

Final Report

Developing an Agent-Based Modeling Verification and Validation Approach for Improving Air Force Analytical Support

Brian L. Heath* and Raymond R. Hill †

BIE Department, Wright State University
3640 Colonel Glenn Hwy, Dayton, OH 45435, United States
heath.19@wright.edu*
rayrhill@gmail.com†

Abstract

Models and simulations have been widely used as a means to predict the performance of systems. Agent-based modeling and agent distillations have recently found tremendous success particularly in analyzing ground force employment and doctrine. They have also seen wide use in the social sciences modeling a plethora of real-life scenarios. The use of these models always raises the question of whether the model is correctly encoded (verified) and accurately or faithfully represents the system of interest (validated). A variety of groups are involved at investigating agent-based model verification and validation. This project seeks a new look at the subject. This project examined the agent-based modeling problem from a historical, first principles perspective, examines the basic philosophies of model validation, develops and presents via a case study an initial agent-based modeling developmental methodology more amenable to validation (or rather sanctioning) and presents the results of applying that methodology to the agent-based modeling case study.

Executive Summary

Models and simulations have been widely used as a means to predict the performance of systems. Agent-based modeling and agent distillations have recently found tremendous success particularly in analyzing ground force employment and doctrine. They have also seen wide use in the social sciences modeling a plethora of real-life scenarios. The use of these models always raises the question of whether the model is correctly encoded (verified) and accurately or faithfully represents the system of interest (validated). A variety of groups are involved at investigating agent-based model verification and validation. This project seeks a new look at the subject. This project examined the agent-based modeling problem from a historical, first principles perspective.

This final report documents the results of this project, a project that involved four phases. In phase one, we took an in-depth historical look at agent-based modeling, not from the general approach of finding early agent-based modeling papers but rather from the perspective of what were the scientific streams of thought that influenced and shaped the current agent-based modeling paradigm. We then took a similar in-depth look at the philosophies of model validation and tied the various philosophies to their influence on agent-based modeling. These efforts are documented in Sections 1 and 2, respectively.

The Bay of Biscay was a WWII application of that newly titled systems approach to problem solving called Operations (or Operational) Research. The historical record of the Bay of Biscay German U-Boat versus Allied Bomber, sub-hunting campaign has long been the course of operational research analysis case studies. The Bay of Biscay has also been used as a case study for agent-based modeling, most recently in [16], [32], and [31]. In Section 3 we describe the Bay of Biscay case study scenario building mostly upon the work in [16].

In Section 4 we define the requirements for an agent-based modeling sanctioning method and lay out an initial methodology suitable for such sanctioning. To motivate this work, we spend time discussing how simulationists develop models. This discussion is based upon a review of the literature inferring from the authors' discussion how they went about building their models. In short, agent-based modeling, with its bottom-up design philosophy, requires sanctioning methods that capture the detail embedded within the models. Thus, we spend time discussing the role of conceptual modeling.

An initial prototype for an agent-based developmental methodology is presented in Section 5. The focus is to provide the modeling details within a conceptual design framework that would ultimately promote the sanctioning of ensuing agent-based models. This methodology is tested against the Bay of Biscay case study. In our particular case, this was limited to Scenario 1 from the work in [16].

Contents

1	Introduction and Background of Agent-Based Modeling	6
1.1	The Beginning: Computers	6
1.2	The Synthesis of Natural Systems: Cellular Automata and Complexity	8
1.3	The Analysis of Natural Systems: Cybernetics and Chaos	11
1.4	Towards Today's ABM: Complex Adaptive Systems	16
1.5	Section Concluding Remarks	18
2	Philosophies of Validation: Can One Actually Validate Models?	20
2.1	Why All Simulations are Invalid	22
2.2	What Does Simulation Validation Really Mean in Practice?	28
2.3	What Good are Simulations?	33
2.4	What Good is ABM?	40
2.5	Section Concluding Remarks	43
3	Case Study and Developmental Methodology	44
3.1	Campaign Issues	46
3.2	Previous Analyses	47
3.3	So Why This Scenario?	48
4	Defining Some ABM Sanctioning Requirements	48
4.1	How Models are Developed	49
4.2	Sanctioning Emphasis	51
4.3	Conceptual Modeling	53
4.4	Systems for which the ABM Paradigm is Useful	55
4.5	Summary	56
5	General Observations of Current ABM Sanctioning Concepts	57
5.1	Understanding the Needs: How to Represent a Complex System	63
5.2	Prototype Design of the New ABM Tool: Theoretical System Simulation Diagram	66
5.2.1	Fundamental Characteristics	66
5.2.2	Basic TSSD Prototype Structure	67
5.3	Section Concluding Remarks	70
6	Proof of Concept Case Study	70
6.1	Scenario 1 of the Bay of Biscay ABM Simulation	71
6.2	Bay of Biscay TSSD Prototype Description	74
6.3	Sanctioning the Reproduced Bay of Biscay ABM Simulation	87
7	Concluding Remarks and the Next Steps Towards Improving and Extending the TSSD Prototype	90
	References	93

List of Tables

1	Cellular Automata Classifications [87]	11
2	Comparison of Complex Systems Classifications	13
3	TSSD Prototype Shape Descriptions	69

List of Figures

1	Relationship between a System, a Theory/Model, and a Simulation	23
2	Different Roles of a Simulation	39
3	Map of Bay of Biscay with Locations of U-boat sinkings, 1942-1944	45
4	Different Roles of a Simulation	49
5	A Simplified Simulation Development Process	50
6	Relationship between System Understanding and Simulation Sanctioning Emphasis .	52
7	TSSD Prototype - Bay of Biscay Model View	75
8	TSSD Prototype - Environment View	76
9	TSSD Prototype - Uboat View	77
10	TSSD Prototype - Aircraft View	78
11	Uboat Decision Shape Property Example	79
12	Uboat Action Shape Properties	81
13	Uboat Block Shape Properties	82
14	Uboat Decision Shape Properties	83
15	Uboat Interaction Shape Properties	84
16	Bay of Biscay ABM Simulation Screen Shot	87
17	Bay of Biscay ABM Simulation Results	88
18	Two-sided Two-Sample t-Test Results	89

1 Introduction and Background of Agent-Based Modeling

Over the years Agent-Based Modeling (ABM) has become a popular tool used to model and understand the many complex, nonlinear systems seen in our world [23]. As a result, many papers geared toward modelers discuss the various aspects and uses of ABM. The topics covered typically include an explanation of ABM, when to use it, how to build it and with what software, how results can be analyzed, research opportunities, and discussions of successful applications of the modeling paradigm. It is also typical to find within these papers brief discussions about the origins of ABM, discussions that tend to emphasize the diverse applications of ABM as well as how some fundamental properties of ABM were discovered. However, these historical discussions often do not go into much depth about the fundamental theories and fields of inquiry that would eventually lead to ABM's emergence. Thus, in this report we re-examine some of the scientific developments in computers, complexity, and systems thinking that helped lead to the emergence of ABM by shedding new light onto some old theories and connecting them to several key ABM principles of today. This report should not be considered a complete account of the field, but instead provides a historical perspective into ABM and complexity intended to provide a clearer understanding of the field, show the benefits to be gained by understanding the diverse origins of ABM, and hopefully spark further interest into the many other theories and ideas that laid the foundation for the ABM paradigm of today.

1.1 The Beginning: Computers

The true origins of ABM can be traced back hundreds of years to a time when scientists first began discovering and attempting to explain the emergent and complex behavior seen in nonlinear systems. Some of these more familiar discoveries include Adam Smith's Invisible Hand in Economics, Donald Hebb's Cell Assembly, and the Blind Watchmaking in Darwinian Evolution [4]. In each of these theories simple individual entities interact with each other to produce new complex phenomena that seemingly emerge from nowhere. In Adam Smith's theory, this emergent phenomena is the Invisible Hand, which occurs when each individual tries to maximize their own interests and as

a result tend to improve the entire community. Similarly, Donald Hebb's Cell Assembly Theory states that individual neurons interacting together form a hierarchy that results in the storage and recall of memories in the human brain. In this case, the emergent phenomena is the memory formed by the relatively simple interactions of individual neurons. Lastly, the emergent phenomena in Darwinian Evolution is that complex and specialized organisms resulted from the interaction of simple organisms and the principles of natural selection.

Although these theories were brilliant for their time, in retrospect, they appear marred by the prevalent scientific philosophy of the time. Newton's Philosophy, which is still common today, posited that given an approximate knowledge of a system's initial condition and an understanding of natural law, one can calculate the approximate future behavior of the system [27]. Essentially, this view creates the idea that nature is a linear system reducible into parts that eventually can be put back together to resurrect the whole system. Interestingly, it was widely known at the time that there were many systems where this reductionism approach did not work. These type of systems were called nonlinear because the sum output of the parts did not equal the output of the whole system. One of the more famous nonlinear systems is the Three Body Problem of classical mechanics, which shows that it is impossible to mathematically determine the future states of three bodies given the initial conditions.

Despite observing and theorizing about emergent behavior in systems, scientists of the time did not have the tools available to fully study and understand these nonlinear systems. Therefore, it was not until theoretical and technological advances were made that would lead to the invention of the computer that scientists could begin building models of these complex systems to better understand their behavior. Some of the more notable theoretical advances that led to the invention of the computer were first made by Gödel with his famous work in establishing limitations of mathematics [15] and then by Turing in 1936 with his creation of the Turing Machine. The fundamental idea of the theoretical Turing Machine is that it can replicate any mathematical process, which was a big step in showing that machines were capable of representing systems. Furthermore, Turing and Church later developed the Church-Turing Hypothesis which hypothesized that a machine

could duplicate not only the functions of mathematics, but also the functions of nature [52]. With these developments, scientists had the theoretical foundation onto which they could begin building machines to try and recreate the nonlinear systems they observed in nature.

Eventually, these machines would move from theoretical ideas to the computers that we are familiar with today. The introduction of the computer into the world has certainly had a huge impact, but its impact in science as more than just a high speed calculator or storage device is often overlooked. When the first computers were introduced, Von Neumann saw them as having the ability to “break the present stalemate created by the failure of the purely analytical approach to nonlinear problems” by giving scientists the ability to heuristically use the computer to develop theories [77]. The heuristic use of computers, as viewed by Von Neumann and Ulam, is very much like the traditional scientific method except that the computer replaces or supplements the experimentation process [77]. By using a computer to replace real experiments, Von Neumann’s process would first involve making a hypothesis based on information known about the system, building the model in the computer, running the computer experiments, comparing the hypothesis with the results, forming a new hypothesis, and repeating these steps as needed [77]. The essential idea of this empirical method is to understand that the computer serves as a simulation of the real system, which allows more flexibility in collecting data and controlling conditions as well as better control of the timeliness of the results.

1.2 The Synthesis of Natural Systems: Cellular Automata and Complexity

Once computers were invented and became established, several different research areas appeared with respect to understanding natural systems. One such area was focused primarily on synthesizing natural systems [49] and was led primarily by the work of Von Neumann and his theory on self-reproducing automata, which are self-operating machines or entities. In a series of lectures, Von Neumann presents a complicated machine that possesses a blueprint of information that controls how the machine acts, including the ability to self-reproduce [77]. This key insight by Von Neumann to focus not on engineering a machine, but on passing information was a precursor to the discovery of DNA which would later inspire and lead to the development of genetic algorithm search processes.

However, despite his many brilliant insights, Von Neumann's machine was very complicated since he believed that a certain level of complexity was required in order for organisms to be capable of life and self-reproduction [52]. Although it is certainly true that organisms are fairly complex, Von Neumann seemed to miss the idea that would later be discovered that global complexity can emerge from simple local rules [27].

With the idea that complexity was needed to produce complex results, reductionism still being the prevalent scientific methodology employed, and perhaps spurred on by the idea of powerful serial computing capabilities, many scientists began trying to synthesize systems from the top-down. As briefly discussed earlier, the idea of top-down systems analysis is to take the global behavior, decompose it into small pieces, understand those pieces, and then put them back together to reproduce or predict future global behavior. This top-down methodology was primarily employed in the early applications of Artificial Intelligence, where the focus was more on defining the rules of intelligence-looking and creating intelligent solutions rather than the focus being on the structure that creates intelligence [15]. Steeped in the traditional idea that systems are linear, this approach did not prove to be extremely successful in understanding the complex nonlinear systems found in nature [49].

Although Von Neumann believed that complexity was needed to represent complex systems, his colleague Ulam suggested that this self-reproducing machine could be more easily represented using a Cellular Automata (CA) approach. As the name may suggest, CA are self-operating entities that exist in individual cells that are adjacent to one another in a 2-D space like a checkerboard and have the capability to interact with the cells around it. The impact of taking the CA approach was significant for at least two reasons. The first is that CA is a naturally parallel system where each cell can make autonomous decisions simultaneously with other cells in the system [49]. This change from serial to parallel systems was significant because it is widely recognized that many natural systems are parallel [77]. The second reason the CA approach had a significant impact on representing complex systems is that CA systems are composed of many locally controlled cells that together create a global behavior. This CA architecture requires engineering a cell's logic at the

local level in hopes that it will help create the desired global behavior [49]. Ultimately, CA would lead to the bottom-up approach now mainly employed by the field of Artificial Life because it is more naturally inclined to produce the same global behavior that is seen to emerge in complex, nonlinear systems.

