INTERNATIONAL JOURNAL OF INTELLIGENT CONTROL AND SYSTEMS
VOL. 14, NO. 1, MARCH 2009, 20-32

Agent-Directed Simulation for Systems Engineering: Applications to the Design of Venue Defense and the Oversight of Financial Markets

Philip S. BARRY, Matthew T. K. KOEHLER, Matthew T. MCMAHON, and Brian F. TIVNAN

Abstract- **This paper briefly introduces the inherent challenges to Systems of Systems engineering. A solution created by the authors is then described, the Infrastructure for Complex-systems Engineering (ICE). The paper concludes with two case studies making use of various aspects of the ICE. The first is an application to large venue protection; the second is an application to the modeling of financial markets.**

***Index Terms*—Systems of Systems, Infrastructure for Complex-systems Engineering, financial markets, large venue protection**

1. INTRODUCTION

With the development of Systems of Systems (SoS) becoming commonplace; developers are faced increasingly with the problem of investing large amounts of money in SoS that cannot be fully specified in a requirements document, and cannot be fully tested or in many cases even prototyped. Compounding the problem is that SoS are not simply very complicated systems like a racecar or a communications satellite. Complicated systems have many parts that interact with each other in nontrivial ways, and can be described using well-understood laws of mechanics and physics. SoS, on the other hand, present a constantly changing topology, evolving over time and space. Furthermore, perhaps the most important component is the human element. Human behavior (e.g., people interacting with each other) in SoS is not as easily modeled as are other system components as humans often create emergent dynamics and capabilities that are often impossible to define *a priori*. This differs from the classical view of systems where humans are often considered outside the system.

With the advent of SoS engineering has come a recognition that new approaches are needed (e.g., [1], [2], [3]and[4]). There has been noteworthy progress in developing paradigms that provide static descriptions of the SoS, hierarchically decomposed views of the enterprise with robust descriptions of the interfaces. However, these analysis techniques do not provide a clear path for gaining insight into the dynamic and evolutionary behavior of SoS.

The mission criticality of many SoS requires that we 'engineer' them to provide a quantitative improvement in desired behaviors and functionality. This effort becomes challenging due to the sheer number of components in fielded SoS and their associated complex behavior argues strongly against using closed forms of analysis. Traditionally, a systems engineer would turn to the standard techniques of operations research for help with this (e.g. multi-attribute utility maximization, and linear programming). We suggest that the employment of agent-directed simulation (ADS) alongside rigorous, proven systems engineering tools and techniques will allow systems engineers to develop and gain new insights into the specification, design, performance, and evolution of SoS. Our argument for ADS is built upon a foundation of work from Wegner, Simon, Doyle, Schelling, Buss, et al., Epstein and Axtell. Wegner [5, 6] showed that as powerful as closed form algorithmic analysis is, that it essentially represents the system as a Turing machine. ADS essentially allows the algorithms to interact, (i.e., ADS as an interaction machine), which is a far more powerful analytic representation. Simon [7] argued that complex systems can be meaningfully represented as collections of subsystems; hierarchies of nearly decomposable systems. Within these collections the state of each subsystem through time is only weakly influenced by the other subsystem. The ability to represent these complex systems (or SoS) in this way makes systems engineering of them possible and creating ADSs of them meaningful and useful. Csete and Doyle [8] stress not only the importance of modules in complex SoS but also the criticality of common interfaces between the components. These interfaces are essential for large collections of systems to function properly. Schelling [9] argued that the interaction of subcomponents of a system, without meaningful centralized control, can have enormous impact on the behavior and performance of the system as a whole.

Moreover, Buss et al. [10] proved that under ideal conditions a model of a SoS (e.g., an ideal SoS is defined as a collections of homogeneous automata with a global control rule that is independent of the states of any of the

With respect to the financial market modeling, the authors gratefully acknowledge: (1) the assistance of both Khalid El-Arini and Neal Rothleder for help defining and producing some functional forms of the statistics and (2) additional support for this research from a Business Network Fellowship at the Santa Fe Institute. This paper is extended from "Agent Directed Simulation for Systems Engineering" published at *SCS 2009 Spring Simulation Multiconference, Agent-Directed Simulation Symposium (ADS'09),* San Diego, CA, March, 2009

All authors are with the MITRE Corporation. 7515 Colshire Dr., McLean, VA, USA (e-mail: pbarry@mitre.org, mkoehler@mitre.org, mcmahon@mitre.org, btivnan@mitre.org).

automata) will be predictable (i.e. the future state of the system is computable) in polynomial time. If, however, the global control rule is not independent of the states of the automata then the prediction problem becomes PSPACE-complete. This means that one can do no better to understand the future states of the system than to simulate it. More recently, Epstein [11] has argued, and demonstrated, that it can be far more insightful to examine the dynamics of a system as it moves through time than to "solve" for the system's equilibria. It is trivial to specify a system that may not be able to obtain its equilibria from plausible initial conditions or that it may take longer than the age of the universe for the system to achieve an equibrium state. This being the case, simulations can prove far more useful from a systems engineering perspective to gain insight into the actual, or meaningful, performance of the system in question (essentially arguing that SoS are broken ergodic systems). Finally, Epstein and Axtell [12], and Epstein [13] argue for a generative nature to our understanding of social systems—"If we didn't grow it, we didn't explain it." They argue that understanding the emergent properties of a SoS (the macro-dynamics) comes from specifying the components and then allowing the system to move forward in time. In this way one generates a sufficiency theorem—*this* specification with *this* set of input is sufficient to generate *this* output. In short, there is a strong theoretical basis for the argument that ADS for SoS engineering is not only useful, but may be one of the only ways to meaningfully understand, design, and engineer SoS.

2. Agent-Directed Simulation for the Systems Engineering of Human Complex Systems

Simulation is frequently the method of choice for researchers (e.g., Carley and Svoboda [14], Epstein and Axtell[12], Levinthal [15], March [16] and McKelvey [17,18] among others) to explore complex dynamics often found in Human Complex Systems [19]. We characterize Human Complex Systems (HCS) as SoS that include active, human participants beyond simple roles of systems operator (e.g, large, public venue including crowd and security personnel; metropolitan area experiencing a pandemic; and financial exchange including traders and regulators). Analyzing HCS requires a refactoring of tools; McKelvey and Cyert and March call for a specific class of simulation for HCS, namely simulation with ADS [18, 20, 21]

While simulations applied to the study of HCS first occurred as much as forty years ago (e.g., Cyert and March [21]), only recently has the approach begun to generate a broader acceptance [22]. Not only special issues but entire journals are now dedicated to simulation and its application to the science of HCS (e.g., Carley [23], Lissack [24] and Gilbert [25]). This acceptance stems from two critical aspects of simulation research: (a) simulation allows researchers to explore the inherent complex dynamics of HCS [22, 26], hence (b) simulation research allows for the conduct of experiments that would typically be impossible or impractical in the physical world [27].

Stressing the value of simulations for theorizing [28], Axelrod [29, p. 23-24] believes that simulation offers a new vehicle for conducting scientific research that differs from induction (i.e., the "discovery of patterns in empirical data") and deduction (i.e., "specifying a set of axioms and proving consequences that can be derived from those assumptions"). On the one hand, simulation research resembles deduction in that simulations start with a set of assumptions. On the other hand, the simulation generates data to be inductively analyzed. Axelrod [29, p. 24] refers to simulation research as "thought experiments" since the assumptions might seem simple but the results are often counter-intuitive (i.e., the nonlinear, macro-level effects of interacting agents known as emergent properties).

Axelrod [29] provides further support for simulation as an alternative to the rational actor / choice assumptions. Because the rational actor / choice assumption allows for deductive, closed-form analysis, researchers are willing to overlook the boundedly rational limitations of their actors [30]. The primary alternative to the rational actor / choice assumption lies in some form of adaptive behavior. Due to the complex effects of social interactions, Axelrod [29] asserts that ADS offers the only vehicle to study sets of actors who possess an adaptive capacity.

