01 Apr 2008

Executable Modeling for System of Systems Architecting: An Artificial Life Framework

Kilicay-Ergin Nil

Cihan H. Dagli
Missouri University of Science and Technology, dagli@mst.edu

SysCon 2008 – IEEE International Systems Conference
Montreal, Canada, April 7–10, 2008

Executable Modeling for System of Systems Architecting: An Artificial Life Framework

Nil Kilicay-Ergin[1], Cihan Dagli[2]
[1]Postdoctoral Research Fellow,
Engineering Management & Systems Engineering Department,
Missouri University of Science & Technology, Rolla, MO, USA
[2]Proffesor of Engineering Management & Systems Engineering Department
Missouri University of Science & Technology, Rolla, MO, USA

Abstract *– There is a diversity of frameworks and methodologies for enabling architecture developments. Static representation frameworks provide a standardized way to communicate the architecture to stakeholders, but do not provide means to analyze the system states and system behavior. Therefore, there is a need to convert static representation frameworks to executable models. The aim of this paper is to present Artificial Life approaches as a methodology for understanding behavior of System of Systems. For this, an Artificial Life based framework for modeling System of Systems is presented. The framework comprises cognitive architectures embedded in multi-agent models. Financial markets are selected as an analysis domain to demonstrate the framework since they are a good example of self-organizing systems that are nonproprietary and exhibit emergence on a grand scale. From the Artificial Life Framework trader-based architectures are formulated as models to analyze system level behavior. The Artificial Life based framework provides a flexible way of modeling sub-systems of System of Systems and it captures the adaptive and emergent behavior of the system.*

Keywords *– Architecting Frameworks, Executable Modeling, Artificial Life, Financial Markets*

I. INTRODUCTION

A dynamically changing meta-architecture for System of Systems can be defined as a collection of different complex adaptive systems that are readily available to be plugged into evolvable net-centric communications architecture. The challenge is to identify the right collection of systems that will collaborate to satisfy the client requirements. This shifts the focus from component and individual system level architecting to meta-level architecting. System-of-systems (SoS) architecture is not just the technical architecture of the system, but the higher level meta-architecture that integrates the physical architecture, the stakeholders, development and deployment considerations into an integrated framework.

There is a diversity of architecture frameworks and methodologies for enabling architecture developments. The fundamental goal of all these enablers is to capture a detailed description of the SoS architecture based on different architectural views, develop an implementation process by utilizing available technological options and knowledge, and then conduct performance evaluations. The architecture enablers can be classified into three types based on their support on the architecture design process. These are mainly enablers for static representation of the architecture, enablers for creating an executable model of the architecture and enablers for logical, behavioral and performance evaluation of the architecture [10].

A good balance of heuristics, analytical techniques and integrated modeling is necessary as architecting tools for SoS. Specifically, model-centric frameworks and executable models become important tools for SoS analysis and architecting as they provide insights to SoS architecture behavior. Static representation frameworks [6] provide a standardized way to communicate the architecture to stakeholders, but do not provide means to analyze the system states and emergent behavior. Therefore, there is a need to convert static representation frameworks to executable models. Colored Petri-nets, state models have been used as executable models [1, 6].

It is feasible to understand any System of Systems as an artificial complex adaptive system. The relation of SoS characteristics and Complex Adaptive Systems (CAS) characteristics are outlined in [2]. Artificial life tools have been successfully used in analysis of Complex Adaptive Systems. Since System of Systems is collections of several Complex Adaptive Systems, we can utilize these tools for analysis of SoS behavior.

The aim of this paper is to present Artificial Life framework as a methodology for generating executable models for SoS and analyze the affect of different architecture changes on the overall system behavior. Financial markets are selected as an analysis domain to demonstrate the framework since they are a good example of self-organizing systems that are nonproprietary and exhibit SoS characteristics, specifically emergence on a grand scale. The rest of the paper is organized such that Section 2 describes the Artificial Life framework for System of Systems analysis, Section 3 demonstrates the framework for analysis of financial markets and Section 4 provides system behavior analysis for different sub-system architectures. Finally, section 5 concludes the paper with the value of this framework and future research directions.

II. APPPROACH: AN ARTIFICIAL LIFE FRAMEWORK

The framework as illustrated in Figure 1 consists of several layers to capture different architectural views of the

System of Systems. It consists of several layers for modeling different components of systems. Layering the framework is important for keeping the architecture simple at each layer. Therefore, the framework consists of several layers: computational intelligence tools, mechanism modules, cognitive architecture, agent level, environment level and system level.