Eventually Von Neumann and Ulam were able to successfully create a paper-based self-reproducing CA system which was much simpler than Von Neumann's previous efforts. As a result, some scientists began using CA systems to synthesize and understand complexity and natural systems. Probably the most notable and famous use of CA was Conway's "Game of Life." In this CA system, which started out as just a Go Board with pieces representing the cells, only three simple rules were used by each cell to determine whether it would be colored white or black based on the color of cells around it. Using this game, it was found that depending upon the starting configuration, certain shapes or patterns such as the famous glider would emerge and begin to move across the board where it might encounter other shapes and create new ones as if mimicking a very crude form of evolution. After some research, a set of starting patterns were found that would lead to self-reproduction in this very simple system [52]. For more information on the "Game of Life," to see some of the famous patterns, and to see the game in action the reader can go to http://en.wikipedia.org/wiki/Conway's_Game_of_Life. However, this discovery that simple rules can lead to complex and unexpected emergent behavior was not an isolated discovery. Many others would later come to the same conclusions using CA systems, including Schelling's famous work in housing segregation which showed that the many micromotives of individuals can lead to macrobehavior of the entire system [69].

Upon discovering that relatively simply CA systems were capable of producing emergent behavior, scientists started conducting research to further determine the characteristics and properties of these CA systems. One of the first of these scientists was mathematician Wolfram, who published a series of papers in the 1980's on the properties and potential uses of 2-dimensional CA. In his papers, Wolfram creates four classifications into which different CA systems can be placed based on their long-term behavior. A description of these classifications is found in Table 1. Langton would

Table 1: Cellular Automata Classifications [87]

Class	Properties
1	Evolves to a homogeneous state, changes to the initial state has no impact on final state
2	Evolves into a set of simple periodic states, changes to the initial state has a finite regional impact on the final state
3	Evolves into patterns that grow indefinitely, changes to the initial state leads large changes to the final state
4	Evolves to complex localized patterns that expand and contract with time, changes to the initial state leads to irregular changes to the final state

later take this research further and described that life, or the synthesis of life, exists only in class 4 systems, which is to say that life and similar complex systems exist between order and complete instability [52]. As a result, it was concluded that in order to create complex systems that exhibit emergent behavior, one must be able to find the right balance between order and instability or else the system will either collapse on itself or explode indefinitely.

Armed with these discoveries about synthesizing complex systems and emergent behavior, many scientists in the fields of ecology, biology, economics, and other social sciences began using CA to model systems that were traditionally very hard to study due to their nonlinearity [20]. However, as technology improved, the lessons learned in synthesizing these nonlinear systems with CA would eventually lead to models where autonomous agents would inhabit environments free from restriction of their cells. Such models include Reynold’s “boids” which exhibited the flocking behavior of birds and Langton’s “vants” which exhibited the behavior of ants [52]. Advanced studies include the influential Epstein and Axtell [20] exposition of CA models involving their Sugarscape model and Illachinski’s [?] ISAAC effort that arguably introduced the military to the use of CA. However, to better understand agents, their origins, and behaviors another important perspective of agents, the analysis of natural systems, should be examined.

1.3 The Analysis of Natural Systems: Cybernetics and Chaos

While Von Neumann was working on his theory of self-reproducing automata and asking, ‘what makes a complex system,’ Wiener and others were developing the field of cybernetics [49] and

asking the question, 'what do complex systems do [1]?' Although these two questions are related, each is clearly focused on different aspects of the complexity problem and led to two different, but related, paths toward discovering the nature of complexity, the latter course of inquiry become cybernetics. According to Wiener, cybernetics is “the science of control and communication in the animal and the machine” [80] and has its origins in the control of the anti-aircraft firing systems of World War II [49]. Upon fine tuning the controls, scientists found that feedback and sensitivity were very important and began formalizing theories about the control and communications of these systems having feedback. Eventually they would discover that the same principles found in the control of machines were also true for animals, such as the activity of recognizing and picking up an object [80]. This discovery would lead cybernetics to eventually be defined by Ashby as a “field concerned with understanding complexity and establishing foundations to study and understand it better” [1], which includes the study of both machines and organisms as one system entity.

One of the main tools used in cybernetics to begin building theories about complex systems was Information Theory as it allowed scientists to think about systems in terms of coordination, regulation, and control. Armed with this new mathematical theory of the time, those studying cybernetics began to develop and describe many theories and properties of complex systems. One of these discoveries about complex systems was the importance of feedback on the long-term patterns and properties of complex systems. In general, complex systems consist of a large number of tightly coupled pieces that together receive feedback that influences the system's future behavior. Based on this information, Ashby explains that complex systems will exhibit different patterns depending upon the type of feedback found in the system. If the feedback is negative (i.e., the Lyapunov Exponent, $\lambda < 0$), then the patterns will become extinct or essentially reach a fixed point. If the feedback is zero ($\lambda = 0$), then the pattern will remain constant or essentially be periodic. Finally, if the feedback is positive ($\lambda > 0$), then the patterns would grow indefinitely and out of control [1].

It is important to note that these three properties of complex systems, as expressed by Ashby in 1962, directly relate to Wolfram's first three classifications of two-dimensional CA systems. However, just as Von Neumann failed to make certain observations about complexity, so did the founders of

Table 2: Comparison of Complex Systems Classifications

Wolfram's CA Classification	Lyapunov's Exponent Value	Properties
1	all $\lambda < 0$	Evolves to a homogeneous (fixed attractor)
2	$\lambda = 0$ & $\lambda < 0$	Evolves into a set of simple periodic states (periodic attractor)
3	all $\lambda > 0$	Evolves into patterns that grow indefinitely
4	$\lambda > 0$ & $\lambda < 0$	Evolves to chaotic state (strange attractor)

cybernetics fail to consider what would happen if both positive and negative feedback simultaneously existed in a system. It was not until later that Shaw used Information Theory to show that if at least one component of a complex system has a positive Lyapunov Exponent, and was mixed with other components with varying exponent values, then the system will exhibit chaotic patterns [27]. As it may be expected, this classification of systems directly relates to Wolfram's fourth classification of two-dimensional CA systems, where life is thought to exist. Table 2 provides our comparisons between Wolfram's classifications and the discussed properties of complex systems.

With Shaw's discovery that complex systems can exhibit chaotic behavior, scientists began considering what further impacts Chaos Theory might have on understanding complex systems. In general, chaos can be defined as deterministic randomness, such that any system exhibiting chaos will appear to behave randomly with the reality being that the behavior is completely deterministic [15]. However, this does not mean that the system is completely predictable. As Lorenz was first to discover with his simulation of weather patterns, it is impossible to make long-term predictions of a chaotic system with a simulated model because it is infeasible to record all of the initial conditions at the required level of significance [27]. This sensitivity to initial conditions results from the fact that the initial conditions are infinitely many random numbers, which implies they are incompressible and infinitely long. Therefore, collecting these initial conditions to the required level of significance is impossible without a measurement device capable of collecting an infinite number of infinitely long numbers as well as finding a computer capable of handling all of those infinitely long numbers.

It may seem that this property of chaos has at some level discredited the previously mentioned

Church-Turing Hypothesis by suggesting that these types of natural complex systems cannot be duplicated by a machine. However, there are several other properties of chaos that help those attempting to model and understand these complex systems despite the inability to directly represent them. The first is that chaotic systems have a strange attractor property that keep these aperiodic systems within some definable region [27]. This is obviously good for those studying these complex systems because it limits the region of study into a finite space. The other property of these complex systems is that they can be generated using a very simple set of rules or equations. By using a small set of rules or equations, and allowing the results to act as a feedback into the system, the complexity of these systems seems to emerge out of nowhere. As one can recall, the same discovery was made in CA when cells with simple rules were allowed to interact dynamically with each other [27]. Therefore, it appears that although natural complex systems cannot be modeled directly, some of the same emergent properties and behavior of these systems can be generated in a computer using simple rules (i.e., the bottom-up approach) without complete knowledge of the entire real system. Perhaps it is not surprising that the idea that complex systems can be represented sufficiently with a simpler model, often called a Homomorphic Model, has long been a fundamental concept when studying complex systems [1].

Whenever discussing the idea that simple rules can be used to model complex systems it is valuable to mention fractals, which are a closely related to and often a fundamental component of Chaos Theory. First named by Mandelbrot, fractals are geometric shapes that regardless of the scale show the same general pattern. The interesting aspect of fractals is that because of their scale-free, self-similar nature they can both fit within a defined space and have an infinite perimeter, which makes them complex and relates them very closely to the effect strange attractors can have on a system. Furthermore, forms of fractals can be observed in nature and, in turn, generated in labs using very simple rules, which shows that they also exhibit the same type of emergent behavior and properties as the previously discussed complex systems [27]. As a result, although fractals, chaos, and complex systems have a lot in common, fractals, due to their physical representation, provide an insightful look into the architecture of complexity. Essentially, fractals are composed

of many similar subsystems of infinitely many more similar subsystems of the same shapes, which results in a natural hierarchy and the emergence of other, similar shapes. It is interesting to note that the architecture of fractals directly shows why reductionism does not work for nonlinear systems. With fractals, a scientist could forever break the fractal into smaller pieces and never be able to measure its perimeter. Another interesting aspect about the architecture of fractals is that they naturally form a hierarchy, which means the properties of hierarchies could possibly be exploited when attempting to model and understand complex systems. For example, the fact that Homomorphic models are effective at modeling complex systems could come from the fact that hierarchical systems are composed of subsystems such that the subsystems can be represented not as many individual entities but as a single entity [72].

Besides showing that emergent behavior can be explained using chaos, which in turn can be simply represented in a model, there are other properties of chaos which give insight into complex natural systems and ABM. Returning to the idea that it is impossible to satisfactorily collect all of the initial conditions to obtain an exact prediction of a chaotic system, one might ask what would happen if the needed initial conditions were collected, but not to the infinite level of detail? It turns out that such a model would be close for the very short term, but would eventually diverge from the actual system being modeled. This example brings about another property of chaotic systems; they are very sensitive to the initial conditions [15]. Because this sensitivity property of chaos ultimately leads to unreliable results when compared to the actual system and the models only being homomorphic, it can be seen that these computer models are unlikely to aid any decision about how to handle the real system. Instead, it can be concluded that these models should be used to provide insights into the general properties of a complex system and not for forecasting 'hard' statistics like mean performance. Essentially, this methodology of using a computer for inference and insight harps back to Von Neumann's idea of using a computer to facilitate an experiment with hopes to gain insights about the system rather than using the computer to generate exact results about the future states of the system [77].

The final property of chaos that can give insight into complex natural systems and ABM is that

a strange attractor not only limits the state space of the system, but it also causes the system to be aperiodic. In other words, the system with a strange attractor will never return to a previous state, which results in tremendous variety within the system [15]. In 1962, Ashby examined the issue of variety in systems and posited the Law of Requisite Variety, which simply states that the diversity of an environment can be blocked by a diverse system [1]. In essence, Ashby's law shows that in order to handle a variety of situations, one must have a diverse system capable of adapting to those various situations. As a result, it is clear that variety is important for natural systems given the diversity of the environment in which they can exist. In fact, it has been seen that entities within an environment will adapt to create or replace any diversity that have been removed, further enforcing the need and importance of diversity [34]. However, it has also been found that too much variety can be counter productive to a system because it can grow uncontrollably and be unable to maintain improvements [4]. Therefore, it appears that complex natural systems that exhibit emergent behavior need to have the right balance between order and variety or positive and negative feedback, which is exactly what a strange attractor does in a chaotic system. By keeping the system aperiodic within definable bounds, chaotic systems show that the battle between order and variety is an essential part of complex natural systems. As a result, strange attractors provide systems with the maximum adaptability.

1.4 Towards Today's ABM: Complex Adaptive Systems

After learning how to synthesize complex systems and discovering some of their properties, the field of Complex Adaptive Systems (CAS), which is commonly referenced as the direct historical roots of ABM, began to take shape. Primarily, the field of CAS draws much of its of inspiration from biological systems and is concerned mainly with how complex adaptive behavior emerges in nature from the interaction among autonomous agents [53]. One of the fundamental contributions made to the field of CAS, and in turn ABM, was Holland's identification of the four properties and three mechanisms that compose all CAS [34]. Essentially, these items have aided in defining and designing ABM as they are known today [53] because Holland takes many of the properties of complex systems discussed earlier and places them into clear categories, allowing for better focus,

development, and research.

The first property of CAS discussed by Holland is Aggregation, which essentially states that all CAS can be generalized into subgroups and similar subgroups can be considered and treated the same. As can be seen, this property of CAS directly relates to the hierarchical structure of complex systems discussed early. Furthermore, not only in 1962 did Simon discuss this property of complex systems, he also discussed several other hierarchical ideas about the architecture of complex systems [72] that can be related to two of Holland's mechanisms of CAS. The first is Tagging, which is the mechanism that classifies agents, allows the agents to recognize each other, and allows easier observation of the system. Essentially, this classification is nothing more than a means of putting agents into subgroups within some sort of hierarchy. The second mechanism is Building Blocks, which is the idea that simple subgroups can be decomposed from complex systems that in turn can be reused and combined in many different ways to represent patterns. Besides being related to Simon's discussion of the decomposability of complex systems, this mechanism also reflects the common theme that simplicity can lead to emergent behavior and the theory behind modeling a complex system. Therefore, it can be seen that the elements of Aggregation, Tagging, and Building Blocks can be related back to the results discovered by Simon when studying the architecture of complexity.

