Chapter 13
Complex Manufacturing and Service Enterprise Systems: Modeling and Computational Framework

Radu F. Babiceanu

Department of Systems Engineering, University of Arkansas at Little Rock, USA
rfbabiceanu@ualr.edu

Abstract. This work comes as a contribution to the efforts that are undergoing within engineering systems community to account for the increased complexity of today's manufacturing or service systems. These systems are becoming more and more complicated due to the increase in the number of elements, interconnections within the system, and necessary integration with other systems. Moreover, through the emphasis on self-organization and considering the multi-stakeholders context and objectives, these systems are crossing the line towards complexity. There is a need for developing a framework to be used in modeling, analysis, and integration of systems that operate in uncertain environments, in which characteristics such as adaptation, self-organization and evolution, or in other words behavior prediction, need to be addressed. The proposed complex enterprise systems framework combines knowledge coming from complex systems science and systems engineering domains, and uses computational intelligence and agent-based systems simulation methodologies. The approach requires computational experience in manipulating large amounts of data and building large-scale simulation models. A significant result to be made possible by this research is that systems may no longer have a fixed, life-cycle long, design based on identified requirements; systems will be engineered to evolve and adapt as needed during the operational phase, while respecting their operational environment constraints.

Keywords: complex enterprise systems, behavior prediction, agent-based modeling and simulation, holonic enterprise systems.

1 Introduction

On January 23, 2000, the famous British physicist Stephen Hawking stated in an interview for the San Jose Mercury News: "I think the next century will be the century of complexity". Other theoreticians described the 21st century as "the systems century" [1]. Systems engineering (SE) as a discipline is in the position to complement its traditional approach of translating operational requirements into optimized systems configurations with elements and characteristics identified in natural complex systems, such as, adaptability, self-organization, and evolution. By having these capabilities available in their toolset, engineered systems and enterprises will be better

T. Borangiu et al. (Eds.): *Service Orientation in Holonic and Multi Agent*, SCI 472, pp. 197–212.
DOI: 10.1007/978-3-642-35852-4_13

prepared to respond to the increased complexity of today's business and operational environment.

This work comes as a contribution to the efforts that are undergoing within engineering systems community to account for the increased complexity of today's manufacturing or service systems. These systems are becoming more and more complicated due to the increase in the number of elements, interconnections within the system, and necessary integration with other systems. Moreover, through the emphasis on self-organization and considering the multi-stakeholders context and objectives, these systems are crossing the line towards *complexity*. Consequently, there is a need for developing a framework to be used in modeling, analysis, and integration of systems that operate in uncertain environments, in which characteristics such as adaptation, self-organization and evolution, or in other words behavior prediction, need to be addressed.

The proposed complex enterprise systems framework combines knowledge coming from complex systems science and systems engineering domains, and uses computational intelligence and systems simulation methodologies. The approach requires computational experience in manipulating large amounts of data and building large-scale simulation models. A significant result to be made possible by this research is that systems may no longer have a fixed, life-cycle long, design based on identified requirements; systems will be engineered to evolve and adapt as needed during the operational phase, while respecting their operational environment constraints. Finally, being generic, the proposed framework is expected to be applicable to all the types of large-scale manufacturing and service complex systems that are designed and work in uncertain environments.

2 Complex Systems, Engineered Systems, and Complex Enterprise Systems

Natural world was engineers' inspiration for centuries regarding the way to build systems that benefit mankind. With the goal of optimal design, traditional engineering disciplines and systems engineering looked at including characteristics such as predictability, controllability, and reliability into the engineered systems. Natural world, however, exhibits also other characteristics such as adaptability, self-organization, and evolution, which were not included in the design and operational sequences of engineered systems until recently. Today's manufacturing and service systems are more intricate and complicated than ever, and achieving the only goal of optimal design may be too restrictive. Engineered systems should have the mechanisms that allow them to adapt to changes in requirements and deal with environment uncertainties, while in operation. For large complex systems and enterprises, deterministic predictability and controllability is no longer sufficient. These systems should be engineered with the goal of providing a meaningful and real-time response, while retaining reliability as a main characteristic.

2.1 Complex Systems

Having its foundations on systems science, complex systems science "emerged from the interplay of physics, mathematics, biology, economy, engineering, and computer science, [with the] mission to overcome the simplifications and idealizations that lead to unrealistic models in these sciences" [2]. The **complex adaptive systems** (CAS) theory is built upon characteristics of many disciplines such as, evolutionary biology, nonlinear dynamical systems, and artificial intelligence. CAS focuses on the interaction between the system and its environment and the co-evolution of both the system and the environment [3]. The main characteristics of CAS are *self-organization* (defined as, the spontaneous appearance of large-scale spatial, temporal, or spatiotemporal order in a system of locally interacting simple components), *emergence* (defined as, the appearance of a large-scale phenomenon or property that cannot be reduced to a superposition of contributions by individual system elements), and *adaptability* (defined as, the process by which a system modifies its processes and/or structures in response to external or internal feedback in order to improve its fitness).

