haridi, S., *Concepts, Techniques, and Models of Computer Programming*, MIT Press, 2004

[8] Object Management Group, *OMG Systems Modeling Language (OMG SysML^{TM}) Specification, Final Adopted Specification*, OMG document number ptc/06-05-04, 2006 (available at www.omgsysml.org)

[9] Object Management Group, *Ontology Definition Metamodel*, OMG document number ad/06-05-01, 2006 (available at www.omg.org/ontology)