Another property of CAS is Nonlinearity, which, as previously discussed, is the idea that the whole system output is greater than the sum of the individual component output. In essence, the agents in a CAS come together to create a result such that it cannot be attributed back to the individual agents. Hopefully, it is now clear that not only is this fundamental property the inspiration behind synthesizing and analyzing complex systems, but that nonlinearity can also be the result of dynamic feedback and interactions. These causes of nonlinearity can be related to two more of Holland's CAS elements. The first is the property of Flow, which states that agents in CAS communicate and that this communication can change with time. As was seen in examples using CA, having agents communicate with each other and their environment dynamically can lead to the nonlinearity of emergent behavior. Also, within the property of Flow, Holland discusses several

interesting effects that can result from changes made to the flow of information such as the Multiplier Effect and the Recycling Effect. In short, the Multiplier Effect occurs when an input gets multiplied many times within a system. An example of the Multiplier Effect is the impact made on many other markets when a person builds a house. Similarly, the Recycling Effect occurs when an input gets recycled within the system and the overall output is increased. An example of the Recycling Effect is when steel is recycled from old cars to make more new cars [34]. Interestingly enough, both of these effects can be directly related back to Information Theory and Cybernetics. The other element that relates to nonlinearity is the Internal Model Mechanism, which gives the agents an ability to perceive and make decisions about their environment. It is easy to think of this mechanism as being the rules that an agent follows in the model, such as turning colors based on its surroundings or moving away from obstacles. From the previous discussion on CA, and from the reoccurring theme, simple Internal Models can lead to emergent behavior in complex systems. Therefore, the link between these three elements is the essential nature of complex systems: nonlinearity.

The final property discussed by Holland is Diversity. Essentially, Holland states that agents in CAS are diverse, which means they do not all act the same way when stimulated with a set of conditions. By having a diverse set of agents, Holland argues that new interactions and adaptations can develop such that the overall system will be more robust. Of course, the idea that variety creates more robust systems relates directly back to Ashby's Law of Requisite Variety, which in turn relates back to strange attractors and Chaos Theory.

1.5 Section Concluding Remarks

For all of positives of ABM there are often just as many, if not more, criticisms of ABM. For the modeler to successfully defend their model and have it be considered worth any more than a new and trendy modeling technique, the modeler needs to have a fundamental understanding of the many scientific theories, principles and ideas that lead to ABM and not just an understanding of the 'how to' perspective on emergence and ABM. By gaining deeper understandings of the history of ABM, the modeler can better contribute to transforming ABM from a potential modeling revolution [8] to an actual modeling revolution with real life implications. Understanding that ABMs were the result

of the lack of human ability to understand nonlinear systems allows the modeler to see where ABM fits in as a research tool. Understanding the role that computers play in ABM shows the importance of understanding the properties of computers and in turn their limitations. Understanding that the fundamental properties of CAS have their origins in many different fields (Computers, CA, Cybernetics, Chaos, etc) will give the modeler the ability to better comprehend and explain their model. For example, understanding Chaos Theory can reveal why ABMs are thought to be incapable of providing anything more than insight into the model. By understanding each of these individual fields and how they are interrelated, a modeler can potentially make new discoveries and better analyze their model. For example, by understanding the theory behind Cybernetics and Chaos Theory a modeler would be better equipped in determining the impact that certain rules may have on the system or in trouble shooting why the system is not creating the desired emergent behavior. Finally, understanding the history of ABM presents the modeler with a better ability to discern between and develop new ABM approaches.

As it is often the case, examining history can lead to insightful views about the past, present, and the future. It is the hope that this report has shed some light on the origins of ABM as well as the connections between the many fields from which it emerged. Starting with theories about machines, moving onto synthesis and analysis of natural systems, and ending with CAS, it is clear, despite this article being primarily focused on complexity, that many fields played an important role in developing the multidisciplinary field of ABM. Therefore, in accordance with the Law of Requisite Variety, it appears wise for those wishing to be successful in ABM to also be well versed in the many disciplines that ABM encompasses. Furthermore, many insights can be discovered about the present nature of ABM by understanding the theoretical and historical roots that compose the rules-of-thumb used in today's ABM. For example, knowing the theory behind Cybernetics and Chaos Theory could help a modeler in determining the impact that certain rules may have on the system or in trouble shooting why the system is not creating the desired emergent behavior. Finally, it could be postulated that understanding the history of ABM presents one with a better ability to discern between and develop new ABM approaches. In conclusion, this article

has provided an abbreviated look into the emergence of ABM with respect to complexity and has made some new connections to today's ABM that can hopefully serve as a starting point for those interested in understanding the diverse fields that compose ABM.

2 Philosophies of Validation: Can One Actually Validate Models?

Since the introduction of the computer, simulations have become popular in many scientific and engineering disciplines. This is partly due to a computer simulation's ability to aid in the decision making and understanding of relatively complex and dynamic systems where traditional analytical techniques may fail or be impractical. As a result of this ability, the use of simulations can be found in just about every field of study. These fields can range anywhere from military applications [18] and meteorology [46] to management science [63], social science [20], nanotechnology [39], and terrorism [65]. What can be inferred from this wide spread use is that not only are simulations robust in their application, but they are also practically successful. Due in large part to this robustness and success, simulations are becoming a fairly standard tool found in most analyst's toolbox. In fact, proof that simulations are becoming more of a generic analysis tool and less of a new technique can be found in the increasing number of published articles that use simulations but do not mention it in the title [48].

However, despite their increasing popularity, a fundamental issue has continued to plague simulations since their inception [60, 74]: is the simulation an accurate representation of the reality being studied? This question is important because typically a simulation's goal is to represent some abstraction of reality and it is believed that if a simulation does not accomplish this representation, then information gained from the simulation is either worthless or at least not as valuable. Therefore, one can understand why answering the question of simulation validity is so important, because having an accurate simulation could mean that knowledge can be gained about reality without actually observing, experimenting, and dealing with the constraints of reality [77]. As a result of this potential, many articles over the years have been devoted to the topic of simulation validity and in particular they each tend to focus on some aspect of the following fundamental questions of

simulation validity:

- Can simulations represent reality? If not, what can they represent?
- If a simulation cannot or does not represent reality, then is the simulation worth anything?
- How can one show that a simulation is valid? What techniques exist for establishing validity?
- What roles do or should simulations play today?

Given the considerable amount of time and effort spent on simulation validity, a reasonable question to ask is why is simulation validity still haunting simulationists today? In short, the fundamental reason why it is still an issue today, and will continue to be one, is that the question of a simulation's validity is really a philosophical question found at the heart of all scientific disciplines [74]. By considering the above questions, one will notice that they closely resemble some typical philosophy of science questions [40]:

- Can scientific theories be taken as true or approximately true statements of what is true in reality?
- What methods, procedures, and practices make scientific theories believable or true?

Therefore, the philosophy of science perspective can shed light on the nature of simulation validity as it is known today as well as the nature of simulation itself. It is from this fundamental philosophy of science perspective that this article will attempt to give insights into the fundamental questions of simulation validity, where current practices in simulation validation fit into the general framework of the philosophy of science, and what role simulations play in today's world.

With this objective in mind, this article has been divided into four more sections. The first section discusses how the relationship between reality and simulation is flawed such that all simulations do not represent reality. The second section describes what is currently meant by the idea of simulation validation in practice. The third section discusses the usefulness of simulations today and how simulations are becoming the epistemological tool of our time. In the fourth section, the

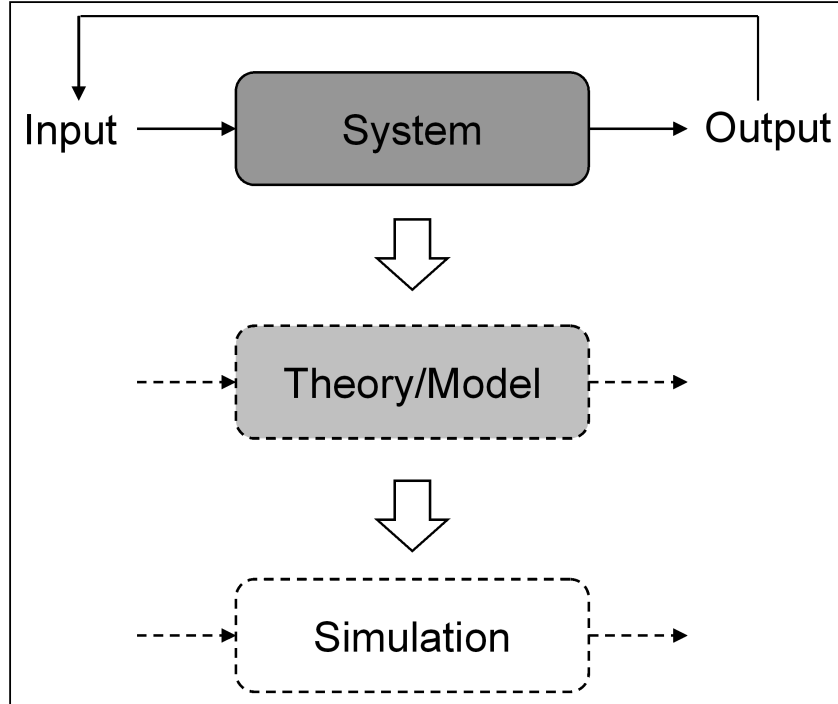
usefulness and role of Agent-Based Models as well as the special case they present to simulation validation is discussed.

2.1 Why All Simulations are Invalid

Prior to proceeding any further, it is valuable to first define simulation and discuss how it is typically seen as related to reality. Although there are many definitions of simulation, for this article a simulation can be generically defined as a numerical technique that takes input data and creates output data based upon a model of a system [51] (for this article the distinction between theory and model will not be made, instead the term model will be used to represent them together). In essence, a simulation attempts to show the nature of a model as it changes over time. Therefore, it can be said that a simulation is a representation of a model and not directly a representation of reality. Instead, it is the model's job to attempt to represent some level of reality in a system. In this case, it would appear that a simulation's ability to represent reality really depends upon the model upon which it is built [18]. Although this relationship between a real system, a model, and a simulation has been described in many different ways [9, 51, 74, 82], a simplified version of the cascading relationship used for this article is shown in Figure 1. Also, before proceeding it is important to note that commonly simulations today are performed by computers because they are much more efficient at numerical calculations. Therefore, we assume for this article that a simulation will be constructed within a computer and that a simulation is a representation of a model which is in turn a representation of a real system (as shown in Figure 1).

Now that the fundamental relationship between a real system, a model, and a simulation have been defined, the implications of this relationship can be examined. As was already discussed, a simulation's ability to represent reality first hinges on the model's underlying ability to represent the real system. Therefore, the first step in determining a simulation's ability to represent reality is to examine the relationship between a real system and a model of that real system. To begin, it must be recognized that a real system is infinite in its input, how it processes the input, and its output, and that any model created must always be finite in nature given our finite abilities [26]. From this statement alone it can be seen that a model can never be as real as the actual system

Figure 1: Relationship between a System, a Theory/Model, and a Simulation



and that instead all that can be hoped for is that the model is at least capable of representing some smaller component of the real system [2]. As a result, it can be said that all models are invalid in the sense that they are not capable of representing reality completely.

The idea that all models are bad is certainly not a new idea. In fact, it is recognized by many people that this is true [2, 26, 74] and there are even articles written which discuss what can be done with some of these bad models to aid in our understanding and decision making [33]. However, if all models are bad at representing a real system and a model is only capable of representing a small portion of that real system, then how will it be known if a model actually represents what happens in the system? In essence, how can we prove that a model is valid at least in representing some subset of a real system?

The basic answer to this question is that a model can never be proven to be a valid representation of reality. This can be shown by examining several different perspectives. The first perspective can be explained using Gödel's Incompleteness Theorem [28]. In his theorem, Gödel showed that a

theory cannot be proven from the axioms upon which the theory was based. In essence, this means that because every model must be based upon some set of axioms about the real system, there is no way to prove that any model is correct [26]. Another perspective to consider is that there are an infinite number of possible models that can represent any system and it would therefore take an infinite amount of time to show that a particular model is the best representation of reality. Together these perspectives hearken back to one of the fundamental questions found in the philosophy of science: how can a model be trusted as representing reality?

Although it has been shown that a model cannot be proven to be a correct representation of reality, it does not mean that the second fundamental question of the philosophy of science (what methods and procedures make models believable?) has not been thoroughly explored. There actually exists many belief systems developed by famous philosophers that attempt to provide some perspective on this question [40, 42]. For instance, Karl Popper believed that a theory could only be disproved and never proved (Falsificationism), others believe that a model is true if it is an objectively correct reflection of factual observations (Empiricism) [63]. However, no matter what one believes to be the correct philosophy, the fundamental idea that remains is that all models are invalid and impossible to validate. A shining example of this idea can be seen by the fact that although both are considered geniuses, Einstein still showed that Newton's model of the world was wrong and therefore it is likely that eventually someone will come up with a new model that seems to fit in better with our current knowledge of reality [40]. Therefore, regardless of how correct a model may be believed to be, it is likely that there will always exist another model which is better.

The analysis from the previous paragraphs on the relationship between a real system and a model have led to the following conjectures about models:

- Models cannot represent an infinite reality and therefore all models are invalid with respect to a complete reality;
- Models can only hope to represent some aspect of reality and be less incomplete;
- There are infinitely many models that could represent some aspect of reality and therefore

no model can ever be proven to be the correct representation of any aspect of reality; and

- A better model than the current model is always likely to exist in the future.

From these conjectures, it appears that a simulation's capability to represent a real system is bleak based purely on the fact that a model is incapable of representing reality. However, as if things were not already appearing bad enough for simulation validity based on the model, there still exists another issue with trying to represent a model with a simulation. As seen graphically in Figure 1, another round of translation needs to occur before the transition from the system to the simulation is complete. At first glance, translating a model into a computer simulation would seem to be relatively straightforward. Unfortunately, this does not appear to be the case even when programming (verification) issues are left out of the equation. This conclusion generally arises from the limitations of the computer. For example, because computers are only capable of finite calculations, often times truncation errors may occur in the computer simulation when translating input into output via the model. Due to these truncation errors alone, widely different results can be obtained from a simulation of a model with slightly different levels of detail. In fact this result is often seen in chaotic systems such as Lorenz's famous weather simulations, which would later lead to the idea of the Butterfly Effect [27].