2.2 Engineered Systems

Blanchard and Fabrycky [4] define an engineered system to be a combination of interrelated elements, parts, methods, or procedures forming a complex unitary whole working together towards a common objective. Traditional SE framework considers a clear distinction between the design and production phases, and the operational phase, and does not allow any type of change in the operational phase; engineered systems need to respect all the hard specifications of the designer. Even systems considered to be adaptive, such as adaptive controllers or neural networks, follow this two-phase approach, allowing changes (i.e., adaptation) only in the superficial sense of parameter adjustment [5]. Every operational behavior at the system level can be traced back in terms of initial requirements. There is a vast literature on modeling and analysis of large-scale systems that presents the design methodologies and the tools used in analysis of these systems. However, these systems are analyzed in terms of their components, and just a few studies attempt to provide a systems approach to modeling and analysis. Individual systems can be analyzed using operations research methodologies and modeled using computer science algorithms, but these tools work at the individual system or component level. Once the big picture is considered, these methods of analysis fail to provide the needed insight to the problem. And, since the emergent characteristics can only be seen at the system level, the big picture needs to be considered in the analysis process.

2.3 Complex Enterprise Systems

In the understanding of this work, complex engineered systems, or *engineering complex enterprise* (manufacturing or service) *systems* is the process of engineering of manufacturing and service large-scale systems such as distributed manufacturing operations, health care delivery systems, local and national infrastructure, globally distributed supply and demand chains (value chains), etc., which involve a large number of interacting entities, and have several stakeholders with different objectives. A characteristic of these systems is their emergent behavior viewed at the system level, behavior that cannot be traced back to the individual system components. As these systems tend to increase in their scale and complexity, systems engineering, as it is today, cannot provide the means for accurate modeling and analysis processes. The principle that states that "the whole is more than the sum of the parts", principle that can be traced back to Aristotle is as true today as it always was: "The systems problem is essentially the problem of the limitations of analytical procedures in science. This used to be expresses by half-metaphysical statements, such as emergent evolution or 'the whole is more than a sum of its parts', but has a clear operational meaning" [6].

Since traditional systems engineering and its methods and tools coming from operations research, computer science, and decision sciences cannot address the increased complexity of today's engineered and organizational systems, engineering should borrow from complex systems research which offers the possibility to build a framework using already studied concepts such as complexity, fractals, emergence, self-organization, adaptation, evolution, etc. In engineering words, complexity is defined by the National Institute of Standards and Technology as the "intrinsic amount of resources, for instance, memory, time, messages, etc., need to solve a problem or execute an algorithm." This definition does infer that in the context of a very large number of resources needed, the behavior of a system that relies on problem-solving or algorithm-execution may become intractable. It is worth noting that complexity, in the sense of this definition, can, however, decrease with learning, so assessing the "true" complexity of a system is at all times dependent on the observing entity.

2.4 Background on Modeling Complex Systems

Traditionally, in physics, chemistry and other sciences, complex systems are modeled using analytical techniques which include nonlinear dynamics, differential and difference equations, time series analysis, graph and network theory, cellular automata, and Markov processes. The interest of this research is on the resulting complexity characteristics of nonlinear dynamic systems, regardless of the way they are modeled analytically. According to Ottino [7], complex systems show a form of organization without any external organizing principle being applied to them, or in other words complex systems demonstrate self-organization characteristics.

The complexity theory can be traced back to Poincaré in the 1890s, who indicated the possibility for certain systems to be subject to sensitive dependence on initial conditions. More than 50 years later, Lorenz came to the same answer while conducting weather forecasting studies and stated that systems in the real world do not behave in a precisely repeatable way. Therefore, prediction of the long-term future is unlikely for the nonlinear dynamical systems. A small difference in starting conditions alters the behavior of the system [8]. This sensitivity dependence on initial conditions is best illustrated by the Lorenz strange attractor [9]. Until the advent of modern computing with its tremendous increase in computational power only a few other complex behavior of differential equations were published. Only two decades later, modern computing showed that simple systems of dynamical equations having a small number of parameters can produce an unlimited variety of complex behaviors. Since, there are no known rules for predicting the complex behavior, an extensive search is required to obtain behaviors of interest. Boccara [10] describes these dynamical models dependent on a previous state as equations of the form: $x_{t+1} = f(x_t)$, where x, represents the state of the system and belongs to a subset $J \in \mathbb{R}^n$, the function $f: J \to J$ is called a model, and $t \in \mathbb{N}_0$ is the state parameter. A complex model is a function of form *f* defined above, which exhibits some sort of complex behavior. In a complex model a bifurcation occurs when a small change in the model parameters causes a sudden change in the long-term dynamical behavior.