3. A Call for Agents in Human Complex Systems

With the growing acceptance of simulation in the design and engineering of HCS due in no small part to March's research, several leading scholars have called for the formal use of ADS (e.g., Anderson [32], Axelrod[29], Dooley [22] and McKelvey [17, 18]). As the primary tool of complexity theorists, ADS assume that agents behave in a stochastic, nonlinear manner and that agents possess a nonlinear capacity to adapt over time. This stochastic, nonlinear behavior of agents is consistent with the stochastic, idiosyncratic microstates of HCS. That is, despite institutional influences [33, 34], strong forces remain to idiosyncratically steer both the behaviors of individuals and the conduct of aggregate processes [31]. Among others, such forces might include unique organizational cultures, the unique set of suppliers and customers (i.e., organizations are each embedded within a unique social network) and the unique interaction network of different individuals each with his/her own personal history in different contexts. Therefore, agent activity in an ADS can offer an excellent representation of the adaptive and idiosyncratic behavior of an HCS and that of its human agents.

4. An Infrastructure for the Engineering of Human Complex Systems

As argued *supra*, an ADS typically represents the *only* way one may experiment and test the SoS in question.

With that in mind, we developed the Infrastructure for Complex-systems Engineering (ICE). With its model-centered core, the ICE is a collection of software tools, computational hardware, and methodologies that allow one to move from abstract thought experiments to “operational” testing and optimization of an HCS.

Consistent with solid system engineering practices, an application of the ICE starts with an assessment of the known information about the SoS in question; this typically includes subject matter expert (SME) understanding of the components of the SoS and how they are interconnected, formal architecture and design products, measurement of performance (MOP) level data on the performance of the individual components (usually in isolation), and so on. Where available, we integrate or develop detailed physical models and vetted behavioral descriptions.

Once enough information is amassed about the SoS in question one moves to the prototyping stage. The prototyping environment is driven largely by the SoS and the questions to be asked about it. We make use of two primary simulation frameworks, NetLogo [35] and Repast [36]. When the model does not need to be overly large in scale (less than 10,000 entities) we have had good results using NetLogo [35]. However, there are times when even the prototype must be very large scale which has dictated the use of Repast [36].

As a matter of course, initial prototyping usually beings with NetLogo regardless of the size. If the scale or high-performance is not a driving concern in our analysis, we may not move away from our prototype However, if these concerns are resident we, typically, port the NetLogo simulation to Repast for deployment on a high-performance computing cluster. Both Repast and NetLogo can be run on a cluster.

As a general approach we use NetLogo and/or Repast to handle the representation of the SoS as a whole. Usually, however, there are components of the SoS that are of particular importance to the functioning and performance of the SoS. These high importance components are modeled separately at as high a resolution as possible (e.g. a high resolution, physics based model of a sensor). The family of simulations is now run together to represent the functioning of the SoS.

Use of a computing cluster allows us to scale very large if necessary (e.g., 10^8 agents) as well as to run many replicates of the simulation to perform the Monte Carlo analysis of the modeled SoS. However, despite the inexpensive computational power, the sheer size of the combinatorial parameter space is virtually infinite which necessitates an intelligent design of experiments [37]. This dictates a requirement to reduce the search space by the implementation of a genetic optimization framework.

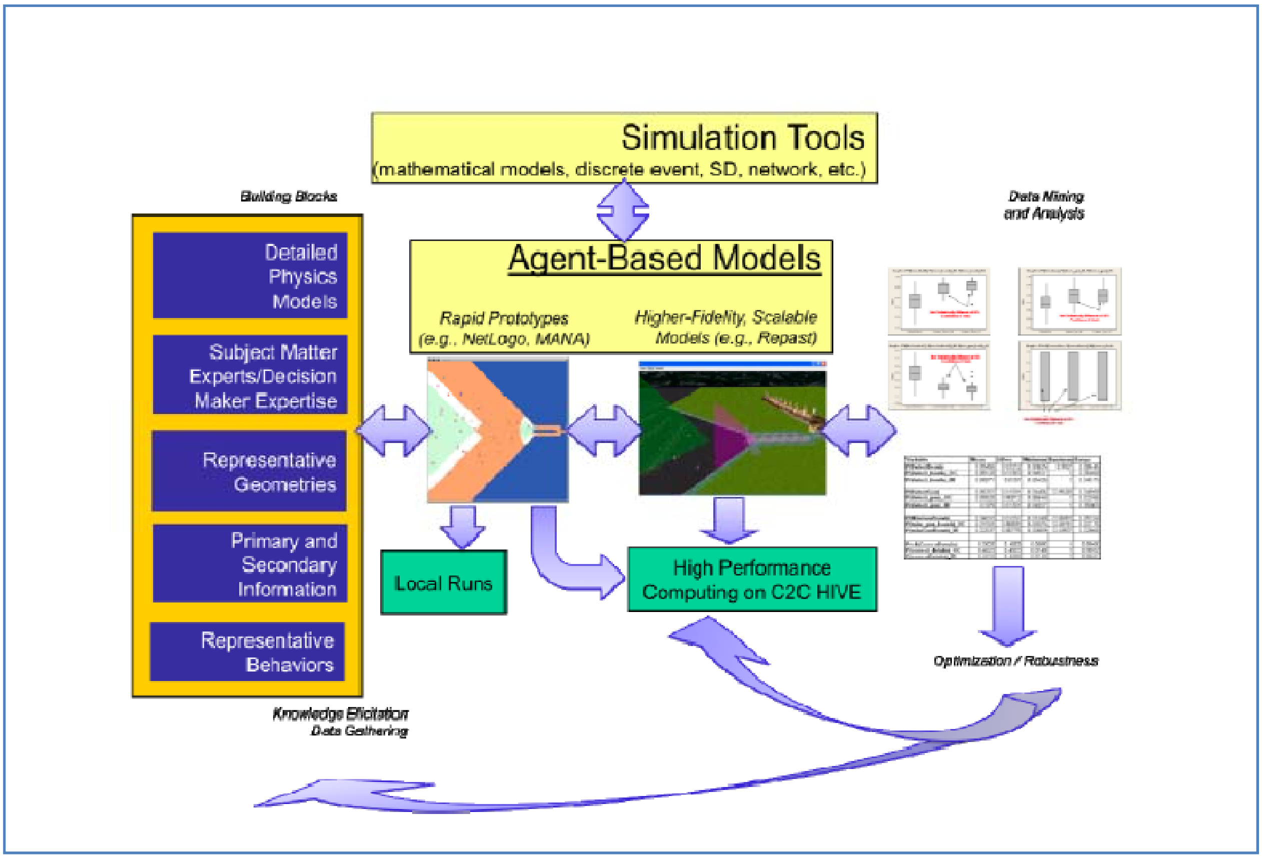

Fig 1. The Infrastructure for Complex-systems Engineering (ICE).