Suppose, however infeasible it may be, that advances in computers would make the issues of memory storage and truncation errors obsolete, then the next issue in a computer simulation's ability to represent a model is the computer's processing speed. Given that computer processing speeds are getting increasingly faster with time, the question about whether a computer can process the necessary information, no matter how large and detailed the model, within an acceptable time seems to be answered by just waiting until technology advances. Unfortunately, there is a conjecture which states that there is a speed limit of any data processing system. This processing speed limit, better known as Bremermann's Limit [2], is based upon Einstein's mass-energy relation and the Heisenberg Uncertainty Principle and conjectures that no data processing system whether artificial or living can process more than $2 * 10^{47}$ bits per second per gram of its mass [14]. From this conjecture, it can be seen that eventually computers will reach a processing limit and that models and the amount of digits

processed in a respectable amount of time will dwarf Bremermann's Limit. Consider for example how long it would take a computer approximately the size ($6 * 10^{27}$ grams) and age (10^{10} years) of the Earth operating at Bremermann's Limit to enumerate all of the approximately 10^{120} possible move sequences in chess [14] or prove the optimal solution to a 100 city traveling salesperson problem ($100!$ or approximately $9.33 * 10^{157}$ different routes). Given that this super efficient Earth sized computer would only be able to process approximately 10^{93} bits to date, it would approximately 10^{27} and $9.33 * 10^{64}$ times longer than the current age of the earth to enumerate all possible combinations for each problem respectively. From this it can be concluded that from the human perspective it would take entirely too long and be entirely too impractical to attempt to solve these problems using brute force.

As a result of this analysis, it can be seen that at the very least it is challenging for a computer, given it's memory and processing limitations, to accurately represent a model or provide accurate results in a practical amount of time. To combat this issue of computing limitations, simulationists will often build a simulation of a model that incorporates many assumptions, abstractions, distortions and non-realistic entities that are not in the model [46, 58, 74, 83, 84, 86]. Such examples include breaking continuous functions into discrete functions [46], introducing artificial entities to limit instabilities [46], and creating algorithms which pass information from one abstraction level to another [85]. From these examples, it can be clearly seen that the limitations of computing makes translating a model into a simulation very unlikely to result in a completely valid representation of that model. This is why often simulation building is considered much more of an art rather than a science, because getting a simulation to appear to represent a model in a computer requires a lot of tinkering that ends up with a simulation that only appears to represent the model. As a result of this discussion, it can be seen that not only are there an infinite number of models that can represent some aspect of reality, but there is probably also an infinite number of simulations that can represent some aspect of a model.

Given the above discussion, the following conjectures can be made about the ability of a computer simulation to represent a model:

- Computers are only capable of finite calculations and finite storage, therefore truncation errors and storage limitations may significantly impact the ability of a computer to represent a model;
- Computers can only process information at Bremermann's Limit, making it impossible for them to process large amounts of information about a model in a practical amount of time;
- To attempt to represent a model with a computer simulation either requires sacrificing accuracy to get results or sacrificing time to get better accuracy;
- Given the limitations of computing and the trade off between accuracy and speed, there are many ways to try and represent a model with a simulation; and
- Because there are many possible simulations that can represent an aspect of a model, it is impossible to have a completely valid and still useful simulation of a model.

The conjectures above show why translating a model into a computer simulation is not a straightforward process and that many times a simulationist is simply trying to tinker with a simulation until it happens to appear to have some representation of the model. This complexity with translating a model into a simulation only makes the question of a simulation's ability to represent reality more pressing.

As the section title suggests, the main goal of this section is to show why all simulations are invalid representations of reality. By examining the relationship between the real system, the model attempting to represent the system, and the simulation attempting to represent the model, the following summary conjectures can be made about simulation validity:

1. A real system is infinite.
2. A model cannot represent an infinite real system and can only hope to be one of any infinite possible representations of some aspect of that real system.
3. As a result of 1 and 2, a model is an invalid representation of the real system and cannot be proven to be a valid representation of some aspect of the real system.

4. There are many possible computer simulations that can represent a model and each computer simulation has trade offs between the accuracy of the results and time it takes to obtain those results.
5. As a result of 4, a simulation cannot be said to be a completely valid representation of a model.
6. Therefore, a computer simulation is an invalid representation of a complete real system and at the very best cannot be proven to be a valid representation of some aspect of a real system.

The above conjectures lay out the issues with a simulation's ability to represent reality. Furthermore, it can be seen why simulation validation continues to be a major issue. If simulations cannot be proven to be valid and are generally invalid representations of a complete real system, then what value to they serve? However, this question is not the primary source of research in simulation validation. Instead, much of the focus still remains on how one can validate a simulation. Given the conjecture that all simulations are invalid, or impossible to prove to be valid, what possibly could all of these articles mean when discussing simulation validation?

2.2 What Does Simulation Validation Really Mean in Practice?

Untill now, it has been shown that generally all simulations are invalid with respect to a real system, but there is still a fair amount of literature that continues to attempt to show how a simlation can be validated. It may initially appear that those practicing simulation building are unaware of the downfalls facing simulation's ability to represent reality, but this is not the case [9, 51]. So what are these articles and books discussing when they are focused on simulation validation? Insight into what practioner's really mean by simulation validation can be seen from their own definitions of validation:

“Validation is the process of determining whether a simulation model is an accurate representation of the system, for the particular objectives of the study” [24, 51]

“Model Validation is substainsiating that a model within its domain of applicability, behaves with satisfying accuracy consistent with the models and simulations objectives...”

[7, 68]

“Validation is concerned with building the right model. It is utilized to determine that a model is an accurate representation of the real system. Validation is usually achieved through the calibration of the model, an iterative process of comparing the model to actual system behavior and using the discrepancies between the two, and the insights gained, to improve the model. This process is repeated until the model accuracy is judged to be acceptable.” [9]

“Validation is the process of determining the manner in which and degree to which a model and its data is an accurate representation of the real world from the perspective of the intended uses of the model and the subjective confidence that should be placed on this assessment.” [17]

These definitions clearly indicate that in practice simulation validation takes on a somewhat subjective meaning. Instead of validation just being the process of determining the accuracy of a simulation to represent a real system, all of the above authors are forced to add the clause ‘with respect to some objectives’ because it has been shown why a simulation can never accurately and completely represent a real system. By adding this caveat, simulationists have inserted some hope that a model is capable of being classified as valid for a particular application, even though any model they create will only at best be a less than perfect representation of reality. With the insertion of this clause, the issue of validation takes on a completely new meaning. No longer is the issue of absolute validity the problem, the problem is now proving the relative validity of a simulation model with respect to some set of objectives.

In order to address this new problem, many articles have been published which provide a different perspective of this relative validity problem. One of these perspectives is to attempt to evaluate the relative validity of the simulation by treating it not as a representation of a model/theory but as a miniature scientific theory by itself and then use the principles from the philosophy of science to aid in proving/disproving its validity [42, 43]. As first introduced by Naylor and Finger in 1967 [60], many authors have since thoroughly examined the many beliefs that have emerged from the philosophy of science and have related them to simulation validity [10, 22, 41, 42, 63, 70]. Often these authors provide insightful views into simulation validation because the philosophy of science has been actively discussing the validity of theories long before the inception of simulation [42].

Although the introduction of scientific philosophy has certainly provided new perspectives and points of view on the subject of validity to simulationists [42, 70], it could be said that overall the philosophy of science has brought with it more questions than answers. There are several key reasons for this. The first is that for every belief system in the philosophy of science there are both advantages and disadvantages. For example, often simulation validation is favorably compared to Falsificationism because it states that a simulation can only be proved false and that in order to have a simulation be considered scientific it must first undergo scrutiny to attempt to prove that it is false [41]. However, under this belief system it is difficult to determine whether a model was rejected based on testing and hypothesis errors or whether the model is actually false [40]. Another reason of concern for using the philosophy of science is that, as was discussed earlier, it is impossible to prove that a model/theory is valid at all. Therefore, using the philosophy of science to aid in simulation validity is more applicable in providing insights into the fundamental philosophical questions stemming from validation as well as potential validation frameworks than in actually proving the validity of a simulation.

Besides this high-level look at the validation of simulation as miniature theories, another perspective considers methods and procedures which can aid simulationists in proving the relative validity of their simulation given the assumption that it can be proven. This assumption is by no means a radical one. It makes sense that if one defines the objective of the simulation to include the fact that it cannot completely represent the real system then it is possible for a simulation to meet the needs of a well defined objective and therefore have relative validity. With a goal in mind to find these methods and procedures, a plethora of techniques have been developed as well as when and how they should be applied within systematic frameworks to aid simulationists in validating their models [7, 68]. As another indication of how much this field has been explored, even the idea of validation itself has been reduced to many different types of validation such as replicative, predictive, structural, and operational validity [21, 88].

Clearly a lot of effort has been spent on summarizing and defining how one can go about validating a simulation given some objectives. It would be redundant to discuss all of them here in detail.

However, to better understand what simulation validation means in practice it is worthwhile to consider what all of these techniques and ideas have in common. Whether the validation technique is quantitative, pseudo-quantitative, or qualitative, each technique has its advantages, disadvantages, and is subjective to the evaluator. Although the statement that all of these techniques are subjective may seem surprising, it can actually be easily shown for all the validation techniques developed today. For example, when qualitatively/pseudo-quantitatively comparing the behavior of the simulation and the real system, it is clear that a different evaluator may have a different belief about whether the behaviors of the two systems are close enough such that the simulation could be considered valid. For a quantitative example consider when statistically comparing two systems and the evaluator is required to subjectively select a certain level of significance for the particular hypothesis test used. For those familiar with statistical hypothesis testing, it is clear that different levels of significance may result in different conclusions regarding the validity of the simulation. From these examples, it can be seen that even though many techniques have been developed, none of them can serve as a definitive method for validating a simulation. They are all subjective to the evaluator.

Given there is no technique which can prove that the relative validity of a simulation is true, the fundamental question of this section resurfaces: what does simulation validation really mean in practice? It has already been seen that in practice simulationists are not trying to validate that a simulation is a representation of a real system and instead are attempting to validate the simulation according to some objective, which also cannot be systematically proven to be true. Therefore, one must consider other alternatives when examining what is really occurring when a simulationist is trying to validate their model according to some objective. From the vast amount of techniques, guides, and systems proposed to validate a simulation, it can be conjectured that simulation validation in practice is really the process of persuading the evaluators to believe that the simulation is valid with respect to the objective. In other words, in practice whether a simulation is validated by the evaluators depends upon how well the simulationist can ‘sell’ the simulation’s validity by using the appropriate validation techniques that best appeals to the evaluator’s sense of

accuracy and correctness.

The idea that simulation validation in practice is really the process of selling the simulation to the evaluator may not appeal to scientists, engineers, and simulationists, but there is a fair amount of evidence which supports this conclusion. First of all, any simulation book or article focused on validation will frequently stress the importance of knowing the evaluator's expectations and getting the evaluator to buy into the credibility of the simulation [9], some will even go as far as to say explicitly that one must sell the simulation to the evaluator [51]. Others go on to say that validating a simulation is similar to getting a judicial court system to believe in the validity of the simulation [22]. Therefore, it can be concluded that generally those practicing simulation understand that validation is more about getting the evaluators to believe in the simulation's validity and less about getting a truly valid simulation, which has been shown to be impossible anyway. From this statement, an interesting insight can be made into the nature of simulation validation in practice that until recently has not been extensively covered or thought to be of much importance. This insight is that simulation validation is not completely removed from society and other social influences. In fact, it appears that simulation validation in practice requires the simulationist to have the ability to actively interact with the community of evaluators and attempt to persuade that community to accept the simulation as correct. As a result, some have argued that simulation validation in practice is similar to how any social group makes a decision [63].

In trying to determine what simulation validation really means in practice, several fundamental points have been made:

- In practice, a simulation is validated based on some objective and not on being a true representation of the real system;
- All of the techniques developed to prove the validity of a simulation in practice are subjective to the evaluator and therefore cannot systematically prove the relative validity of the simulation; and
- Validating a simulation in practice depends upon how well the simulationist sells the valid-

ity of the simulation by using the appropriate validation techniques that best appeals to the evaluator's sense of accuracy and correctness. Furthermore, this means that simulation validation in practice is susceptible to the social influences permeating the society within which the simulation exists.

Given the above conclusions, it can be seen that simulation validation in practice really seems to have little to do with actual validation, where validation is the process of ensuring that a simulation accurately represent reality. Instead, simulation validation in practice is more concerned with getting approval from evaluators and peers of a community relative to some overall objective for the simulation. In other words, simulation validation in practice is really the process of getting the simulation sanctioned [82], not validated. As a result, perhaps it is time for simulationists to consider adopting the term simulation sanctioning instead of simulation validation since sanctioning implies a better sense of what is actually occurring and validation implies that the truth is actually being conveyed in the simulation when it is not. However, it is unlikely that this transition will occur given the fact that simulation validation today is mainly concerned with getting evaluators to buy into the results of the simulation, the current paradigm in simulation has been established, and saying a simulation is valid sounds much better to a seller than does saying a simulation is sanctioned. This brings up an interesting dilemma for simulationists because if simulations cannot represent reality, then what good are they?

2.3 What Good are Simulations?

Since simulations in practice are sanctioned and not validated, the next logical question to ask is if simulations are incapable of representing reality and therefore are incapable of providing true results with respect to the system, then what good are simulations? To answer this question, it is first important to jump out of the world of logic and philosophy and see that practically speaking simulation would not be growing in popularity if it did not provide some demonstratable good to those using the technique. In fact, it could be said that the continuing widespread use of simulation in practice, the number of commercial simulation software packages, and the number of

academic publications meaning simulation are clear indications that simulation has provided enough robustness to be considered useful and indeed successful [48].