Computational intelligence and other analytical methods are utilized to design mechanism modules to represent the lowest level architectural components of sub-systems. These mechanism modules can vary from learning capabilities to other domain specific capabilities such as attention, bias, associative memory etc. These modules are used to design different sub-system architectures. Cognitive architectures [5] are utilized at the cognitive level because they represent a promising approach to explaining mental processes and human behavior with error generation mechanisms. Sloman's cognitive architecture [9] is selected at this level because it is a generic cognitive architecture framework that has the flexibility and modularity to be integrated with multi-agent architectures. Besides, various sub-system level architectures can be created utilizing this architecture. The cognitive architecture consists of a reactive layer which responds to environment states immediately. The deliberative layer conducts reasoning activities such as planning, scheduling etc. The meta-management layer controls the activities of the lower layers. The architecture provides means to model the intelligence and independent behavior capabilities of the sub-systems.

On the other hand, multi-agent models [3] are a suitable tool for modeling SoS because they provide means of integration for the social, information and physical components of SoS. At the agent level, various agent architectures are designed utilizing the cognitive architecture as the blueprint. These agents (Sub-systems) need physical interface to function. The environment level captures the physical architecture of SoS. At this level the dynamics of the environment such as physical laws, rules of engagement of the environment, operational context is specified. These rules model the static characteristics of the environment and scope the type of behaviors that are allowed in that environment. The artifacts that agents can utilize to communicate the semantics of system laws among themselves are also identified at this level. Finally, the selection criteria for adaptation are also determined for selecting the successful actions in that environment.

The environment model of the framework and the way the agents are connected to the environment model create the meta-architecture of the SoS. The system level of the framework (multi-agent model) creates an executable model of the meta-architecture, which captures the emergent system level behavior of the meta-architecture. By utilizing this framework as a blueprint, different architecture alternatives can be tested. Following section demonstrates the Artificial Life framework for analysis of financial markets by extracting two different trader-based architectures.

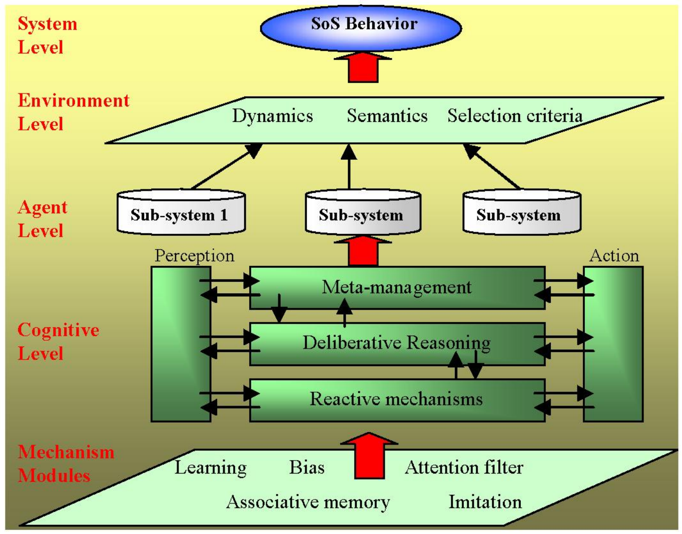

Figure 1. Artificial Life Framework

III. MARKET SYSTEMS

System-of-systems are not always created with a directed mission, can also be created by the collaboration of its users [4]. One example of such a system is the financial market organization. The market organization and the rules for trading at the market create a meta-architecture. Traders are connected to the market trading grid through some communication architecture and different system dynamics are observed based on trader behaviors. The way the traders are connected to this architecture and their behavior form the market SoS architecture. Other SoS market architectures can be created by connection of other systems such as trust funds etc. Figure 2 illustrates the market meta-architecture concept.