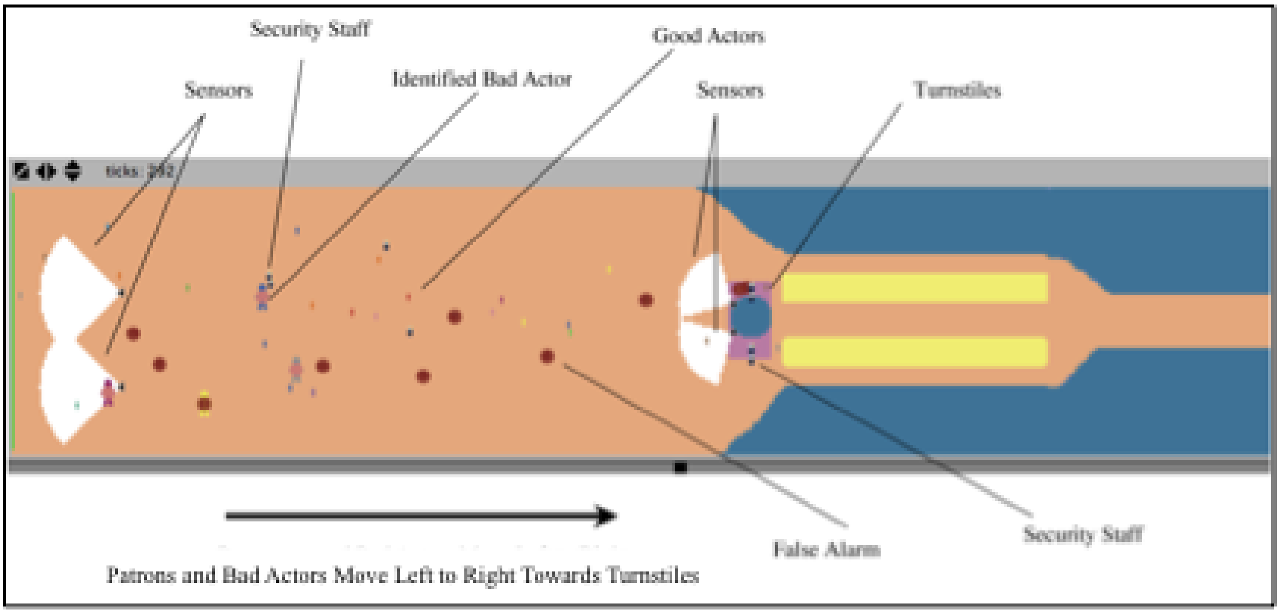

Fig 2. Stadium Simulation Setup.

Concurrent with the model development is the development of a preference or utility model to evaluate the performance of the simulated SoS. Most decisions involve a number of criteria, frequently competing. Our methodology characterizes the goodness of a given decision (e.g., SoS design) by developing a preference structure and an overall utility equation through interaction with SMEs as well as appropriate documentation.

It would be unnecessarily complicated if not impossible to discretely describe each interaction between the components of the SoS. Consequently, to explore this complex interaction we employ an ADS framework. To provide an iterative approach to progressively gain additional understanding of the problem, we rapidly prototype initial simulations to facilitate communication with the subject matter experts and gain additional insight into the problem. Subsequent to the initial runs, we develop a higher fidelity model that is more scalable and designed to make use of a high-performance computing environment. We additionally employ supplementary tools such as Matlab to develop high-resolution models as appropriate.

As a best practice we have found that parallel model development (prototypes that are ported to scalable frameworks while spiraling in new components and features) is important in systems where emergence is a feature of study. This is the case because once the scalable model is developed we can compare its dynamics with those of the prototype and determine their similarities [23]. If both models demonstrate the same dynamics then we can be more confident that these results are a true attribute of the system rather than a bug in our code or an artifact of the modeling framework we chose. The overall methodology is illustrated in Fig. 1.

The subsequent sections describe two case studies that show the applicability of this approach with two very different phenomena. The first explores the use of ADS for designing effective stadium defense strategies. The second shows the utility of using ADS to understand and experiment with continuous double auction financial markets. These two case studies highlight the extreme flexibility of ADS as a systems engineering tool. While stadium defense is very concrete, very spatial, financial market modeling is very abstract. In stadium defense there is a stadium, police officers, and terrorists. In a financial market there are buyers and sellers meeting in a virtual marketplace. As discussed in the next two sections, in both cases, ADS tools can provide insight into problems that would be extremely difficult to gain with other tools.

5. Case Study 1: Defending the Stadium

The objective of this case study was to find the optimal systemic defense for a stadium against an adversary intent on smuggling explosives through a security cordon. The foundation of the defense is dependent upon the optimal placement of a heterogeneous mix of sensors to conduct standoff interrogations of likely bomb carriers. Security forces then employ tactics, techniques and procedures to interpret and fuse the data from the sensors as well as a flexible decision structure to determine when to interdict suspects. The scenario is highly dynamic and resource constrained; interdiction decisions use precious resources and also inform potential adversaries as to the tactics of the defenders. Further, there is an additional cost for incorrect decisions as the overall objective for most venues is to quickly get the patrons into the facility. Any additional delay may have significant revenue implications.

The ICE methodology was used to develop the systems engineering models to provide insight for the decision makers. As shown in Fig. 2, three types of agents were created for the model: Security, Civilians, and "bad actors". The number of agents created each time-step is drawn from a random-exponential distribution. At runtime, a small percentage of the patrons (0.005%) may be further instantiated as an individual with a bomb; thus, becoming a bad actor. Bad actors have a goal to get through the turnstiles undetected while carrying explosives. Bad actors, as well as the rest of the agents, must traverse a corridor from left to right where they are likely to encounter sensors as well as security guards. If a bad actor successfully crosses the turnstile, that agent is considered successful.

The primary goal of security agents is to interdict the bad actors while not engaging "good actors", patrons that are not carrying an explosive. Security agents patrol the corridors as well as around the turnstiles. They base their decision to interdict on fused data from the sensors; when the evidence from the sensors exceeds a threshold, the decision is taken to interrogate the agent under suspicion. As false alarms are possible, non-bad actors may be targeted as bad actors and stopped by the security agents. Bad actors explicitly take evasive measures to avoid security guards. If the security guards close off potential escape routes and the bad actor considers the situation hopeless, the bad actor will exhibit satisficing behavior and detonate the explosives. For this case study the bad actors are essentially rogue individuals that do not act collaboratively.

5.1 Experimental Results

The simulation was run for a wide variety of sensor placements, varying the x and y coordinates of two passive infrared (IR) sensors and a passive millimeter wave (MMW) sensor as well as their respective angle with respect to the incoming traffic. Additionally, the amount of evidence necessary for the security guards to interdict a possible suspect was varied, resulting in a variation of the likelihood of false positives or false negatives. The "goodness" of a configuration of sensors in combination with the likelihood of interdiction was modeled as a utility function. Two primary measures of effectiveness were considered; for a given configuration the probability that an explosive would be detected and the probability that there would be a false alarm, meaning an explosive was indicated where no explosive existed. Two thousand possible combinations were modeled; each design point was run thirty times as there were a number of stochastic features in the simulation.

Although wide variability in the data was observed, the data indicated a number of interesting aspects to the performance of this SoS. From sensor configurations 0 – 26, the general trend is that the MMW sensor starts out significantly in front of the IR sensors and then progressively moves closer. At sensor configuration 27-53, the MMW sensor is behind the IR sensors. As a general practice, it would appear that placing the IR sensors behind the MMW sensor is preferred. Fig. 3 (top) plots utility against numbered sensor configurations.

Fig. 3 (bottom) illustrates the maximum utility plotted against sensor configuration. Consistent with the indication shown in Fig 3, the best performing configurations from a maximum utility perspective are where the MMW sensor is behind the IR sensors. There are two groupings of similar configurations that performed roughly the same; however, the highest utility occurred during a run using sensor configuration 10, where the MMW sensor is in front of the IR sensors. This can be viewed as an anomalous result as the average utility of this configuration was low.

For both the average utility as well as the maximum utility, optimal temperament configuration for the security forces was a mildly aggressive profile. Overly aggressive security forces result in too many false alarms whereas passive security forces miss too many threats. This result seemed relatively consistent across all of the sensor configurations.