At first glance, the success of simulation in practice may appear to be a contradiction to the previous statement that all simulations are invalid. However, this is resolved by making clear that all simulations are invalid with respect to an absolute real system, which does not mean that a simulation is not capable at some abstraction level to get relatively ‘close’ to representing a small portion of an absolute real system. For example, a simulation of a manufacturing system may come very close to representing the outcome of a process, but nevertheless it is still invalid with respect to actual system because of all the reasons discussed earlier. This same conclusion can also be related to purely scientific theories, such as Newton’s laws of motion. Although they can produce reliable results in certain abstraction levels, overall they have been shown to not be completely correct representations of reality. Given the popularity of simulation it appears that in spite of the fact that they are not capable of representing an absolute real system, simulations are capable of getting close and robust enough results to become practically useful [44, 47]. Simulations are useful because they are capable of providing reliable results without needing to be a true representation of a real system [86]. However, this is often more true for some types of simulations and real systems than others.

In general, it can be said that the success of a simulation at obtaining reliable and predictable results is dependent upon how well the real system is understood and studied, because a well understood system will result in better underlying theories that form the foundation of the simulation. When a simulation is used to represent a well understood system, the reason for using the simulation is not solely to try to understand how the real system may operate but instead is to take advantage of the processing power and memory of the computer as a computational device. In this role, the simulation is likely to produce reliable results because it is simply being used to express the calculations that result from a well-established theory. A typical example of this can be found in a standard queuing simulation. Since queues have been extensively studied and well-established laws have been developed [38], it would be likely that a sanctioned simulation of a queue would

be capable of producing reliable results and predictions because the simulation's role is to simply be a calculator and data storage device. For these types of well understood systems, simulation can provide predictive power. However, as less is known about the system, the likelihood that the simulation will provide reliable data decreases to the point where the usefulness of the simulation takes on a new meaning.

As less is understood about the real system, a simulation takes on a new role of becoming a research instrument acting as a mediator between theory and the real system [58], because the theories about the real system are not developed enough to truly provide reliable predictions about future states of that system. One can think of a simulation in this case as being a research tool in the same sense that a microscope is a research tool [46]. While the microscope can provide insight into the real system, it does not directly reflect the nature of the real system and cannot directly provide reliable predictions about the real system. Instead the microscope is only capable of providing a two-dimensional image of a dynamic three-dimensional real system. However, the microscope is capable of mediating between existing theories about the real system and the real system itself. Experiments can be designed and hypotheses can be tested based on information gained from the microscope. In this same way, a mediating simulation is capable of providing insight into the real system and the theory on which it was built without being a completely valid representation of that real system. Although only formally recognized recently [58], the idea that computers can be used to facilitate experiments and mediate between reality and theory has existed for a relatively long time. In the early years of computing, John von Neumann and Stanislaw Ulam espoused the heuristic use of the computer, which is an alteration of the scientific method to replace real system experimentation with experiments performed within a computer [77].

An interesting aspect of mediating simulations, which is valuable to consider, is the apparent interplay between the simulation and the real system. Just as a microscope started off as a relatively crude research tool that only provided a limited view of the real system and with time was improved to provide continuously deeper understandings of the real system, so too can a simulation of a real system be improved to gain better insights into that real system [84]. Furthermore, as the

real system is better understood, so can the simulation of that real system be improved which in turn can allow new insights to be gained about the real system. Examples of this mediating role of simulation can be seen in many different fields. One such example can be found in the field of nanotechnology where without computer simulations to aid in the complex and difficult experiments, certain advances in nanotechnology would not be possible [39]. Another example can be found in any complex production system, where the simulation provides insights into how the real system might behave under different circumstances. In the world of Agent-Based Modeling, “toy models” such as ISAAC (Irreducible Semi-Autonomous Adaptive Combat) have been used as to explore and potentially exploit behavior that emerges in battlefield scenarios [37]. A final example can be seen in the field of physics where some “think of sciences as a stool with three legs - theory, simulation, and experimentation - each helping us to understand and interpret the others” [50].

It may seem odd to conduct an experiment using a computer simulation of a real system when the real system would seem to be the only way to guarantee that the conclusions obtained were impacted by the real system and not an error built into the simulation. However, since a real system is infinite in nature it makes error a natural aspect of any finite experiment conducted on that real system. For example, attempting to measure the impact different soils have on the growing rate of a plant will always have variation and error simply because the experimenter is trying to conduct a finite experiment and make finite measurements on an infinite system. Therefore, error will always be present in experiments, regardless of whether they are done on the real system or a simulation of the real system [84]. A huge difference is that simulation errors are largely repeatable giving the researcher greater control for potential insight into the real system. Nevertheless, this does not mean that the simulation should be not properly sanctioned prior to being used to facilitate an experiment. In order for a simulation to be a mediator in this sense, the simulation must contain some sort representation of the real system and be sanctioned by the evaluators, otherwise the results produced may not be reliable enough to provide any insight.

As the relationship between the real system, the theory, and the simulation begins to blur together into non-distinct parts due to the lack of knowledge about the real system, what is meant

by a simulation representing some part of reality also begins to become hazy. The more traditional belief is that for simulations to represent some aspect of a real system they should have at least some structural resemblance to the real system. However, even a simulation of a real system that is very well understood has been shown to contain many built-in assumptions and falsifications which do not match what is known about the structural aspects of the real system. This becomes especially true for simulations when not much is known about the real system because how can a simulation be a structural representation of a real system when the structure of the real system is not fully understood? This flaw of structural representation perhaps has led to one of the current paradigms in simulation where the performance of a simulation, i.e. how well the simulation translates realistic input into realistic output, and not accuracy is the fundamental benchmark in determining the usefulness of that simulation [44, 46]. Indeed, many of the technical validation techniques proposed today emphasize the use of this realistic output or black-box paradigm [6, 9, 51, 68].

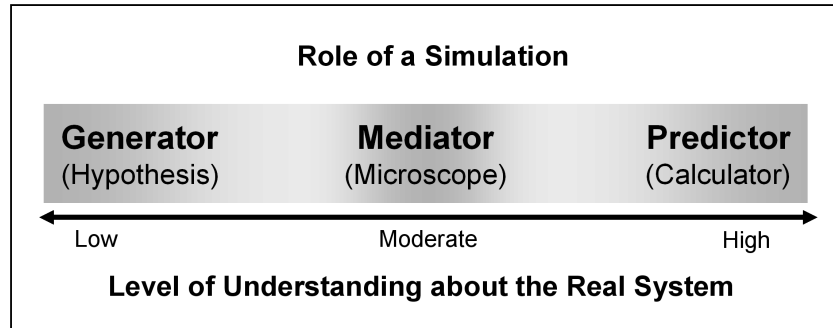
In general, this shift away from white-box evaluation (structural representation) and towards black-box evaluation (output is all that matters) [13] can lead to several interesting conclusions. The first is that this shift indicates the general acceptance of the idea found in Simon's *The Sciences of the Artificial*. Essentially, Simon argues that artificial systems (ones man-made such as simulations) are useful because it is not necessary to understand the complete inner workings of a real system due to the fact that there are many possible inner workings of an artificial system that could produce the same or similar results [73]. One way to think about this is to consider whether the differences between the inner workings of a digital clock and an analog clock really matter if they both provide the correct time. Clearly, someone interested in knowing the correct time would be able to gain the same amount of information from either clock even though both clocks are structurally very different. This fundamental aspect that different structures are capable of producing the same or similar results should not be a complete surprise; it has already been discussed that there are potentially an infinite number of possible models which can represent a single abstraction of a real system. Another conclusion that can be drawn by the shift towards black-box evaluation is that simulations are beginning to catch up and pass the theoretical understanding of the systems

that they are being built to represent. The question now becomes, what possible usefulness can a simulation be when little to nothing is known about the underlying principles of the real system of interest?

At first glance, the usefulness of a simulation for a system that is not well-understood appears to be nonexistent, because there is clearly nothing from which to build the simulation. But it is from this lack of underlying theory and understanding of the real system that the usefulness of this type of simulation becomes evident. Consider what a simulationist would encounter if asked to build a simulation of a poorly understood system. The first steps they would probably take, besides trying to understand the evaluators, would be to observe the system and, upon observing the system, the simulationist would then attempt to generate the same behavior observed from the real system within the simulation. This ability of a simulation to act as a medium in which new theories about a real system can be generated points to the third role of simulation, which is that of a theory or knowledge generator. Although certainly not a traditional means, using a simulation as a medium to generate new theories and ideas about the real system is no different in any way from using pencil and paper or simply developing mental simulations about the real system [2]. One could observe a system and attempt to test the implications of a theory by using pencil and paper or develop elaborate thought experiments as those made famous by Einstein just as easily as one could use a simulation to test whether a theory is capable of representing the real system. Examples of simulations being used for this role abound in the new simulation paradigm of Agent-Based Modeling (ABM), where simulationists are typically trying to understand problems that are difficult for us to grasp due to the large amount of dispersed information and the high number of interactions that occur in these systems (more about the special case that ABM simulations present to the world of simulation will be discussed in detail in the next section) [20, 56, 61].

There are several clear advantages for using simulations as generators, the most important of which is the ability of a simulation to create ‘dirty’ theories of systems where the simplicity of the real system eludes our grasp. Typically, scientific theories are often idealized for a particular case and do not allow for much deviations from these idealizations. It could be thought that

Figure 2: Different Roles of a Simulation



these idealizations are partly the result or desire humans have to make simplifications and elegant equations to represent the complex world we live in. However, simulations allow a theorist to build a representation of a system within a simulation that is capable of having many non-elegant aspects, such as ad-hoc tinkering, engineering, and the addition of logic controls such as if-then statements [47, 85]. As a result of this flexibility, it could be predicted that as more problems venture into the realm of Organized Complexity (medium number of variables that organize into a highly interrelated organic whole) [79] that the use of simulation to generate new ‘dirty’ theories about the real system will be needed because these systems are irreducible and typically hard to understand to the point that often simulationists are surprised about the results obtained from these systems [15].

By using simulations as theory generators, simulationists can attempt to generate a potential theory that explains the phenomena observed in the real system, just as a more traditional scientist would do but without the simulation medium. Therefore, there are no apparent implications of using a simulation as a generator to the philosophy of science [25]. As a result, those using simulations in this manner should perhaps not consider themselves as disconnected from science because they are an engineer or computer programmer by trade, but perhaps should attempt to ascribe to the practices, rigor, and roles taken on by scientists to make true progress in the practitioner’s field of interest. Furthermore, as simulationists and scientists continue to push the limits of simulations beyond that of the current knowledge of some system of interest, it can be seen why some researchers are considering simulation to be the epistemological engine of our time [36].

Despite the inability of a simulation to completely represent some abstraction of reality, simulation has still proven to be useful in several different ways. By making the connection between the level of understanding known about the real system and the simulation, a clearer picture can be rendered about what simulation is capable of as well as where it fits into today's scientific and engineering endeavors. This continuous relationship, as shown in Figure 2, shows that when much is known about the system, the simulation tends to take on more of a predictor role, similar to that of a calculator. As less is known about the real system, the simulation begins to take on the role of being a mediator between the system and the theory, much like the role a microscope plays a mediator between microscopic systems and the understanding of those systems. Finally, as the understanding of the system approaches almost nothing, the simulation can act as a generator of potential theories about the nature of the system. Although some of the implications of using simulations as generators have been discussed, it would be valuable to further breakdown the use of simulations as generators in order to better understand their relation to other simulations, the limitations they present, and to evaluate what might be lacking due the fact that the use of simulation in this manner is relatively new. Since ABM directly fits into this category, the deeper issues involved with generator simulations will be discussed in the next section from the ABM perspective.

2.4 What Good is ABM?

Despite the fact that any simulation paradigm could potentially be used as a generator, probably the most popular one used today is ABM. Emerging from Cellular Automata, Cybernetics, Chaos, Complexity, and Complex Adaptive Systems, ABM is used today to understand and explore complex, nonlinear systems where typically independent and autonomous entities interact together to form a new emergent whole <insert reference to previous article if needed>. An example of such a system can be observed by the flocking behavior of birds [52]. Although each bird is independent, somehow they interact together to form a flock, and seemingly without any leading entity, manage to stay in tight formations. With this in mind, simulationists using ABM attempt to discover the rules embedded in these individual entities that could lead to the emergent behavior and eventually attempt to make inferences about future states of these systems based on the simulations they de-

veloped. It can be clearly seen that ABM is indeed used as generators of hypotheses for these type of complex systems.

As may be expected, theories generated by ABM have many of the same problems encountered by more traditional methods of generating scientific theories. As discussed earlier, such fundamental problems include whether the truth of a system be known and what methods can be used to sanction these theories. Although the issue of truth will always remain for any scientific theory, when it comes to the methods needed to sanction ABM it appears that several new problems begin, at least in terms of simulation, to emerge. These problems typically come from the fact that currently ABM is used to investigate problems where no micro level theory exists (it is not known how the individual entities operate) and where it is often very difficult to measure and collect macro level data (the emergent behavior) from a real system and compare it to the data generated from the simulation [8, 52, 56, 61]. Ultimately, this current characteristic of these complex problems means that the current traditional and accepted quantitative sanctioning techniques which promote risk avoidance based on performance and comparing outputs are not applicable [71], since too little is known about these systems. From this statement, several interesting conclusions can be made about ABM and generator simulations.

The first is that because ABM is relatively new as a paradigm, either accepted techniques to sanction these simulations have not yet been created to match the current sanctioning paradigm or a new sanctioning paradigm with new sanctioning techniques is needed for generator simulations. In order for the first statement to be the case, the underlying theory behind the real system being studied by these generator simulations would need to be known to the point that the simulation would no longer be a generator but instead be a predictor; the current sanctioning paradigm has a majority of its interest is predictability and in turn has created sanctioning techniques that are mainly focused on this paradigm. Therefore, it is impossible for generator and ABM simulations by their nature to fit into the current predictive sanctioning paradigm. Furthermore, if a ABM simulation ever became predictable it would no longer be a generator simulation and traditional quantitative sanctioning techniques could be used. As a result, simulationists using ABM today as

a generator should not be focused on meeting current predictive sanctioning paradigms, but should shift their focus to creating a new generator sanctioning paradigm and the development of new techniques to match. Attempting to create this new sanctioning paradigm will certainly not be an easy task, but it is absolutely necessary if ABM simulations as generators are to become acceptable for what they are: generators of hypotheses about complex systems. Only after this paradigm has been created can both simulationists and evaluators come to a firm conclusion about whether a generator simulation should be sanctioned as a scientific research tool or an engineering alternative analysis tool.