Polack and Stepney [11] noted that, in terms of systems analysis, in emergent systems there is a discontinuity between the global and local system description. Their work explored the development of emergent systems using cellular automata by using simple algorithms that have emergent characteristics. The objective of their research is to systematically determine the system components and integration environment that is capable of forming the required properties as emergent effects. Johnson [12] characterizes the system level emergent properties as both beneficial (users adapt products to support tasks that designers never intended) and harmful (they undermine important safety requirements of the system), so the process of modeling the emergent characteristics needs to augment the beneficial emergent effects, while suppressing the harmful ones.

2.5 Background on Large-Scale Systems Simulation

Even not similar to a regular knowledge discovery process, represented usually by data mining techniques, identifying the nonlinear processes that potentially control complex systems is also done by studying very large amounts of data. Behavior characterization through simulation does not extract directly previously unknown knowledge from large databases, but uses entire databases to build knowledge by discovering the complex model(s) that governs system behavior. The literature review identified, however, a series of research papers related to this current work, that use knowledge discovery processes applied to study complex systems phenomena. An important issue in the knowledge discovery process is a reliable filtering of meaningful patterns from those trivial. The patterns of data extracted should be non-trivial, valid, novel, useful and comprehensible [13]. McGarry [14] reviews the methods of evaluation for the actual worth of the discovered patterns in the data mining process. Last *et al.* [15] discuss the knowledge discovery process in time-series databases and their approach includes cleaning and filtering of time series data, identifying the most important predicting attributes, and extracting a set of association rules that can be used to predict the time series behavior in the future. Their methodology is based on signal processing techniques and information-theoretic fuzzy approach to extract the rules, which are then further reduce using the computational theory of perception. Two types of time series, stock-market data and weather data are used as examples for their approach. Large-scale agent-based and cellular automata simulations are also identified in the literature as means for modeling and characterizing the behavior of physical and natural phenomena and systems [16-17].

3 Framework for Engineering Complex Enterprise Systems

This work proposes a multi-scale, multi-objective modeling framework applicable to complex man-made systems regardless of their nature. Since complexity as a term and complex systems as concept are debatable within science and engineering community, for the purpose of this work, a distinct delimitation is made between systems whose behavior can be completely understood through functional decomposition and systems that exhibit emergent behavior. The former will be named *complicated*, while by *complex* system, this work refers to the latter. This delimitation is in accordance with the recent advancements in engineering systems research exemplified by the published work in the systems engineering area. As a comparison, a modern manufacturing plant, which includes a large number of machines, whose work in process is determined based on reliable forecasting data is a good example of a complicated system conforming to the above definition, while the global value chain for the same manufacturing plant, which includes several other companies whose operations are exhibiting uncertain behavior, is a reasonable example of a complex system.

3.1 Factors Influencing Complexity

According to Bar-Yam [18], a complex system exhibits behaviors not understandable, and which may not be inferred from the structure and behavior of its component parts. These perceived complex behaviors can be attributed to one or more of the following characteristics, “large numbers of elements, large numbers of relationships among elements, nonlinear and discontinuous relationships, and uncertain characteristics of elements and relationships” [19]. Since these systems show evidence of complex structural and operational characteristics that are not accounted for within the traditional systems engineering framework, engineering research needs to propose a new modeling framework that addresses these characteristics. As presented in Fig. 1, complexity in enterprise systems is due to one or more factors: *system modeling* through its architecture and multiple scale and time characteristics; *system interactions* through its internal interconnections and interfaces with other systems; *system multiple objectives* and *multiple stakeholders*, resulting in frequent trade-offs in the analysis process; *system learning*, resulting in adaptation and reconfigurability capabilities; *system context*, dealing with the environment in which the system operates; and, *system information* through the collection and distribution of data [19].

The complex enterprise systems architecture framework needs to address the physical component systems, the social organization of the component systems, as well as behavior characteristics of individual humans and social organizations components. Since the systems are intended to be engineered for evolution and adaptability, the architecture framework should be capable of generating adaptive and evolvable behaviors and demonstrate agility in responses to external and internal stimuli. Moreover, it should include fixed system elements and be capable of accounting for system changes in terms of structure, relationships, controls and incentives. Trade-off study capabilities to address the balance between system efficiency and evolution characteristics should be considered in the framework development, as well. Nevertheless, the overall generic complex enterprise system architecture needs to address the system multiple goals, while respecting individual sub-systems hard constraints.