6. Case Study 2: The Engineering of Financial Markets

The previous case study provided a discrete problem whose goal was to find an optimal design in a stochastic, multivariate environment. This section describes a forensic study to use ADS to provide insight into extant, complex phenomena. Whereas case study one described a situation where we have the ability to significantly change the design of the SoS, frequently we have the situation where we are presented with a complex SoS where developing insight into its behavioral rules is mission critical yet highly elusive due to the complexity and sheer size of the SoS. Case study two discusses work we have undertaken to use ADS for providing insight into financial markets. Whereas the goals and in fact many of the parameters of the previous case study were concrete in that they represent objects in the physical world, the work described in the next section is much more abstract.

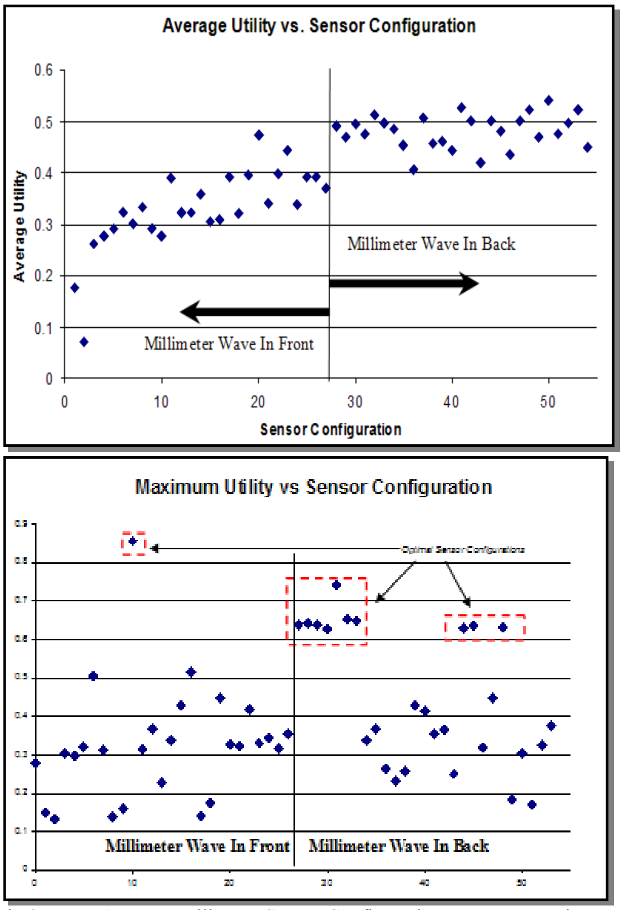

Fig 3. Top: Average Utility vs. Sensor Configuration, Bottom: Maximum Utility vs. Sensor Configuration.

6.1 Introduction

This case study describes a comprehensive study to replicate two ADS of financial markets, that of Farmer and colleagues [38, 39] and that of Cont and colleagues [40]. This case study will introduce another ADS-related concept, namely, docking. The docking framework of Axtell and colleagues [41] is used herein to compare the replicated models to their original counterparts (i.e., referents) and demonstrate the relational equivalence between the referents and replicated models. This is a methodology important to ADS as it can form a logical base for verification, validation, and accreditation of the ADS used for systems engineering and SoS design decision-making. This case study also highlights the importance of only making the SoS simulation as complex as necessary. As will be demonstrated, very simple models can capture much of the dynamics exhibited by a system as complex as a financial market. As the topic of financial market modeling may be new to some readers this case study will begin with an overview of related agent-based market models, then it will describe the replication of the Farmer and Cont models. The case study concludes with a discussion of an extension to the Cont model and a hybrid model combining the market structure of Farmer's model with the trading agents of Cont's model.

6.2 Overview of Agent-Based Models of Stock Markets

Agent-based models of equity stock markets began in the 1992-1993 timeframe with the Santa Fe Institute (SFI) market model [42]. Largely predicated on Holland's [43] genetic algorithms, the SFI market model was well received as a novel contribution, largely based on its qualitative agreement with empirical observations of market dynamics. Following the SFI market model, Lux and colleagues [44, 45] introduced a model with a single trade type (i.e., market orders) that was the first to demonstrate clustered volatility, one of the stylized facts common to many markets. A shortcoming of the Lux model is that its results proved to be sensitive to the size of the trader population. Similar to the Lux market with only market orders (defined, *infra*), LeBaron [46] introduced a market model with agents that learn based on a neural network. An agent can invest a fraction of its total wealth; therefore successful agents can have a large impact in the market. Darley and colleagues [47] built a model of the NASDAQ market and were the first to infuse limit orders (defined, *infra*) into their model and they made five of six correct predictions about the NASDAQ transition to a decimal-based, tick size. Farmer and colleagues [38] built an empirically driven model of zero-intelligence traders intended to depict the structure of a continuous double auction. Cont and colleagues [40, 48, 49] have developed a decentralized trader model that qualitatively represents the five prevailing stylized facts common to most modern markets and consistent over wide time periods. The traders in the Cont models have heterogeneous trading thresholds, and the traders—many of whom trade rather infrequently—adapt their thresholds based upon performance feedback.

6.3 Market Models

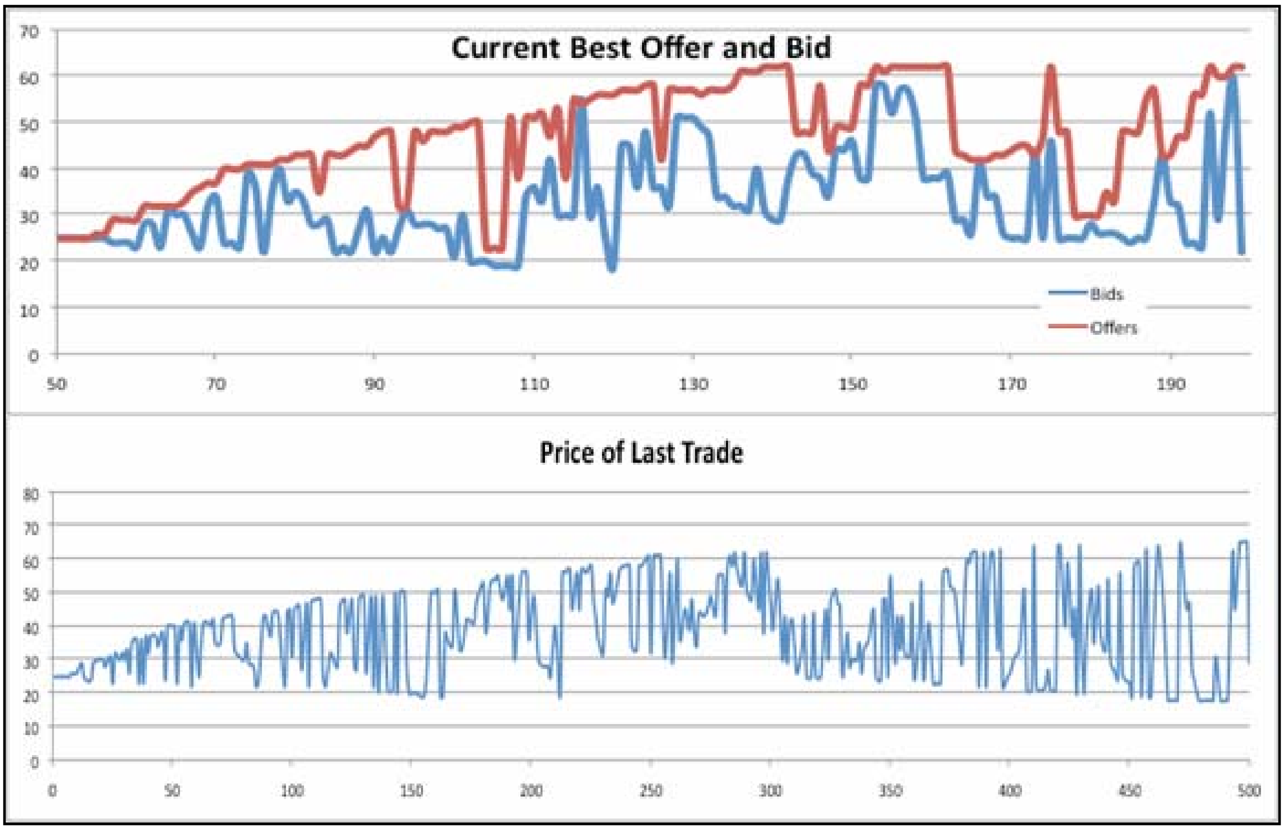

Fig 4. Spread Magnitude Dynamics and Price of Last Trade.