Although the overall lack of micro level theory for these types of problems may push the current limits of today's simulation sanctioning for ABM, it should by no means be seen as a completely new problem. For almost every simulation the simulationist must make some assumption or build some theory about how the system works in order for the overall simulation to function appropriately. What ABM and generator simulations really do is take this engineering and ad-hoc notion to the extreme. As a result, it can be seen why that this departure from the engineering 'norm' in the world of simulation could produce a fair amount of skeptics. In fact, even von Neumann was skeptical about using a computer in this manner, even though he recognized the practical usefulness of this approach [47]. However, this type of skepticism is to be expected whenever pushing the limits of any paradigm accepted by a community [63]. Overall, what is needed to gain acceptance of generator simulations and a new sanctioning paradigm will be time and compelling evidence that eventually these ad-hoc simulations will move up the continuum of understanding about the real system such that they become mediators and then predictors.

Until the complex systems simulated by ABM are well understood, ABM simulations should be viewed practically for what they provide as strictly generator simulations. This means that ABM simulation should be viewed currently as a research tool, which is not only capable of providing insight into the real system but also points to what needs to be understood about the real system in order for a theory to be developed that can predict some aspect of the real system [8]. In order to know that the knowledge gained from ABM simulations is reliable, a new sanctioning paradigm is

needed that is not based on predictability because it is impossible for the systems being studied with a generator simulation to be strictly predictable. Instead, this new sanctioning paradigm should be based on precision and understanding as it relates to the more traditional methods employed by scientists [71]. Furthermore, as this new sanctioning paradigm expands, new sanctioning techniques can be created which provide value to the generator simulationist such that the real system is understood to the point that generator simulation paradigms such as ABM can become mediator or predictor simulations.

2.5 Section Concluding Remarks

As simulation continues to grow in popularity in scientific and engineering communities, it is invaluable to reflect upon the theories and issues that serve as the foundation for simulation as it is known today. With this in mind, this article attempted to add context and reconcile the practices of simulation with the theory of simulation. In particular, this article took fundamental philosophy of science issues related to simulation sanctioning and built a framework describing the crucial relationships that exist between simulation as a medium and real systems. The first relationship discussed is a simulation's inability to represent a complete abstraction of that real system. As a result, all simulations are invalid with respect to a complete real system. From this conclusion, the current practice of simulation validation was investigated to gain insights into what simulation validation really means. Despite the attempts of simulationists today, simulation validation really boils down to selling the simulation correctness to the evaluators of the simulation, since it is impossible to prove that a simulation is valid. With this in mind, it has been suggested that simulations are not really validated in practice but are instead sanctioned. In turn, this inability of a simulation to be validated brings into question the usefulness of simulations in general.

However, a simulation does not need to be a complete representation of some aspect of a real system to be useful. Therefore a general framework was developed that related the role of simulation based upon the level of understanding of the real system of interest. In this continuous framework, a simulation can take on the role of generator, mediator, or predictor as the level of understanding increases with regards to the real system. From this framework a clear set of expectations for

a simulation can be distinguished based upon the level of understanding about the real system. Furthermore, it is hoped that this framework can further provide a base onto which new techniques and perceptions of a simulation's usefulness can be developed. Ultimately, this relationship shows partly why today simulation is becoming the research and knowledge generating tool of choice. The epitome of this new use of simulation as a generator and research tool has emerged in the form of ABM, because ABM aids in the understanding of complex, nonlinear systems. However, because ABM is relatively new to simulation, the current practices developed to sanction simulations that are focused on predictive performances are not applicable. Therefore, simulationists using ABM should develop a new sanctioning paradigm for generator simulations that is focused on understanding and accuracy. With a new sanctioning paradigm, evaluators and simulationists can have better expectations of ABM and other generator simulations as a research tools that point to what is needed and what is possible. In the context of this framework and that as ABM improves the understanding of complex systems, perhaps eventually the level of understanding will increase concerning the real system and in turn allow ABM to take on the role of mediator or predictor.

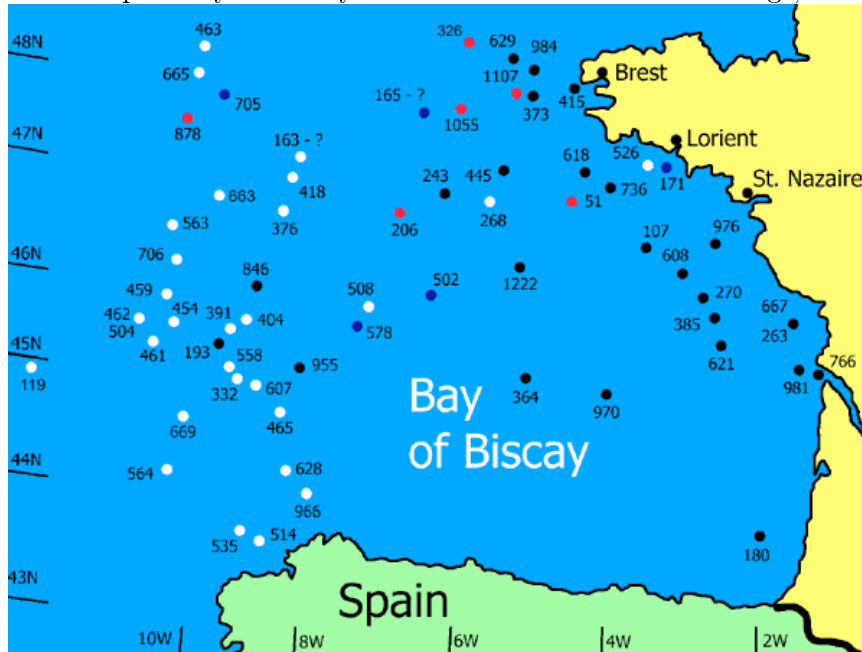
3 Case Study and Developmental Methodology

During World War II (WWII), the Germans were the first to effectively use the submarine against non-military targets, specifically against the logistical forces supporting the Allied war effort. U-boats (from the German word for submarine, *Unterseeboote*) were, in fact, used primarily to sink Allied merchant ships crossing the Atlantic Ocean to re-supply the Allied forces in Europe.

From 1941 through 1944, U-boats operated out of captured ports on the western coast of France. From these ports, the U-boats transited the Bay of Biscay to the Atlantic where they hunted the convoy targets. The Bay of Biscay is bordered on the east by France, to the south by Spain and Portugal, and in the north by Great Britain and Ireland (Figure 3, <http://uboat.net/maps/biscay.htm> accessed June 2008). In departing from and returning to their ports in France, the Bay provided the only access route to the Atlantic.

The Allies concentrated their search efforts in the Bay in offensive endeavor to counter the U-

Figure 3: Map of Bay of Biscay with Locations of U-boat sinkings, 1942-1944



boat threat. The Bay, with a few exceptions, was the only feasible area to conduct U-boat hunting operations. The open waters of the Atlantic were simply too vast providing ample area for U-boats to navigate and hide. The German occupied ports in France were heavily defended and hardened against bomber attacks. Additionally, German fighter aircraft patrolled the skies over and around the ports, deterring direct attacks against the ports or attacks against U-boats in the coastal waters near France.

There were two exceptions to this search and counter policy: near the merchant convoys and at U-boat re-supply points in the Atlantic. Escorting merchant convoys could only be accomplished close to Britain, within the range of the Allied aircraft. This was not as effective as searching the Bay because U-boat captains generally engaged in attacking the convoy ships in the Atlantic before this escort point. The decryption of the German radio code, Enigma, by Allied intelligence allowed aircraft to locate and patrol several U-boat resupply points in the Atlantic. but this was not generally exploited to avoid leaking Allied knowledge of the Enigma decryption.

Although it was the primary feasible search area for the Allies, the Bay of Biscay still contained roughly 130,000 square miles of potential search area. However, due to transit requirements, the

U-boat density in the Bay was higher than in any other potential search area. Also, submersible technology at the time did not enable submarines to stay underwater indefinitely. U-boat engines ran on two sources of power, diesel engines and batteries. Batteries only enabled the U-boat to travel approximately 100 nautical miles underwater before it was forced to surface for nearly three hours to recharge the batteries while operating under diesel power. As a consequence, U-boats were forced to surface during their transit of the Bay. A surfaced U-boat was much easier to locate than a submerged U-boat.

Between 1941 and 1944, Allied air forces vigorously patrolled the Bay. Early on, the finding and “killing” of a U-boat was an infrequent event; however, with the development and implementation of new technologies such as radar, the Allied search forces became more adept at finding and sinking U-boats.

3.1 Campaign Issues

So what is the importance of the Bay of Biscay campaign? This conflict between the Allied search aircraft and the avoiding German U-boats is rich with strategic and tactical decisions on both sides, driven predominantly by the state of technology and its progression. Additionally, the field of operations research was born out of the Allied attempt at modeling the conflict and optimizing their search strategies. This campaign has been described as a technology “see-saw” or “tug-of-war” [54]. Until the Germans began outfitting the U-boats with snorkels, allowing them to stay underwater indefinitely, the technological focus was on the Allied use of radar for searching and the radar countermeasures employed by the U-boats.

The result of technology battles forced larger tactical and strategic issues on both sides. Since U-boats could not transit the entire Bay submerged, their issue was when, and for how long, to surface. U-boats traveled faster on the surface than underwater (average of 10 knots versus average of 2.5 knots), and less time spent in the Bay meant more time to operate in the Atlantic. Increased surface exposure, however, increased the risk of being detected and destroyed by Allied aircraft. Traveling underwater reduced their vulnerability, but greatly increased the transit time. While surfacing at night was usually safer, a strategy of only surfacing at night would allow Allied search

efforts to concentrate solely at night. Finally, the use of countermeasures in the form of search receivers could give U-boats advance warning of approaching aircraft. The downside was that a side effect from the use of these primitive search receivers was the broadcast of radio waves that Allied aircraft could detect and track.

The Allied search forces were faced with similar issues: the time of day to search, where to search, and how often to search a given area with a limited set of resources. There was a vast difference in the effectiveness of day searching versus night searching. At one point in the campaign, it was calculated that the probability of killing a sighted U-boat during the day was approximately 40% while at night it was merely 11% [78]. If Allied search effort were restricted to day searches, when Allied aircraft effectiveness was higher, the U-boats could capitalize by surfacing only at night. Such were the game theoretic issues facing Allied decision makers during the war, and in particular this aspect of the war. Such issues makes the Bay of Biscay quite amenable to ABM-based analyses.

3.2 Previous Analyses

The first analysis of the Bay of Biscay campaign was written in 1946 shortly after WWII, but not cleared for publication until 1973. Waddington [78] details the efforts of the Operational Research Section (O.R.S.) of the Royal Air Force Coastal Command in countering the U-boat threat. He describes how the O.R.S. engaged a wide variety of topics, from radar to navigation issues to weather, altitude and attack methods, with a goal of demonstrating that scientific methods of analysis might give useful assistance to effective executive action [78].

McCue [54], re-examines the analyses that were accomplished during WWII, and in some cases completing them with modern techniques. He addresses a wide range of issues from the technology seen-saw, seasonal impact, and the concept of a balanced search effort. McCue's work shows that one can quantify and analyze the campaign against the U-boats not merely by applying...index numbers and coefficients based on sound military judgment but through mathematical reasoning systematically applied to knowable physical quantities [54].

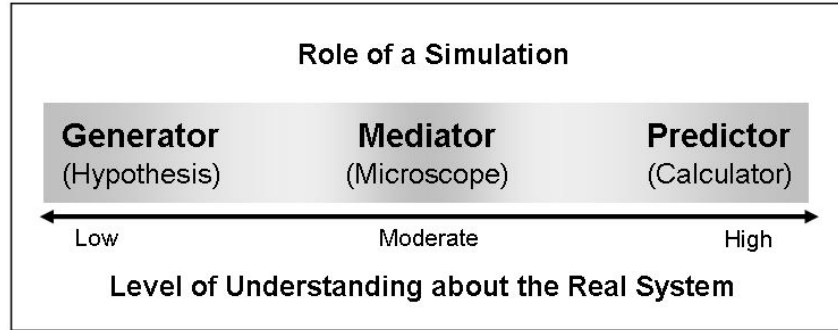
3.3 So Why This Scenario?

The Bay of Biscay campaign is useful as an agent-based model for several reasons. First, the amount of information available on the subject is immense. Besides the firsthand account analysis of Waddington and the later analysis of McCue, other sources, including Grand Admiral of Submarines Karl Donitz's War Diary, are accessible. Donitz's source is valuable as it gives insight into German U-boat strategies and tactics utilized in the Bay, as well as numbers for comparison. This availability of information lends itself to the development of a detailed model, and is also useful for the comparison of model results with historical results. Second, the technological developments, and the tactical and strategic decisions necessitated by this campaign raise a lot of "what if?" questions. These questions can be investigated by such a model with tradeoff analysis techniques. And finally, the model of a historical campaign like this, which is essentially a type of predator/prey model, will easily transition to a variety of present-day scenarios for investigation. Such scenarios include immigration, drug-running, smuggling, and terrorism, as well as some more traditional military operations.