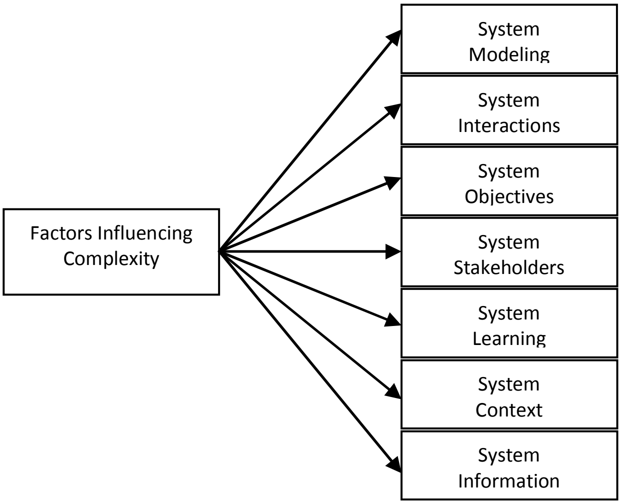

Fig. 1. Factors influencing systems complexity

3.2 Proposed Modeling and Simulation Approach

Experiments, modeling, and simulation are essential in order to analyze a self-organizing system and to engineer systems that exhibit emergent properties. Since a complex enterprise system evolves continuously, it has to be engineered such that its evolution is within a predictable pattern. The proposed process of engineering of complex enterprise systems can be performed either on an existing, or a new system, and it consists of two main components, *descriptive* and *predictive components*, as presented below. For each of the two components, their objectives, methodologies to accomplish them, and the outcome of the processes are considered.

- **Descriptive Component**
  - Purpose: Theoretical Description of the System
  - Methodology: System Modeling
  - Outcome: Complex Model Identification
- **Predictive Component**
  - Purpose: Prediction of System Behavior
  - Methodology: System Simulation
  - Outcome: Complex Behavior Identification

3.3 System Modeling

The system modeling process is performed to obtain a theoretical description of the system in the form of a complex model. Considering the current system stakeholder, states, goals, and factors influencing its complexity previously identified, the process of system modeling is presented in Fig. 2.

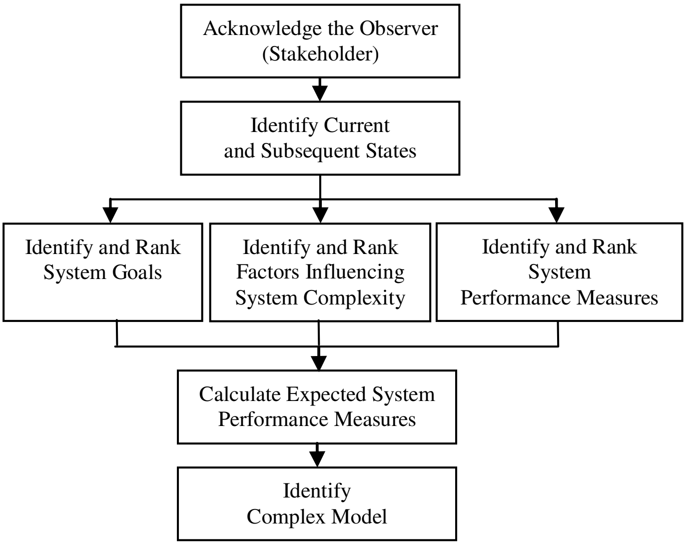

Fig. 2. System modeling for complex model identification

The first steps are to acknowledge the observer by selecting one of the multiple stakeholders and identify the current and subsequent system states based on the selected stakeholder view of the system. Then, system goals are identified and ranked based on hard operational requirements and stakeholder's point of view; out of the remaining five factors previously linked to system complexity, the ones that affect the system are identified and ranked based on factorial experiments; systems engineering process is used to identify and rank system *technical performance measures* (TPM); and, all this information is used to calculate the expected values of the TPMs. In the last step, the transition of the system from one state to another is approximated as close as possible with nonlinear complex mathematical models based on the expected values of the system TPMs. The complex model identification is a computational process that involves searches in the models' space based on the values of TPMs calculated for the current and subsequent states.

The ranking of TPMs is obtained using factorial experiments with three TPMs as factors at three different levels of systems complexity. The interaction between the TPMs and the system complexity levels is also considered in the factorial experiments. The output of the factorial experiments will give the most important TPM based on which the complex model will be identified. Ranking of the factors that influence the complexity of the system can also be obtained as a result of factorial experiments.

3.4 System Simulation

The advances in systems characterization using complexity theory provide opportunities for the computational simulation techniques to attempt at prediction of the future behavior of complex systems. The system simulation component works in two ways. First, by using observed systems data as well as simulation techniques, the predictive component undertakes the complex enterprise systems behavior from both the experimental and computational directions. The purpose of this approach, presented in Fig. 3 below, is to characterize the observed data and identify the underlying complex models that control the system behavior, functions which the traditional analytical approach cannot uncover. These complex models are further used in the prediction process for complex behavior identification. The second approach is to use the complex models identified in the above system modeling component and proceed directly to the second set of simulation experiments.