When modeling an artificial financial market the potential complexities are daunting. Further, as lamented in Ghoulmie et al [40] the addition of many of these features makes determining each feature's individual impact very difficult. This brings up a common problem with ADS for SoS design; namely, the simulation can become so complex that it is nearly as difficult to understand as the SoS it is designed to emulate. Therefore, some researchers have taken a different tact, specifically, how simple can a financial model be and still produce results that map to the real world in a non-trivial way? This case study will focus on two of these models: the "Zero Intelligence model" as reported in Iori, et al. [39] and Farmer, et al. [38], and the "Cont model" as reported in Ghoulmie, et al. [40]. We found these two models of particular interest as they approach the problem from very different perspectives. The Farmer model stresses the importance of market structure; whereas the Cont model abstracts the market away almost entirely in favor of traders that have greater than zero intelligence.

6.3.1 The Zero Intelligence Model

The Zero Intelligence Model (ZIM) is, essentially, an analytic (meaning data driven) model. It is a model of a continuous double auction market; therefore, traders can both buy and sell and can do so at any time. There are two types of orders in the ZIM: market orders and limit orders. Market orders are orders that enter the market with an intent to buy or sell a certain number of shares and *do not* specify a particular price. Limit orders, on the other hand, enter the market with both a specified quantity of shares and a specified price. As market orders do not specify a price they are executed immediately upon entering the market at the best available price (see *infra*). Limit orders, however, will accumulate in the market until their specified price is met or they are cancelled. The accumulation takes place in a prioritized queue by price and arrival time. This accumulation of limit orders is called the order book. It is this accumulation of limit orders that creates liquidity (i.e., the ability for market orders to be executed) in the market. In the ZIM both market and limit orders arrive and are cancelled as a Poisson distributed process.

Even with these simple dynamics the behavior of the market model is reasonably similar to those of a real market. An important feature of these markets is the "spread", the distance in terms of price between the best offer to buy shares (i.e., the bid) of an asset and offer to sell an asset (i.e., the ask). The ask and the bid will change through time affecting the spread and the best available prices for market orders. This drift is caused by changes in the proportion of market orders to limit orders. If more market orders arrive than limit orders, the spread will increase as will volatility in the price. This will occur because an increase in market orders will begin to deplete the limit orders in the market. The price impact of a trade is another important feature of these systems. This simple model also appropriately predicts this feature, specifically, that the price impact function is concave [38]. This is caused by the increase in density of limit orders as market orders are executed against limit orders farther down the prioritized (i.e., first in, first out) queue.

6.3.2 NetLogo and RepastS Implementations of the Zero Intelligence Model

We instantiated the ZIM in NetLogo 4.0.2 and RepastS 1.0 so as to mitigate any potential nuances introduced by a particular modeling toolkit. The orders arrive at a Poisson generated rate. All orders are for one unit of the asset. Prices for orders are generated in one of two ways: 1) each order draws a unique price, or 2) a single price is drawn and then given to all arriving orders. The distribution for order price is a function of the current spread and consistent with the ZIM. While, technically the distribution for bid orders can range from -∞ to the best offer and the distribution for offers ranges from the best bid to ∞., within the simulations the distributions are bounded by the "market space" rather than infinity. As market orders arrive they are immediately executed against the best available limit order. At each time step existent limit orders have a random Poisson probability of being cancelled or expiring.

The simulations create dynamics that are consistent with the results reported by Iori et al [39] and Farmer et al. [38]. Both models create order book depth profiles that are sigmoidal in shape.

Furthermore, our simulations of the ZIM also demonstrate a random walk with respect to last trade price and the magnitude of the spread (i.e., distance between best Offer (upper line) and Bid (lower line), as shown in Figs 4 and 5 (Fig. 5 compares NetLogo and Repast runs)[1]. The impact of a trade on price is also consistent between the simulations and the reported results (bottom graph in Fig. 4). Most trades have little impact. But the distribution is heavy-tailed, so while few trades have a large impact, the number of these high-impact trades cannot be ignored completely.

6.3.3 The Heterogeneous Feedback Model

The Farmer model concentrates on demonstrating market order-book structure and price behavior with respect to randomly placed (zero intelligence) trades, whereas the Cont Model [40] introduces the notions of heterogeneity and price feedback: Traders in the market are a heterogeneous group of agents with behaviors that are constantly modified in a feedback process with the market.

There are 5 behaviors exhibited by a wide range of markets and time periods:

1. Excess volatility
2. Heavy tails
3. Absence of autocorrelations in returns
4. Volatility clustering
5. Volume/volatility correlation

While, there are models that demonstrate these statistical properties, most are very complex making it difficult to determine where the statistical properties originate, leading to a diminished explanatory power of the model. This presents a particular problem for evaluation of the SoS in question. If one cannot understand how dynamics arise what hope is there in determining how to design particular dynamics into the SoS as well as engineering the SoS for generating desired behaviors? The ZIM and Cont Model are very good examples of models being driven by a need for simplicity so the causes of dynamics and SoS properties are *highlighted* rather than obscured.

The Cont Model is a heterogeneous feedback model that describes a market where a single asset is traded by a set of agents. At each time step of the model, the agents all receive a common normally distributed "news" signal. Each agent compares that signal to its own unique threshold. If the signal exceeds the agent's threshold, an order is generated; otherwise the agent does not trade during that time period. The excess demand generated by all the agents' market orders causes the price to move, according to a linear price impact function.

The model is parsimonious, comprising of only 4 parameters: the frequency of updating agent thresholds, the standard deviation of the normally distributed "news" process, the market depth (affecting the slope of the price impact function), and the number of agents. Despite this simplicity the model has been shown to produce time series that capture the stylized statistical facts observed in asset returns.

6.3.4 Implementation of the Heterogeneous Feedback Model

As with the Zero-Intelligence Model, our initial goal was to develop models in both NetLogo and RepastS, and then verify results consistent with those reported by Cont. This model was refined and implemented in RepastS to increase scalability and allow for the use of the ICE framework discussed *supra*.

6.3.5 Docking and Relational Equivalence of the Models

[1] Fig. 5 also demonstrates good concordance between the Netlogo and Repast models.