4 Defining Some ABM Sanctioning Requirements

In the previous report, a high-level perspective was taken to better understand what simulation validation really meant and what implications it has on simulation and in particular Agent-Based Modeling (ABM) today. By examining the philosophy of science and current simulation practice we conjectured that simulations cannot be validated with respect to a real system and therefore simulations can only truly be validated subjectively by evaluators, a process we called sanctioning. With this conclusion, the value of simulations were examined and a framework was developed that described the usefulness of a simulation based upon the level of understanding of the real system that the simulation is intended to mimic (see Figure 4). Based upon this framework, we concluded an ABM mainly falls within the Generator Simulation category since they are typically used to model complex systems that are currently not well understood. Furthermore, we concluded that new sanctioning techniques are needed specifically geared towards ABM and other Generator Simulations

Figure 4: Different Roles of a Simulation



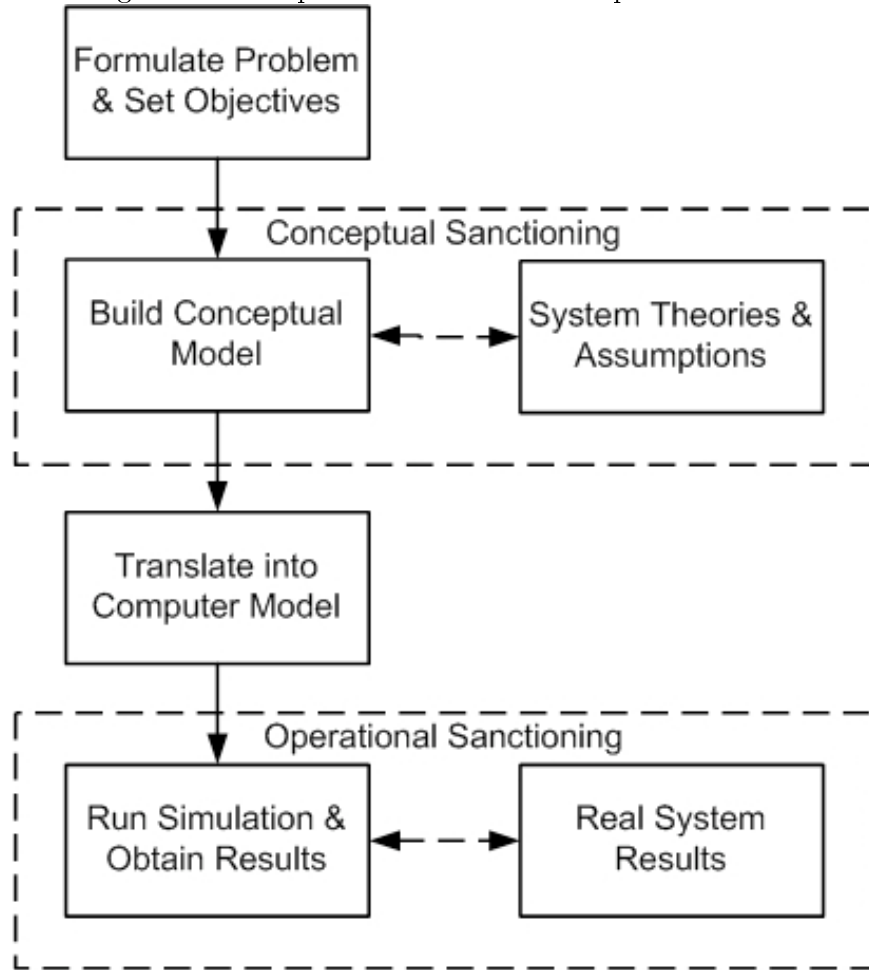
because current sanctioning paradigms do not match well with the abilities and usefulness of these new types of simulations. It was further suggested that new sanctioning techniques for ABM and other Generator Simulations need to emphasize understanding and accuracy as opposed to the common sanctioning techniques today that emphasize performance of the ABM.

Although analyzing simulation validation from a very high-level has yielded, among other things, the call for sanctioning techniques for ABM that emphasize understanding and accuracy, this high-level analysis does not provide detail when it comes to actually developing a new technique for sanctioning ABM simulations. Therefore, we explored how simulation models are developed and how they incorporate sanctioning techniques within the development process to gain insight into more specific needs for a new sanctioning technique that emphasizes both understanding and accuracy.

4.1 How Models are Developed

We first conducted a literature survey on how researchers build simulation models. In general, there exists a wide variety of step-by-step instructions, and associated figures, that each individual author proposes as a guide for the simulationist to create a good simulation. Some guides are fairly linear in nature with iterative steps to ensure that the model meets the objectives of the project and is properly sanctioned [9, 51]. Other guides are more complicated in structure because they emphasize the need for continuous sanctioning and they attempt to convey the complex task that is required if one wishes to build a good simulation model [7, 68]. Even though all of these simulation development processes differ, they contain similar fundamental elements of simulation building,

Figure 5: A Simplified Simulation Development Process



elements which can provide further insight into what the entire process of sanctioning Agent-Based Models really entails. Based upon these similarities, a simplified simulation development process is shown in Figure 5. This simplified process emphasizes the role that sanctioning (both conceptual and operational) plays in simulation building.

The first step, as shown in Figure 5, is to formulate the problem and set the objectives to be achieved by the simulation. In this step, arguably the most important step, the overall idea is to determine that the simulation paradigm is a good fit for the problem, determine the proper abstraction level for the simulation, and clearly define the expectations of the simulation project. The second step is to build the conceptual model. There is currently no clear and concise definition of what exactly a conceptual model is [66]. A conceptual model can be described as “the process

of abstracting a model from a real or proposed system” [66] and it is typically a mathematical, logical, and/or verbal representation of the real system of interest [68]. A conceptual model is the abstracted model intended to mimic the desired behaviors of the real system. This is often referred to as the “art” portion of the model building process [9, 57]. This second step relies heavily upon the known system theories driving the model development and the variety of assumptions required in the model abstraction process. Sanctioning in this step is referred to as Conceptual Sanctioning.

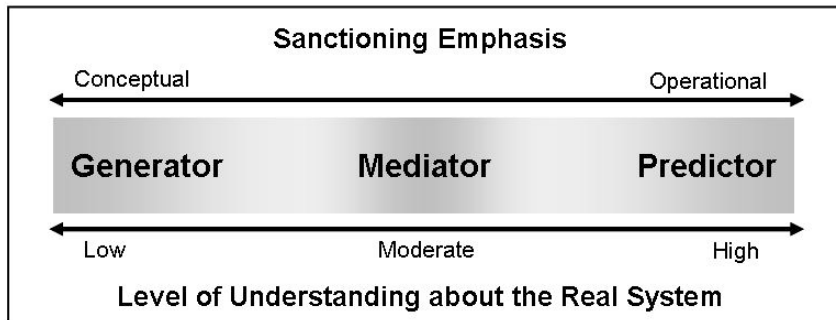
Upon successfully sanctioning the conceptual model, the next part of building a simulation is to translate the conceptual model into a computerized model. This is where verification issues come into consideration [7, 9, 51, 68]. Once the computerized model is verified, the final step of this process is to run the simulation and obtain results which can then be used to gain insights into the real system. However, before the simulation can reasonably be used as a proxy of the real system, it must first undergo another round of sanctioning we call Operational Sanctioning. For the simulation to be operationally sanctioned, its output behavior must sufficiently match the real system’s output behavior at the desired abstraction level and intended purpose [68]. Although this simulation development process appears simple, it highlights the major steps involved in building a good simulation.

4.2 Sanctioning Emphasis

Several key aspects about simulation building with respect to sanctioning arise by examining our simulation development process. The first is that there are really two types of sanctioning processes that occur during the building of any simulation. The second is that the two sanctioning processes occur at very different points in the simulation development process and as a result have very different objectives. While Conceptual Sanctioning occurs at the beginning of development process, and is concerned with how well the conceptual model matches the theory and assumptions of how the real system operates, Operational Sanctioning occurs at the end of the simulation building process and is concerned with how well the output of the simulation matches the output of the real system.

We compare Conceptual versus Operational Sanctioning with white-box versus black-box evalu-

Figure 6: Relationship between System Understanding and Simulation Sanctioning Emphasis



ation, respectively. Both Conceptual Sanctioning and white-box evaluation place more emphasis on understanding the details of how the system works while both Operational Sanctioning and black-box evaluation place more emphasis on matching results (performance) rather than the internal structure of the system. This comparison highlights where Conceptual and Operational Sanctioning fit into our simulation framework. If the major emphasis of a simulation study is on performance (Operational Sanctioning), and not on understanding the theories of the system, then we assume that the simulation is based upon a well understood system (otherwise the simulation study would have failed the Conceptual Sanctioning phase). Conversely, if the emphasis of a simulation study is on understanding the system (Conceptual Sanctioning) and not on how well the outputs match reality, then we assume the simulation is based upon a less understood system and it is unlikely that a simulation would be sanctioned based purely on performance when the entire simulation is built upon a soft and assumption-laden conceptual model (this has been called the Base of Sand Problem [18] in a military context). Based on this analysis, Figure 4 can be modified to relate the appropriate emphasis of sanctioning of a simulation based upon how much is understood about the real system. This modified framework is shown in Figure 6.

Although we assume the relationship between the level of understanding about the system and the appropriate sanctioning emphasis, we must clarify an important point before proceeding to the implications that this relationship has on developing a new sanctioning technique for the ABM paradigm. Upon examining Figure 6, one may conclude that if little is understood about the system then only Conceptual Sanctioning is required or if the system is highly understood

then only Operational Sanctioning is required. However, this is not the case. Regardless of the level of system understanding, any simulation should undergo both types of sanctioning to provide reasonable confidence in ABM results. Figure 6 helps to indicate which sanctioning type is most crucial in the simulation development process.

To re-enforce this idea, consider the emphasis within a typical scientific article. Even though the literature review will always be a crucial part of any article, the time spent critiquing and testing past work depends upon how well the foundational issues concerning the system of interest are generally understood. It is unlikely that a physicist using Newton's Laws for an experiment will spend much time on the conceptual validity of Newton's Laws if they are already accepted as applicable for the system, especially if the experimental results contributes to the field. As Hooker states [35], and as extended to our case, emphasis should be to the degree dictated by the critically of the ABM component. We next take this idea and examine the implications it has on developing requirements for a new sanctioning technique for ABM.

The Figure 6 model implies that simulations of poorly understood systems require more Conceptual Sanctioning emphasis than they would require Operational Sanction emphasis. Furthermore, it is our conjecture that most systems being simulated using the ABM paradigm are also not well understood. Therefore, it appears that the natural sanctioning emphasis for the ABM community is Conceptual Sanctioning. This further supports the earlier conclusion that Generator Simulations need sanctioning techniques that emphasize understanding and accuracy. The next logical step in the pursuit of a new sanctioning technique for ABM is to further explore the process of conceptual modeling and what Conceptual Sanctioning techniques currently exist.

4.3 Conceptual Modeling

Despite the fact that conceptual modeling is probably one of the most important aspects of building an effective simulation, there is actually little literature that specifically address how to build a conceptual model [67, 66], particularly for ABM. There seems to be two main reasons for this. The first reason is that the building of a conceptual model guidelines come in literature that discusses how to generally build a model; a conceptual model is traditionally an assumed part of the model

building process. While the actual model is the representation of a system’s behavior, a conceptual model defines what is going to be modeled and how it is going to be modeled. Therefore, although there is often no direct discussion of conceptual modeling, there is a fair amount of literature that discusses conceptual modeling, or at least the essence of conceptual modeling, as part of how one should build a model [67]. However, these model building articles also touch on the second point of why there are not many of articles discussing conceptual modeling; conceptual modeling is more of an art than a science [9, 57, 67, 66]. Conceptual modeling cannot be detailed into a step-by-step process that guarantees some particular result. Instead, all that can be offered to those attempting to build a conceptual model are general guidelines, such as keeping things simple and creating analogies to other developed structures [57]. Although more could be said about conceptual modeling, the fundamental conclusions that can be drawn is that “conceptualizing a model requires system knowledge, engineering judgment, and model-building tools” [64].

Examining the process of conceptual modeling reveals several important considerations for a new ABM sanctioning technique: needing to understand and have system knowledge, engineering judgment, and access to model-building tools. The next step is examining existing Conceptual Sanctioning techniques.

In a 2005 article, Sargent identifies two focuses that together encompass the idea of Conceptual Sanctioning [68]. The first part is ensuring that “the theories and assumptions underlying the conceptual model are correct” by using mathematical analyzes and statistical methods as well as ensuring that the theories are properly applied [68]. In other words, Sargent suggests using empirical sanctioning techniques to ensure that all assumptions and theories of the conceptual model match that of the real system. Thus, there are certain aspects of any conceptual model that are quantitative in nature and therefore the more traditional Operational Sanctioning techniques focused on quantitative sanctioning mentioned earlier can be used.

The next part of Conceptual Sanctioning is ensuring that “the model’s representation of the problem entity and the model’s structure, logic, and mathematical and causal relationships are ‘reasonable for the intended purpose of the model’” which are primarily evaluated using face vali-

ation and program traces [68]. Face validation and program traces involve subject matter experts examine all of the logic of the conceptual model, typically via some flowchart or other graphical device, in order to sanction the conceptual model. This means there is a qualitative aspect of Conceptual Sanctioning that requires an expert to subjectively review the structure and logic of the conceptual model.

To summarize, several key ideas regarding Conceptual Sanctioning and conceptual modeling have been identified. First, Conceptual Sanctioning should be emphasized when little is understood about the system. Second, conceptual modeling is not a straightforward process but requires system knowledge, engineering judgment, and model-building tools. Finally, Conceptual Sanctioning involves both quantitative evaluation and a qualitative evaluation, with each of these types of evaluations having long-standing sanctioning techniques. An immediate question to consider is why another technique or methodology is needed to sanction ABM when clearly one could use pre-existing tools to sanction their conceptual model? To answer this question, we next explore the kinds of systems the ABM paradigm are used to understand.