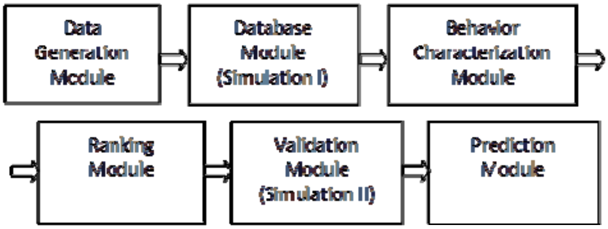

Fig. 3. System modeling for complex model identification

The last process presented in Fig. 3, part of the prediction component, checks for compliance with the SE specification values for the TPMs, and rejects any complex model that leads to a status that does not respect the critical, hard constraints. This process can be done using a general decision evaluation display as presented in Fig. 4 below. The most important TPM identified in the system modeling component of the framework is plotted on the horizontal axis, and all other TPMs are plotted on the vertical axes. Except for the TPMs that are critical and must be respected for a safe operation of the system, all other TPMs plotted vertically may be subject to trade-off processes between them.

For a given system, large sets of data are generated by fitting probability distributions to the data points obtained from the observed system behaviors. These probability distributions are designed such that slightly changes in the initial conditions for the system under study are obtained. These sets of data are used in the first simulation model to build a large database of potential output system behaviors. The behavior characterization module selects through all these potential output system behaviors and removes from the database all unfeasible generated behaviors. Ranking the remaining and potential effective behaviors is a step further in the overall simulation process, and includes grouping the behaviors in sets based on their scores for each of the TPMs considered. Since complex systems have several stakeholders, each of them with its own set of objectives, there will be several performance measures associated with each set of objectives.

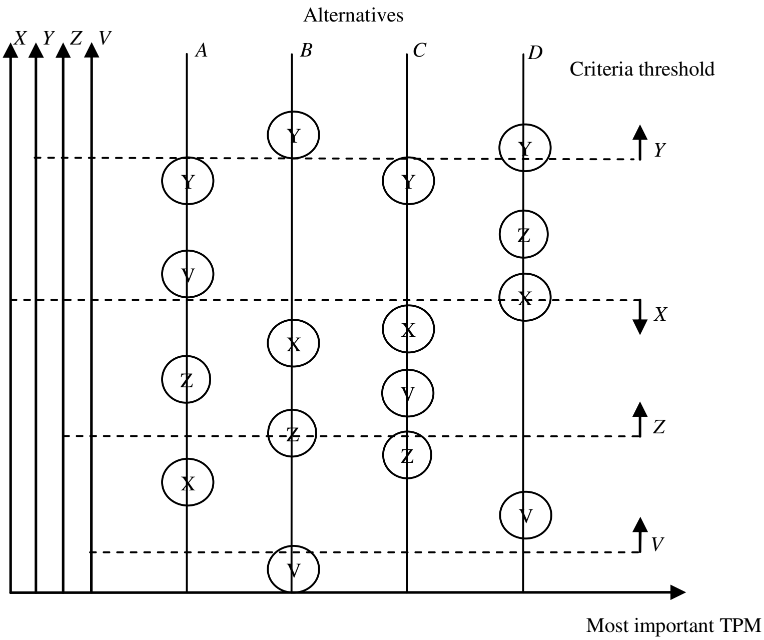

Fig. 4. General decision evaluation display for TPMs

The second simulation model is used to statistically validate the feasible and potential effective behaviors identified through the characterization process or in the system modeling component of the complex enterprise systems framework. The complex models identified through the system modeling process are placed in a temporary database for system validation purposes. The same database is used for the second approach of system simulation component to store the potential valid complex models identified through simulation. Another set of experimental work is necessary to validate the predictions stored in the temporary database. Once the predictions proved to be valid, the complex models are sent to the prediction module and grouped in sets corresponding to each of the system TPMs considered.

3.5 Design of Simulation Experiments

Since the time scale characteristic was identified as an important factor affecting the complexity of systems, the simulation models are developed based on a timed step approach. There is a correspondence between the variables of interests considered in the observational data points and the random variables considered in the simulation models. As stated above, two series of simulations experiments will be performed, one using observed experimental data and the other one using generated data fitted to the observed data by using probability distributions.

The first series of simulations, presented in Fig. 5, will rely on comparison techniques to select among the generated behaviors. This series of simulation experiments will use huge amounts of data generated by fitting input probability distributions to the observed data. For each time period [i, i + 1] between two consecutive time steps [i], [i + 1], the algorithms embedded in the evaluation engine of the simulation model are run independently or in combinations of two or more such that the entire spectrum of the observed behaviors is covered. The behavior characterization process is done offline and uses the database delivered by the evaluation engine of the simulation model.