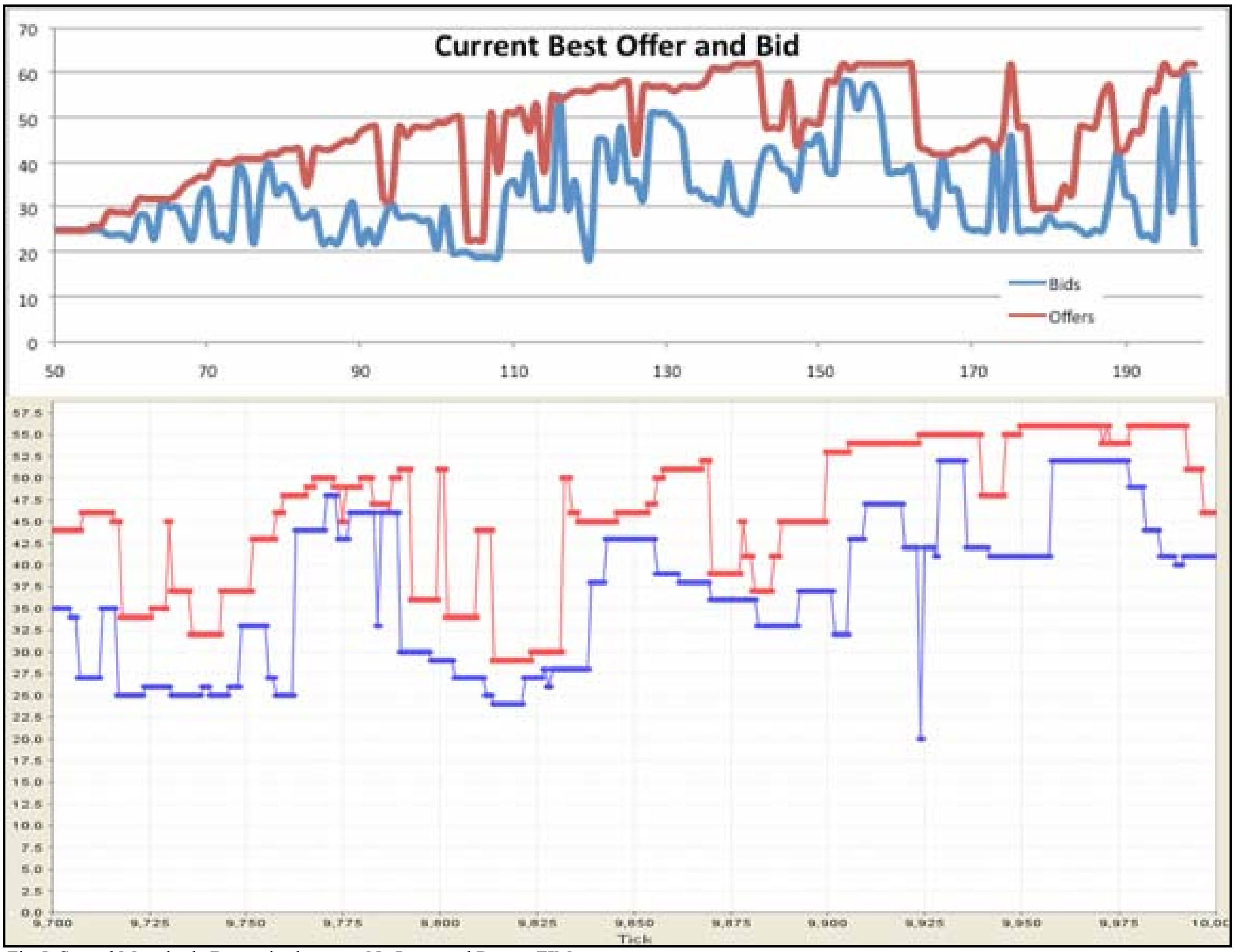

Fig 5. Spread Magnitude Dynamics between NetLogo and Repast ZIMs.

The next step for the simulation effort was to formally "dock" [41] the instantiations of the Farmer and Cont models together and relate them to the published results. For the purposes of this docking exercise, when possible, we will utilize a set of the "Cont statistics" as a common frame for comparison. Of note: the two models discussed herein (ZIM and Cont) compare their output to aggregated measures of system behavior (e.g., asset price, clustered volatility, and particular structures to the autocorrelation of returns); thus, placing them in Level 2 of Axtell's Empirical Relevance (i.e., macro-level quantitative correspondence with the real-world phenomena of interest) [50]. Due to the fact that agent-level correspondence is not at issue here, it is our contention that distributional equivalence (or a lack of statistical dissimilarity between distributions of results) is adequate for this docking exercise.

Unfortunately, we do not possess the original Cont or Farmer models or datasets from them; therefore, all that can be claimed is relational equivalence to the published results. Though not displayed here due to space limitations, the RepastS instantiation replicates the Cont results quite well but the results were less convincing for the NetLogo results. A likely explanation is that this seems to be a function of agent activation regimes (i.e. when agents trade and update).

Our implementation of the ZIM is generally consistent with those described in the Farmer works, of particular note is the random walk of the price, time series of best offer and bid, and sigmoidal shape of the order book. One may also produce Cont statistics for the Farmer model. When one does that it can be shown the Cont statistics for our RepastS and NetLogo instantiations are likely distributionally equivalent.

6.4 Extensions to the Market Models

6.4.1 Cont Market Model Extension

A proof-of-concept extension was made of the Cont model to demonstrate that differentiating the information received by the trading agents would affect the returns received by those groups of agents. In the original Cont model agents received a random signal drawn from a normal distribution with a mean of zero. If the absolute value of this signal was greater than the agent's threshold the agent would sell if the signal was negative and buy if the signal was positive. Within the extension of the Cont model agents were divided into two groups: a large group that continued to receive the random signal and a small group that received information on the actual return being received lagged one time period. Fig. 6 shows the returns received by these two groups of agents (the red line is the small group of agents receiving the signal based upon actual returns). As can be seen in Fig. 6, the returns associated with the non-random signal deviate significantly from returns associated with the agents receiving the random signal. Fig. 7 shows the population returns and a 95% confidence interval around it. It should be noted that a heterogeneous Cont population produces slightly higher variance. It should be noted that Fig. 7 also shows a common feature of ADS; namely, there is "burn in" at the beginning of this SoS simulation. This means that the agents require a period of time after the simulation is started to "settle" down to the long run behavior. It will depend upon the application as the whether or not this is important for the analysis or if it can be discarded.

6.4.2 Combining Cont Traders with a Farmer market structure

An extended model was implemented, combining Cont heterogeneous trading agents in a Farmer order-book market. Inserting heterogeneous trading agents into the order-book based market model required three extensions:

- Random trades (a la Farmer) were eliminated by inserting trading agents from the Cont trading model.
- Trades generated by the Cont agents were executed using limit orders from the order book. The order book was otherwise populated with random limit order arrivals, as in the original Farmer model.
- Actual price returns were used in place of the price impact function since actual transaction information is available from the order book.

The numerical results for the extended model are very similar to the original zero-intelligence Farmer model. This dynamic lends support to the idea that the structure of the market has a great deal of impact on the functioning of the market.

7. Conclusion

As our development of systems becomes increasingly complex, the practice of systems engineering must evolve. Our tools and techniques must expand to handle issues of combinatorial complexity, long-term system evolution and interdependence, managerial independence and emergent behavior. Adequately taking these features into account during the systems engineering of SoS requires concordant maturation of our toolset. This phenomenon is a natural effect of engineering HCS and provides interesting opportunities for engineering and research.

This paper has shown the application of Agent Directed Simulation for Systems Engineering in two very different case studies. The first indicated that this approach has significant potential for providing valuable insight into the design and implementation of Human Complex Systems. The second showed how Agent Direct Simulation can be used to develop deep understanding of instantiated complex enterprises, showing the way for helping us to interact in beneficial ways with these enterprises. Specifically, we have found that these computational capabilities allow us to glean insights into complex SoS, allowing us to deliver systems engineering direction to all phases of the system's lifecycle.

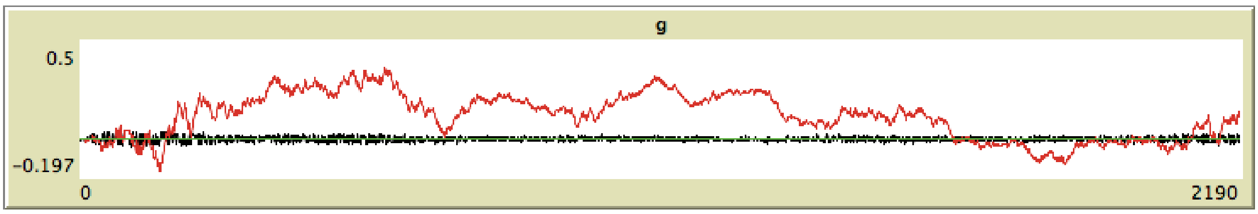

Fig 6. Differentiated Returns for a Small Group of Agents Receiving Additional Information.