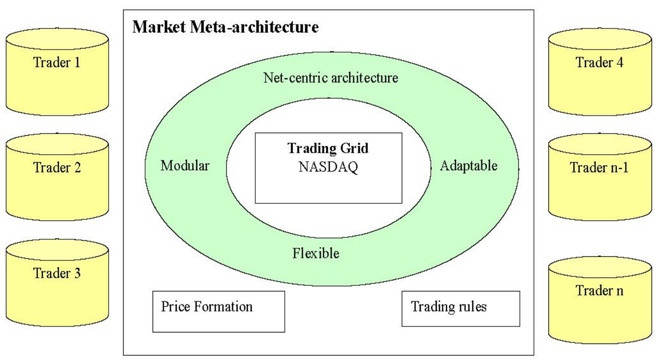

Figure 2: Market Meta-architecture

Analysis architecture for financial markets is designed to demonstrate the framework as an executable model. Different architectures can be designed based on the system abstraction level. Extensive review of artificial stock market studies can be found in [8]. The objective is to understand the relationships among system components and how sub-system components affect overall system behavior.

Two mechanisms are modeled for trader architecture; the learning mechanism and bias mechanism. Learning classifier systems are used from the computational intelligence toolbox to design the learning mechanism. Traders' trading strategies are encoded as rule-based (condition; action) form. The mechanism selects the decision action, rewards the successful actions afterwards and utilizes genetic algorithms to discover new trading strategies for the evolving market environment. Detailed description of the learning mechanism module design can be found in [4].

Markov process based model is used to design the bias mechanism. The bias mechanism mimics the conservative and trend following biased behavior of traders observed in real markets. The bias mechanism assumes the market is in two states, the conservative and trend following states and selects the decision action based on the selected market state. Detailed description of the bias mechanism module can be found in [4].

At the cognitive level, two different trader architectures are designed to analyze the effect of these mechanisms on the overall system behavior. In one model, trader cognitive architecture consists of the learning mechanism at the deliberative reasoning layer. In another model, trader architecture combines the bias model and the learning mechanism. Traders have trading strategies that evolve based on market dynamics, but they also have a bias model that interrupts their learning mechanism. Figure 3a and Figure 3b illustrates the cognitive architecture alternatives tested to analyze the market system behavior.

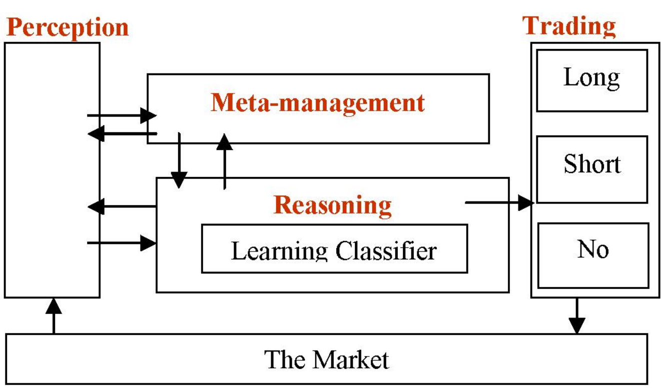

Figure 3a. Trader Architecture 1

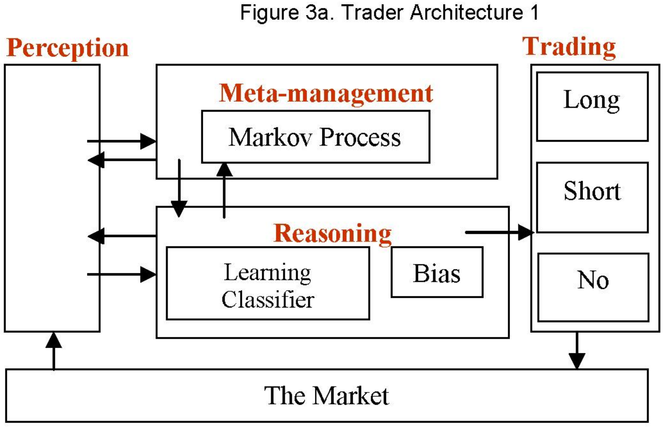

Figure 3b. Trader Architecture 2

At the environment level, the financial market trading rules and market price formation mechanism provides a physical interface for traders to function. In this market, there is one stock for trading and the stock gives dividends at certain intervals. Investors take one of three different decisions: They can take a long position where they buy stock at the current market price and then sell at a higher price to make profit. They can take a short position where they sell the stock at the current price and then buy the stock at a lower price (cover position) to make profit. They can also choose not to trade.

Response to excess demand determines the price of the stock. Market demand and supply from each trader are summed, and if there is excess demand, the price of the stock is increased by a constant amount. If there is excess supply, the price is decreased by α amount.

At the meta-architecture level, aggregate demand/supply from all traders connected to the trading grid is calculated. The market price is adjusted and other market indicators are updated and revealed to all traders. This cycle continues through each trading period.