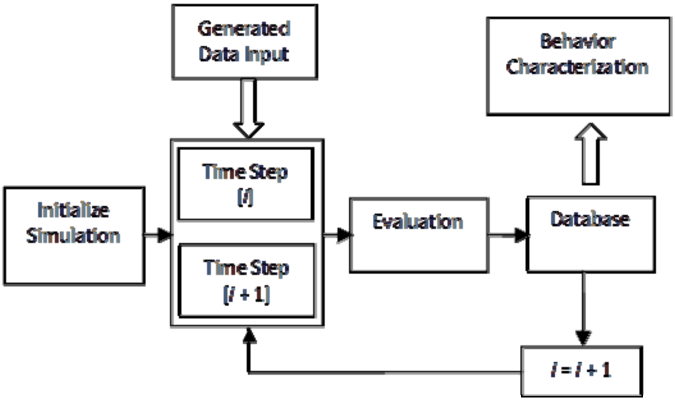

Fig. 5. Generation of behavior characterization data

The second series of simulation experiments, presented in Fig. 6, will use only experimental data and tries to validate the potential nonlinear mechanisms for the underlying process identified in the previous series of simulation experiments and the characterization process. Just like in the first simulation model, for each time period [i, i + 1] between two consecutive time steps [i], [i + 1], the algorithms embedded in the evaluation engine are run independently or in combinations of two or more. The potential matches of the simulation output with previously observed data are monitored and stored in a temporary database. Running the potential complex model for all available observed data will provide sufficient information to decide if there is a fit between the experimental data and the output of the identified algorithm or combination of algorithms. To validate a potential complex model, a statistical analysis of the output given by the evaluation engine will be performed.

The evaluation engine of the simulation modules is the same for both series of experiments and is presented in Fig. 7. Borrowing from multi-agent and holonic systems theory, each of the potential complex models to be tested individually or in combination is modeled as a holon having autonomy in running independently and also having the ability to cooperate with the other holons in the architecture, such that combinations of two or more complex models can be tested. The types of complex models tested are fed with generated or observed data, corresponding to the first or the second series of simulation experiments, respectively.

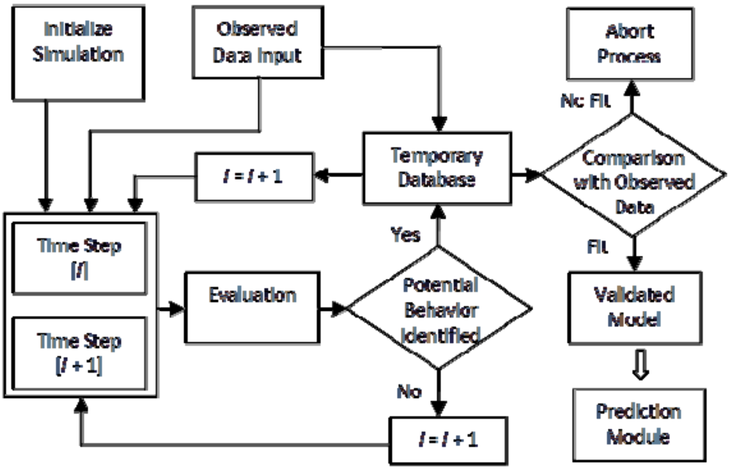

Fig. 6. Simulation logic that validates selected complex models

3.6 Simulation Input Modeling

The purpose of the simulation part is to find the complex model that can deliver the observed output considering the known input of the process under study. Since the amount of experimental observed data may not be sufficient for a massive simulation, large databases of input data will be generated in the form of probability distributions. It is more than likely that no theoretical distribution will provide an adequate fit for the observed data, so to generate large amounts of input data an empirical distribution needs to be used. Bézier distributions are an alternative to pure empirical distributions and have the advantage that the distribution function can have any shape, in contrast with pure empirical distributions were observed data need to be sorted in increasing order.

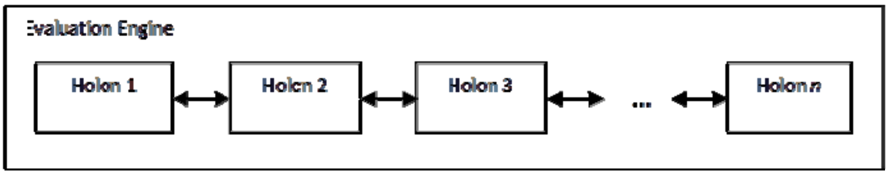

Fig. 7. Internal holonic architecture of the evaluation engine

To specify a probability distribution that models a set of observed data X_1, X_2, …, X_n using Bézier distributions a continuous random variable X with finite range $[a, b]$ is considered. A distribution function $F(x)$ is approximated closely by a Bézier distribution with a degree of certainty m. The Bézier distribution is constructed by fitting a curve to a specified number of points called control points [20].