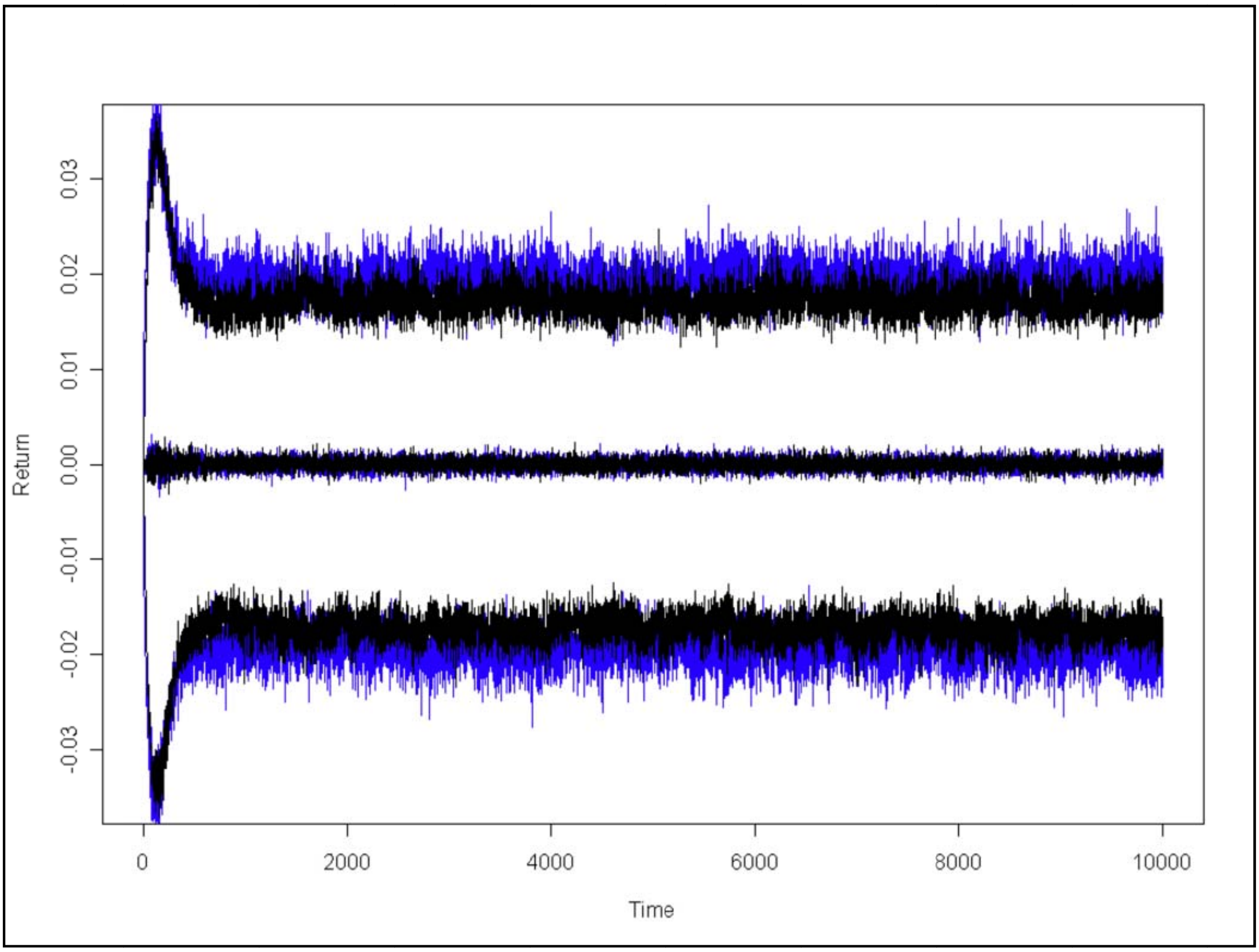

Fig 7. The 95% Confidence Interval for Standard Cont and Extended Cont.

REFERENCES

[1] M. W. Maier, "Architecting Principles for Systems-of-Systems," *Systems Engineering*, vol. 1, pp. 267-284, 1998.[2] P. G. Carlock and R. C. Fenton, "Systems of Systems (SoS) Enterprise Systems Engineering for Information Intensive Organizations," *Systems Engineering*, vol. 4, pp. 242-251, 2001.

[3] A. P. Sage and C. D. Cuppan, "On the Systems Engineering and Management of Systems of Systems and Federations of Systems," *Information, Knowledge, and Systems Management*, vol. 2, pp. 325-345, 2001.

[4] A. J. Kriegel, *Behind the Wizard's Curtain: An Integration Environment for a System of Systems*. Washington, DC: CCRP, 1999.

[5] P. Wegner, "Why Interaction is More Powerful than Algorithms," *Communications of the ACM*, vol. 40, pp. 80-91, 1997.

[6] P. Wegner and D. Goldin, "Computation Beyond Turing Machines: Seeking Appropriate Methods to Model Comutinging and Human Thought," *Communications of the ACM*, vol. 46, pp. 100-102, 2003.

[7] H. A. Simon, *The Sciences of the Artificial*, 3rd ed. Cambridge, MA: MIT Press, 1996.

[8] M. E. Csete and J. C. Doyle, "Reverse Engineering of Biological Complexity," *Science*, vol. 295, pp. 1664-1669, 2002.

[9] T. C. Schelling, *Micromotives and Macrobebahavior*. New York: Norton, 1978.

[10] S. R. Buss, C. H. Papadimitriou, and J. N. Tsitsiklis, "On the Predictability of Coupled Automata: An Allegory about Chaos," *Complex Systems*, vol. 5, pp. 525-539, 1991.

[11] J. M. Epstein and R. A. Hammond, "Non-Explanatory Equilibria: An Extremely Simple Game with (Mostly) Unattainable Fixed Points," *Complexity*, vol. 7, pp. 18-22, 2002.

[12] J. M. Epstein and R. L. Axtell, *Growing Artificial Societies: Social Science from the Bottom Up*. Washington: Brookings Institution Press, 1996.

[13] J. M. Epstein, "Agent-Based Computational Models and Generative Social

[14] K. M. Carley and D. M. Svoboda, "Modeling Organizational Adaptation as a Simulated Annealing Process," *Sociological Methods & Research*, vol. 25, pp. 138-168, 1996.

[15] D. A. Levinthal, "Adaptation on Rugged Landscapes," *Management Science*, vol. 43, pp. 934-950, 1997.

[16] J. G. March, "Exploration and Exploitation in Organizational Learning," *Organization Science*, vol. 2, pp. 71-87, 1991.

[17] B. McKelvey, "Avoiding Complexity Catastrophe in Coevolutionary Pockets: Strategies for Rugged Landscapes," *Organization Science*, vol. 10, pp. 294-321, 1999.

[18] B. McKelvey, "Self-Organization, Complexity Catastrophe, and Microstate Models at the Edge of Chaos," in *Variations in Organization Science: In Honor of Donald T. Campbell*, J. A. C. Baum and B. McKelvey, Eds. Thousand Oaks, CA: Sage, 1999, pp. 279-310.

[19] B. F. Tivnan, "Coevolutionary Dynamics and Agent-Based Models in Organization Science," in *Proceedings of the 2005 Winter Simulation Conference*, M. E. Kuhl, N. M. Steiger, F. B. Armstrong, and J. A. Joines, Eds. Piscataway, NJ: Institute for Electrical and Electronics Engineers, 2005, pp. 1013-1021.

[20] B. McKelvey, "Complexity Theory in Organization Science: Seizing the Promise or Becoming a Fad?," *Emergence*, vol. 1, pp. 5-32, 1999.