Different market dynamics are observed at the systems level based on the two different sub-system architectures outlined in this section. Following section discusses the results.

IV. SYSTEM BEHAVIOR ANALYSIS

The two cognitive architectures are tested by generating executable models using AnyLogic simulation software. For both cases 100 traders are generated and the simulations are conducted for 1000 trading periods. For artificial financial markets, a benchmark is useful for comparing the simulation results. Homogenous rational expectations equilibrium is utilized as the benchmark for comparison of results [8].

Figure 4 shows the price formation for the first trader architecture alternative which consists of only the learning mechanism. The rational expectations equilibrium model (REEM) price is also shown on the graph for comparison. The dark thick red line represents the model price formation, whereas the thin green line represents the REEM price formation for the same market structure.

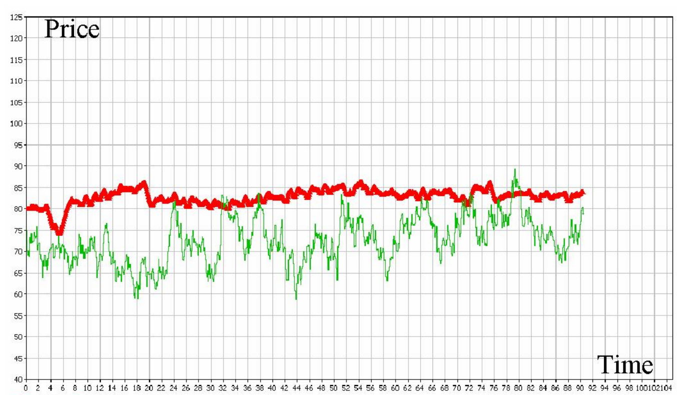

Figure 4: Effect of Trader Architecture 2 on Market Dynamics

In this case, the model price and REEM price follow each other closely. The price dynamics are also similar where the model price increases as the REEM price increases. However, there are periods when the model price deviates from the REEM price.

Figure 5 shows the price formation for the second trader architecture alternative which consists of a learning mechanism and a bias mechanism. The probability of a trader using bias mechanism is set to 80%. In this case, the model price at some point in the simulation gets close to REEM price, but after some time the price drastically deviates from the REEM price. Also, the model price dynamics moves towards an upward trend, whereas the REEM price dynamics fluctuates. Since the bias mechanism dominates the trader behavior, the drastic deviation from the REEM price is an expected outcome. The learning mechanism is not successful enough to pull the market back to the REEM price dynamics.

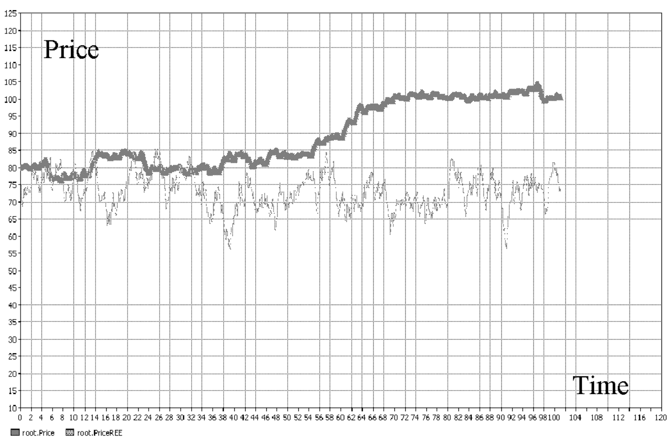

Figure 5. Effect of Trader Architecture 2 on Market Dynamics

The statistical properties of the artificial price series indicate whether the artificial series can successfully capture the real market characteristics. The statistical properties of the price series generated from the two different system architectures are provided in Table 1. The statistical analyses show that the price is not normally distributed in both systems because the kurtosis and skewness values are non-zero values. It is known that real market time series are not normally distributed. Therefore, both system architectures capture a portion of the real market characteristics. System architecture 1 which is based on trader architecture 1 has positive kurtosis value (leptokurtosis) which confirms that this market exhibits the fat tail phenomenon observed in real markets. Therefore, the fist architecture captures additional real market characteristics and conforms in some respects to real-life financial market behavior.

The Dickey-Fuller test is another test to analyze the artificial time series and shows whether a unit root is present in an autoregressive model. Time series are autoregressive models. If the unit root is present, the time series is said to have a stochastic trend. The Dickey-Fuller tests conducted on both artificial time series reveal that both scenarios tend to have a stochastic trend and are not linearly related. This test also reveals that the Artificial Life models conforms in some respects to real-life financial market behavior.