Let $\{p_0, p_1, \ldots, p_n\}$ be a set of control points, where $p_i = (y_i, z_i)$ with $i = 1, 2, \ldots, n$. Then, the Bézier distribution function $P(t)$ for $t \in [0, 1]$ is given by:

$$P(t) = \sum_{i=0}^{n} B_{n,i}(t) p_i \tag{1}$$

$$B_{n,i}(t) = \frac{n!}{i!(n-i)!} t^i (1-t)^{n-i} \tag{2}$$

Let $F(x; m, y, z)$ be the empirical distribution defined by the Bézier distribution function above, where y and z are the vectors of y_i's and z_i's respectively. Then, for a defined m, using a suitable optimization technique (e.g., least-square estimation), $F(x; m, y, z)$ is fit to the X_i's [21].

3.7 Simulation Output Analysis

The paired-*t* approach is used to build a *confidence interval* (CI) on the difference between the expected response of the evaluation engine when using observed data and the actual observed data. Since the paired-*t* approach requires that observations be identically distributed and independent, bunches of replications are simulated and their mean is taken in consideration when constructing the confidence intervals [22]. The paired-*t* confidence interval method does require also that the number of observations be equal for the two sets of data. This condition is satisfied since the evaluation engine uses the actual observed data as input. If $\overline{X}_{Sim-Obs}$ is the sample mean of the random variable that denotes the difference between the simulated and observed data, $t_{n-1,1-\alpha/2}$ is the upper $(1-\alpha/2)$ critical point for the t distribution with $(n-1)$ degrees of freedom, and, $Var\lfloor \overline{X}_{Sim-Obs} \rfloor$ is an unbiased estimator calculated using the sample standard deviation, then an $100(1 - \alpha)$ CI on the expected difference between the response of the evaluation engine and the actual observed data is given by:

$$\overline{X}_{Sim-Obs} \pm t_{n-1,1-\alpha/2} \sqrt{\lfloor \overline{X}_{Sim-Obs} \rfloor} \tag{3}$$

4 Simulation Results

This section presents a hypothetical complex enterprise system operational scenario for which the proposed modeling and computational framework is applied step by step. Probabilistic models for the Descriptive and Predictive Components are considered, and their outcomes, the *Complex Model Identification* and the *Complex Behavior Identification*, are derived statistically. To identify the underlying complex model, the hypothetical operational scenario considers probability distributions for the occurrence of three of the input modeling measures: the identification and ranking of system goals, the identification and ranking of the complexity factors, and the identification and ranking of the system performance measures. These probability distributions are used as input for the 1,000 trials Monte Carlo simulation model developed to derive the 95% confidence intervals for the Complex Model Identification measure, the outcome of the Descriptive Component, which are presented in Fig. 8 below.

Using the input given by the Monte Carlo simulation, the *Predictive Component* considers a large-scale simulation model to identify the probability of detection the behavior of the hypothetical system for a 50-time step, with the results depicted in Fig. 9.

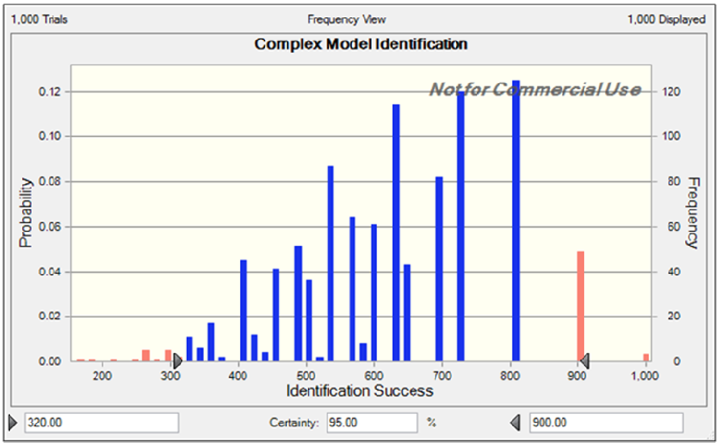

Fig. 8. Complex Model Identification for the hypothetical system

The above results can be interpreted as follows: it is likely that the system performance measures and the underlying system complex model are identified, given that adequate information about the system goals and the complexity factors that may influence the operational scenarios are well defined and considered in the model. Based on the candidate complex models, the next step in the proposed framework (i.e., the Predictive Component) is to run large-scale simulation experiments and attempt to predict the behavior of the hypothetical system.

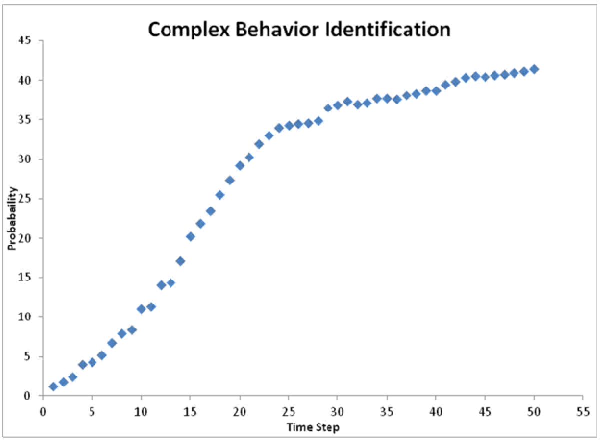

Fig. 9. Complex Behavior Identification for the hypothetical system

As it can be seen form Fig. 9, the probability of detection is low at the beginning of simulation and increases with the simulation time step, as more models are added to the database of complex models behaviors and potential matches are compared with the database of observed and validated models.