[21] R. M. Cyert and J. G. March, *A Behavioral Theory of the Firm*. Englewood Cliffs, NJ: Prentice-Hall, 1963.

[22] K. J. Dooley, "Simulation Research Methods," in *Companion to Organizations*, J. A. C. Baum, Ed. Oxford, UK: Blackwell, 2002.

[23] K. M. Carley, "Computational and Mathematical Organization Theory: Perspective and Directions," *Computational and Mathematical Organization Theory*, vol. 1, pp. 39-56, 1995.

[24] M. R. Lissack, "Editor's Note," *Emergence*, vol. 1, pp. 3-4, 1999.

[25] N. Gilbert, "Editorial," *Journal of Artificial Societies and Social Simulation*, vol. 1, 1998.

[26] K. J. Dooley and A. H. Van de Ven, "Explaining Complex Organizational Dynamics," *Organization Science*, vol. 10, pp. 358-375, 1999.

[27] N. Gilbert and K. G. Troitzsch, *Simulation for the Social Scientist*. Philadelphia, PA: Open University Press, 1999.

[28] K. E. Weick, "What Theory is *Not*, Theorizing *Is*," *Administrative Science Quarterly*, vol. 40, pp. 385-390, 1995.

[29] R. Axelrod, "Advancing the Art of Simulation in the Social Sciences," in *Simulating Social Phenomena*, R. Conte, R. Hegselmann, and P. Terna, Eds. Berlin: Springer, 1997, pp. 21-40.

[30] H. A. Simon, *Administrative Behavior*, Third ed. New York: Free Press, 1976.

[31] B. McKelvey, "Quasi-Natural Organization Science," *Organization Science*, vol. 8, pp. 352-380, 1997.

[32] P. Anderson, "Complexity Theory and Organization Science," *Organization Science*, vol. 10, pp. 216-232, 1999.

[33] L. G. Zucker, *Institutional Patterns and Organizations: Culture and Environment*. Cambridge, MA: Ballinger, 1988.

[34] W. R. Scott, *Institutions and Organizations*. Thousand Oaks, CA: Sage, 1995.

[35] U. Wilensky, "NetLogo." Evanston, IL.. Center for Connected Learning and Computer-Based Modeling, Northwestern University, 1999.

[36] E. Tatara, M. J. North, T. R. Howe, N. T. Collier, and J. R. Vos, "An Introduction to Repast Modeling by Using a Simple Predator-Prey Example," presented at Agent 2006 Conference on Social Agents: Results and Prospects, Chicago, IL, 2006.

[37] J. P. C. Kleijnen, S. M. Sanchez, T. W. Lucas, and T. M. Cioppa, "State - of - the - Art Review: A User's Guide to the Brave New World of Designing Simulation Experiments," *INFORMS Journal on Computing*, vol. 17, pp. 263-289, 2005.[38] J. D. Farmer, P. Patelli, and I. I. Zovko, "The Predictive Power of Zero Intelligence in Financial Markets," *Proceedings of the National Academy of Science*, vol. 102, pp. 2254-2259, 2005.

[39] G. Iori, M. G. Daniels, J. D. Farmer, L. Gillemot, S. Krishnamurthy, and E. Smith, "An Analysis of Price Impact Function in Order-Driven Markets," *Physica A*, vol. 324, pp. 146-151, 2003.

[40] F. Ghoulmie and R. N. Cont, Jean-Pierre, "Heterogeneity and Feedback in an Agent-based Market Model," *Journal of Physics: Condensed Matter*, vol. 17, pp. S1259-S1268, 2005.

[41] R. L. Axtell, R. Axelrod, J. M. Epstein, and M. D. Cohen, "Aligning Simulation Models: A Case Study and Results," *Computational and Mathematical Organization Theory*, vol. 1, pp. 123-141, 1996.

[42] W. B. Arthur, J. H. Holland, B. LeBaron, R. Palmer, and P. Tayler, "Asset Pricing Under Endogenous Expectations in an Artificial Stock Market," Santa Fe Institute, Sante Fe, NM 96-12-093, 1996.

[43] J. Holland, "Genetic Algorithms," *Scientific American*, vol. 267, pp. 66-72, 1992.

[44] T. Lux, "The Socioeconomic Dynamics of Speculative Markets: Interacting Agents, Chaos and the Fat Tails of Return Distributions," *Journal of Economic Behavior and Organization 33: 143-165*, vol. 33, pp. 143-165, 1998.

[45] T. Lux and M. Marchesi, "Scaling and Criticality in a Stochastic Multi-Agent Model of a Financial Market," *Nature*, vol. 397, pp. 498-500, 1999.

[46] B. LeBaron, "Evolution of Time Horizons in an Agent-Based Stock Market," *Macroeconomic Dynamics*, vol. 5, pp. 225-254, 2001.

[47] V. Darley, A. Outkin, T. Plate, and F. Gao, "Learning, evolution and tick size effects in a simulation of the NASDAQ stock market," presented at Proceedings of the 5th World Multi-Conference on Systemics, Cybernetics and Informatics, Orlando, FL, 2001.

[48] R. Cont, "Volatility Clustering in Financial Markets: Empirical Facts and Agent-Based Models," in *Long Memory in Economics*, G. Teyssiere and A. Kirman, Eds.: Springer, 2005.

[49] R. Cont and J.-P. Bouchaud, "Herd Behavior and Aggregate Fluctuations in Financial Markets," *Macroeconomic Dynamics*, vol. 4, pp. 170-196, 2000.

[50] R. L. Axtell, "Three Distinct Kinds of Empirically-Relevant Agent-Based Models," Center on Social and Economic Dynamics, Brookings Institution, 2005, pp. 24.

Philip Barry received his B.S. in Aerospace Engineering from the University of Virginia, his M.S. in Systems Engineering and his Ph.D. in Information Technology from the George Mason University. He is currently a Department Head at MITRE Corporation where he leads groups in language technology, information semantics, the knowledge discovery and human centered design. Phil has applied systems agent-based systems engineering to problems in the department of defense as well as the department of homeland security. He has published a number of papers in various venues in the areas of agent based modeling, modeling and simulation and systems engineering. He also teaches courses at George Mason University as an adjunct faculty in the areas of Enterprise Systems Engineer, Experimental Design, Artificial Intelligence and Project Management.

Matthew Koehler received his A.B. in Anthropology from Kenyon College in 1994, his MPA from the School of Public and Environmental Affairs, Indiana University in 1998, and his JD *cum laude* from the George Washington University School of Law in 2001. He attended the Santa Fe Institute's Complex Systems Summer School in 2000 and is now studying Computational Social Science at George Mason University. He is currently an artificial intelligence engineer with the MITRE Corporation where he also leads the Knowledge Discovery and Decision Analytics group. Matt has applied agent-based modeling to many problems faced by the US Federal Government. He has published many papers dealing with agent-based modeling, analysis, validation, and visualization.

Matthew McMahon received his B.S. in Electrical Engineering from Auburn University and his M.S. in Computer Science from Virginia Polytechnic Institute and State University. Currently, he is studying Computational Social Science at George Mason University. Prior to joining The MITRE Corporation as a Lead Engineer, Matt was the Owner and Chief Architect of MSM Consulting.

Brian Tivnan received his B.S. in Mechanical Engineering from the University of Vermont, his M.S. in Operations Research from the Naval Postgraduate School and his doctorate in

Computational Organizational Theory from the George Washington University. He is currently a Principal Engineer at The MITRE Corporation where he is the chief engineer of the modeling and simulation department. Brian has conducted both basic and applied research in the areas of human complex systems and agent-based modeling; as well as addressed critical engineering problems for the U.S. Departments of Defense and Homeland Security and the Federal Reserve Board. Prior to joining MITRE, Brian spent ten years on active duty as an infantry officer in the U.S. Marine Corps.