Table 1: Statistical Properties of the Artificial Price Series

	Kurtosis	Skewness	Mean	Standard Deviation
Architecture 1	4.33	-1.48	82.85	1.85
Architecture 2	-1.65	0.29	89.14	9.38

V. CONCLUSION

The need for better upper level descriptive and analysis frameworks is a challenging area in SoS studies. The motivation behind this study was to develop an analysis framework that integrates physical, information, cognitive and social components of SoS. The study presents Artificial Life based framework for modeling and analysis of emergent behavior of SoS architectures. The framework comprises cognitive architectures embedded in multi-agent models. Various computational intelligence tools can be utilized to design mechanism modules, which can be incorporated into the cognitive architecture. This type of framework provides a flexible and modular way of modeling sub-systems of System of Systems and captures the adaptive and emergent behavior of the system architecture. Specifically, a combination of deliberative and reactive reasoning provides a flexible architecture for modeling sub-systems of SoS.

Two trader-based system architectures are derived from the Artificial Life framework for analysis of financial market behavior. One of the alternative architectures utilizes learning classifiers as reasoning and learning mechanism. Another combines the Markov process based bias model and the learning classifier based learning model into one hybrid model. Both alternative architectures are embedded into an agent-based financial market model to analyze the effect of the trader architecture on market dynamics. The system behavior analyses reveal that when a learning mechanism dominates the trader behavior, the model price and price dynamics closely follow the REEM price dynamics. When the bias mechanism dominates the trader behavior, the model price and dynamics drastically deviate from the REE price dynamics. Both alternatives illustrate that the models derived from the Artificial Life framework can capture the real market price series characteristics.

The model derived from the framework contributes to understanding the market behavior and potential sources of deviation from efficient market equilibrium. Different architecture alternatives can be designed utilizing this framework. The framework provides a suitable way of testing alternative architectures in terms of physical, social and behavioral perspectives.

This type of framework is especially beneficiary during what-if analysis of systems and can minimize cascading failures of systems by capturing different emergent behaviors of system architectures.

Both structural and executable models are required for comprehension of SoS. Simulation tools that combine various modeling paradigms should be used in analysis of SoS to capture different behavioral views. Future studies of this framework should focus on how the framework can be integrated with structural and other system analysis frameworks.

REFERENCES

[1] Campbell J., Anderson D., Lawton C., Shirah D. and Longsine D., 2005, "System of Systems Modeling and Simulation", Proceedings of Conference on Systems Engineering Research 2005, March 23-25, NJ USA.

[2] Correa Y. and Keating C., 2003 "An Approach to Model Formulation for Systems of Systems", IEEE Conference on Systems, Man and Cybernetics, Vol.4, Oct. 2003, pp. 3553-3558.

[3] Ferber J., 1999, Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence, Addison-Wesley.

[4] Ergin N. H., 2007, "Architecting System of Systems: Artificial Life Analysis of Financial Market Behavior", Dissertation submitted to University of Missouri-Rolla.

[5] Madni A. M., Sage A. P. and Madni C., 2005, "Infusion of Cognitive Engineering into Systems Engineering Processes and Practices", IEEE International Conference on Systems, Man and Cybernetics, Vol. 1, 10-12 Oct. 2005, pp.960-965.

[6] Madwaraj R., Ramakrishnan S., Dagli C. and Miller A.., 2006, "Modeling the Global Earth Observation System of Systems", Industrial Engineers Research Conference, 2006, FL, USA.

[7] Maier M., 1998, "Architecting Principles for Systems-of-Systems", Systems Engineering, Vol. 1, No. 4, pp. 267-284.

[8] LeBaron B., 2006, "Agent-Based Computational Finance," in *The Handbook for Computational Economics, Vol. II*, Judd & Tesfatsion (eds.), Elsevier/North-Holland.

[9] Sloman A., Logan B., 2000, "Evolvable Architectures for Human-Like Minds," in *Affective Minds*, G. Hatano, N. Okada and H. Tanabe (eds.), Elsevier, Amsterdam.

[10] Wagenhals L. W., Haider S., Levis A. H., 2003, "Synthesizing Executable Models of Object Oriented Architectures", Systems Engineering, Vol. 6, pp. 266-300.