5 Conclusions and Future Work

The proposed framework and approach may provide the capability for behavior prediction of large-scale complex enterprise systems. Since the present and future's engineered systems need to adapt to changes in requirements and deal with environment uncertainties, while in operation, behavior prediction becomes a very important capability to be contained in the enterprise systems' toolboxes. The near future work will look at applying the proposed framework to actual manufacturing or service complex enterprise system, such as complex demand network systems. Validation of the predictions can be obtained by comparing them with the performance of, let's say, the demand network system over a certain period of time. Moreover, the framework can be further improved as a result of gaining more understanding of the underlying processes taking place in concepts such as emergence, self-organization, and evolution in complex enterprise systems.

References

1. Calvano, C.N., John, P.: Systems Engineering in an Age of Complexity. Systems Engineering 7(1), 25–34 (2004)
2. Chu, D., Strand, R., Fjelland, R.: Theories of Complexity: Common Denominators of Complex Systems. Complexity 8(3), 19–30 (2003)
3. Choi, T.Y., Dooley, K.J., Rungtusanatham, M.: Supply Networks and Complex Adaptive Systems: Control Versus Emergence. Journal of Operations Management 19, 351–366 (2001)
4. Blanchard, B.S., Fabrycky, W.J.: Systems Engineering and Analysis, 4th edn. Pearson Education, Inc., Upper Saddle River (2006)
5. Minai, A.A., Braha, D., Bar-Yam, Y.: Complex Engineered Systems: A New Paradigm. In: Braha, D., Minai, A.A., Bar-Yam, Y. (eds.) Complex Engineered Systems: Science Meets Technology. NECSI, Cambridge (2006)
6. von Bertalanffy, L.: General System Theory: Foundations, Development, Applications. George Braziller, Inc., New York (1968)
7. Ottino, J.M.: Engineering Complex Systems. Nature 427, 399 (2004)
8. Hitchins, D.K.: Advanced Systems Thinking, Engineering, and Management. Artech House, Inc., Norwood (2003)
9. Sprott, J.C.: Can a Monkey with a Computer Create Art? Nonlinear Dynamics, Psychology, and Life Sciences 8(1), 103–114 (2004)
10. Boccara, N.: Modeling Complex Systems. Springer, New York (2004)
11. Polack, F., Stepney, S.: Emergent Properties Do Not Refine. Electronic Notes in Theoretical Computer Science 137(2), 163–181 (2005)
12. Johnson, C.W.: What are Emergent Properties and How Do they Affect the Engineering of Complex Systems. Reliability Engineering and System Safety 91(12), 1475–1481 (2006)

13. Fayyad, U., Piatetsky-Shapiro, G., Smyth, P.: From Data Mining to Knowledge Discovery: An Overview. In: Advances in Knowledge Discovery and Data Mining, pp. 1–34. AAAI Press (1996)
14. McGarry, K.: A Survey of Interesting Measures for Knowledge Discovery. The Knowledge Engineering Review 20(1), 39–61 (2005)
15. Last, M., Klein, Y., Kandel, A.: Knowledge Discovery in Time Series Databases. IEEE Transactions on Systems, Man, and Cybernetics - Part B: Cybernetics 31(1), 160–169 (2001)
16. Macal, C.M., North, M.J.: Tutorial on Agent-Based Modeling and Simulation. In: Proceedings of the Winter Simulation Conference, Orlando, FL, pp. 2–15 (2005)
17. Wainer, G.A.: Modeling and Simulation of Complex Systems with Cell-DEVS. In: Proceedings of the Winter Simulation Conference, Washington, DC, pp. 49–60 (2004)
18. Bar-Yam, Y.: Dynamics of Complex Systems. Westview Press, Perseus Books Group, Boulder, CO (2003)
19. Rouse, W.B.: Complex Engineered Organizational and Natural Systems: Issues Underlying the Complexity of Systems and Fundamental Research Needed to Address these Issues. Systems Engineering 10(3), 260–271 (2007)
20. Nelson, B.L., Yamnitsky, M.: Input Modeling Tools for Complex Problems. In: Proceedings of the Winter Simulation Conference, Washington, DC, pp. 105–112 (1998)
21. Law, A.M., Kelton, W.D.: Simulation Modeling and Analysis. McGraw-Hill Companies, Inc., Boston (2000)
22. Babiceanu, R.F., Chen, F.F.: Distributed and Centralized Material Handling Scheduling: Comparison and Results of a Simulation Study. Robotics and Computer Integrated Manufacturing 25(2), 441–448 (2009)