4.4 Systems for which the ABM Paradigm is Useful

From our previous work detailing the history and development of ABM, we found that the ABM paradigm is a relatively new simulation paradigm that emerged out of need for a tool to aid in understanding Organized Complexity Problems, or problems with a medium number of highly interrelated variables causing the system to be highly nonlinear [79]. With these new types of problems in mind, consider the following general conditions that make modeling a system easier and in turn make developing a conceptual model easier [64]:

- Physical laws are available that pertain to the system;
- A pictorial or graphical representation can be made of the system; and
- The uncertainty in system inputs, components, and outputs is quantifiable.

Since Organized Complexity Problems are just now beginning to be explored in detail, we can see why the ABM simulation paradigm needs a new sanctioning technique or methodology; every one

of the above conditions does not currently hold universally. Although general progress is being made in defining laws and theories governing Organized Complex Systems (such as found in the fields of Chaos, Cybernetics, and Complexity), these types of systems are not yet so well understood that there are solid physical laws available from which to build a model. Since this is a new type of problem, there are not many pictorial or graphical representations one can use to represent a Organized Complex System. Attempting to use traditional two dimensional graphs with arcs and nodes to represent nonlinear, complex, and highly interrelated states can get cumbersome, even infeasible, and too often increases confusion and complexity, the opposite intention of having the graphical representation [29, 30]. Attempting to quantify the uncertainty of inputs, components, and outputs can be a challenging task simply because Organized Complex Systems are not well understood, which is partly why the ABM simulation paradigm is being used in the first place. Perhaps the real problem with ABM sanctioning is not with sanctioning, or the lack of sanctioning techniques, but with the building of the conceptual model. Not having the tools or methods build a conceptual model of these systems makes sanctioning of the entire simulation extremely difficult and can immediately bring into question how well the simulationist understands the system being modeled.

4.5 Summary

What is needed to improve ABM sanctioning is a holistic methodology that incorporates the needs of building a conceptual model of complex systems with the needs to conceptually sanction the models. A methodology or tool is needed for ABM that aids in learning and conveying system knowledge, incorporates proper engineering judgment, aids in translating the conceptual model into a computerized model, easily displays theories and assumptions built into the model for quantitative evaluation with the real system, and can easily display the conceptual model's logic and structure in a graphical manner to subject matter experts of the system and not necessarily experts of ABM or simulation for qualitative evaluation. We next evaluate current ABM sanctioning techniques as well as other methods that aid in understanding complex systems to develop a basic methodology or tool that can meet those requirements.

5 General Observations of Current ABM Sanctioning Concepts

Once the requirements for a new ABM sanctioning methodology have been defined, it is important to review current sanctioning techniques of these types of simulations to see where this new methodology fits into the current practice. In general, there are two main sanctioning focuses for those currently building ABM simulations: model fitting and model testing (this terminology and fundamental concept is borrowed from [75], however there is no direct correlation with Stasser's definitions and the ones used in this report). For both of these focuses the ultimate goal is to have a simulation that appropriately mimics the real system macro behavior because matching the macro behavior indicates that the simulation *could be* operationally effective, however each focus has different ways to achieve this. In the model fitting focus, the parameters and theories that compose the micro-level of the model are 'optimized' via some algorithm. The optimization is *not* based on observations from the real system. In essence, the micro-level portion of the model is systematically changed until the macro-level results are achieved. Conversely, in the model testing focus, the parameters and theories that compose the micro-level of the model are based on observations and experiments performed on the real system.

Even though these focuses are at opposite ends of the spectrum, and certainly hybrids of these focuses exist, each extreme focus addresses the fundamental problem with ABM today; not much is understood about the real system. Model testing directly attacks the lack of knowledge by using the more traditional scientific method. Model fitting synthetically generates a feasible model to produce the macro-level behavior. An advantage of using model fitting is potentially obtaining novel micro-level theories about how the real world operates. However, an infinite number of models could represent a real system and thus it is impossible to prove that any model is an accurate representation of the real system; in fact one can only disprove that a model does not represent the real system of interest. Therefore, even if the model fitting focus results in a good representation of the macro-level system behavior it does not mean that the model accurately reflects the real system. Instead, the simulation is only a proposed theory that needs to be thoroughly tested to determine if it is feasible in reality, which means that model testing will still be required.

If the simulation exhibits the intended macro level behavior then what does it matter if the micro level behavior is correct? There are several examples that highlight why strict model fitting without empirical evidence can be problematic. One example comes from professional car racing where prior to the race crews can make adjustments to their car to attempt to optimize the car's performance for that particular track. After considering the layout of the track, road conditions, and after making several trial runs, suppose the crew outfits their car such that maximum performance is achieved during the race and the car ends up winning. As a result, for the next race that the crew decides to use the same car adjustments because they would expect to see the same performance. However, each track and race is different. Thus using the same car adjustments may result in poor performance for the next race. In the same way, adjusting a simulation until it matches one particular performance measure from one particular real system may only be a good result for that one real system. The problem here is ensuring that the extendability and robustness of the simulation exists in order to explore and extrapolate implications of other real systems in the same domain [9, 18, 51]. Having a micro-level model that is not properly sanctioned can ultimately lead to unreliable simulation results beyond the particular performance measures obtained for that particular real system modeled.

Another example to consider is a standard single server queuing model where the objective of the simulation is to achieve the theoretical performance [38]. For example, typical performance measures are the average time in the queue or system throughput. The standard approach to build this simulation is to observe the real system, measure arrival rates, measure server processing times, and then build a realistic representation of the system using some discrete event simulation packages. We expect the simulation to behave like the real system. Another way to build this simulation, utilizing a model fitting focus, is to create an ABM simulation with reproducing bugs. In this simulation, the bugs move about their environment looking for food and reproduce with other bugs, much like those of Sugarscape [20]. Key measures about the bugs, such as lifespan and birthrate, are mapped to the goal performance measures of the single server queuing model. Then, in the spirit of model fitting, parameters concerning the bugs and their environment are adjusted

using some algorithm until the simulation's performance measures match the expected queuing performance measures. The bug model is then deemed useful for queuing analysis. Although this is an extreme example, we have no doubts that both of these approaches can meet the expected theoretical performance measures. However, when it comes to sanctioning these techniques it is clear that the ABM bug simulation does not really represent the real system and therefore it is unlikely that anyone should sanction this simulation.

What emerges from analyzing these two sanctioning focuses is two conditional statements about sanctioning ABM simulations. The first condition comes from model fitting: if the macro behavior is sanctioned, then the micro behavior may be sanctionable. However, taking this approach requires one to also ensure that the micro behavior is sanctionable or else the simulation as a whole may not be sanctionable. The second condition comes from model testing: if the micro behavior is sanctioned, then either the macro behavior will be generated by the simulation or the appropriate micro behavior has not been captured by the simulation. This is a strong condition particularly if macro behavior, sometimes referred to as emergent behavior, is unexpected and surprising by definition.

ABM emergent and macro-level behavior is only really surprising or counter intuitive to us because *the way in which we can generate that behavior* and not in the fact that the phenomena exists in first place. Within the short history of ABM, the initial belief was that to generate complex and emergent behaviors a complex model was required. The true surprise came when it was found that very simple models could generate emergent behavior. Since this discovery, discussion of emergent behavior has proliferated and has resulted in some terminology confusion [19]. However, the idea of emergent behavior is not new and it can be observed in every scientific discipline in the form of abstraction levels. For example, from atoms emerge molecules, from molecules emerge cells, from cells emerge organisms, and etc. Therefore, the issue with our statement should not be with the fact that emergent behavior is surprising, but with how we can say that the macro-level behavior will appear when the appropriate micro-level behavior is included in the model.

We make this statement because micro-level behavior naturally leads to macro-level behavior. In

any system, entities can be identified that exhibit micro-level behavior that when examined together create macro-level behavior. In fact, the fundamental difference between micro-level behavior and macro-level behavior is the abstraction level of interest, even though both belong to the same system. Every macro-level behavior is the result of some micro-level behavior in nature. If one can appropriately mimic the micro-level behavior then the macro-level behavior will follow. Even though emergent behavior appears unpredictable, it is not unexplainable [19]. Thus, each ABM simulation study should begin by coming up with some statement similar to: we want to model these entities and their interactions in this environment to get appropriate macro-level behavior.

While model fitting is a useful model generating tool, it is not a proper sanctioning technique because it can generate one of the infinite models that does not represent the real system. Model testing should be the focus of any sanctioning technique because it emphasizes the need for the model to represent some abstraction of reality. Furthermore, sanctioning should focus on the micro-level behavior; having an the appropriate micro-level behavior helps to ensure that the complete model is sanctionable. If only the macro-level behavior is sanctioned, then considerable effort is still be required to ensure that the micro-level behavior is sanctionable. Up front, effort in micro-level sanctioning pays off in the long run as the complete model is sanctionable. If it turns out that the macro-level behavior is not emerging from the micro-level behavior, then there is some fundamental part of the puzzle missing from the micro-level model. This missing result will encourage designers to further research the micro-level behaviors, which is more in line with the spirit behind the scientific revolution. As pointed out in previous reports, simulations of less understood systems are becoming scientific theories in themselves, so it is vitally important that they be based on some representation of reality. Therefore, in order to represent that reality more emphasis needs to be on conceptual modeling (micro-level) of these less understood systems.

A survey of literature regarding how ABM simulations are sanctioned today generally indicates that the focus is on obtaining macro-level behaviors that match those of the real system. In other words, the emphasis of sanctioning techniques for these simulations of less understood systems is on Operational Sanctioning (macro-level) and not Conceptual Sanctioning (micro-level). This does

not mean that simulationists are completely neglecting micro level sanctioning, it only means that they tend to mention it in passing rather than making it the focus of their sanctioning efforts. For example, Epstein acknowledges the fact that many micro-level models can produce the desired macro-level behavior and that empirical testing of the micro-level model needs to be conducted. However, the general emphasis of Epstein's article is on how we can generate the macro-level behavior, in particular using Genetic Algorithms to generate the appropriate micro-level specifications [19]. Midgley et al. also discusses using Genetic Algorithms in their ABM simulation to obtain the desired macro-level behavior [55]. Moss and Edmond indicate that micro-level sanctioning needs to be qualitatively evaluated, but then spend the majority of article discussing statistical techniques to sanction the macro-level behavior [59]. Finally, Usher and Strawderman discuss the micro-level behavior of a crowd movement model and then state that their simulation is sanctionable because the macro behavior matches what is observed in real crowds [76].

Although there is some emphasis on the macro-level sanctioning over micro-level sanctioning, even those articles that discuss techniques to obtain macro-level behavior by developing the micro-level models emphasize achieving the macro behavior and not ensuring that the micro-level behavior is correct. Windrum et al. discuss three techniques to 'calibrate' a simulation [81]. In the first two techniques, Indirect Calibration and Werker-Brenner Calibration, empirical knowledge about the real system is used to help get the micro-level model close and then systematic adjustments are made to the model to obtain the macro-level behavior. Again, obtaining the appropriate micro-level behavior is not emphasized and instead the ultimate goal is to obtain the macro-level behavior by adjusting the micro-level parameters accordingly. In their last approach, called History-Friendly, the design of the micro-level behavior is based on empirical studies and historical data and the simulation is sanctioned based on whether the output matches the real systems output. Ultimately, this approach is probably the closest to having a focus on both micro and macro-level sanctioning, even though the final test as to whether the simulation is sanctionable depends upon the macro-level behavior and not on the micro-level behavior.

Micro-level sanctioning is not typically the emphasis of simulation. Macro-level behavior sanc-

tioning is still the important player. Therefore, a new methodology or tool that emphasizes micro-level sanctioning must address several fundamental issues with respect to ABM simulation sanctioning. These issues stem from the fact that ABM simulations are scientific theories by themselves that describe how micro-level behaviors creates macro-level behaviors.

As discussed in the previous validation report, simulations are beginning to become the epistemological engine of our time because they can be used to represent complex nonlinear systems and show the implications of those systems. As an epistemological engine, simulations have become the theories of the systems they are intended to mimic and therefore should be treated the same way as other scientific theories. Up to this point, the issues related to ABM simulation sanctioning have been discussed from the point of view of those developing the simulation, or in other words how individuals develop the theory of how the system behaves. However, as with any scientific theory, the theory must survive peer scrutiny and the theory must be reproducible. Regardless of how well an ABM simulation is built, it must undergo peer evaluation for scientific progress to be made.

This conclusion that ABM simulations of real systems need to be independently peer evaluated is not new. Several articles discuss the need for these simulations to be evaluated independently [3, 5, 81]. Hindrances to independent peer sanctioning include ambiguity in published papers, gaps in published descriptions or unclear descriptions, and technical difficulties related to simulation [3]. Apparently, since these theories, in simulation form, cannot be represented in simple equations or descriptions, attempts to describe them completely in words in a journal paper is either impossible or extremely difficult. This should not be a surprise given difficulty representing the complex nonlinearity of these systems and the fact that journals are probably not willing to publish an article long enough to completely describe a simulation.

One natural solution to this problem is for the authors to provide their peers with access to their simulation model; this, however, raises several issues. First of all obtaining a copy of the simulation model may be difficult due to proprietary issues. But even if the simulation is obtained, there are many simulation languages and packages that the simulationist could have used. If the evaluator is not familiar with the simulation language of the simulation, then understanding the simulation

can be a problem. Even if the evaluator is familiar with the simulation language and can run the simulation independently, the evaluator may be able answer the ‘how’ questions of the simulation but they still cannot effectively answer the ‘why’ questions. For example, the evaluator knows how an entity A behaves when it encounters an entity B, but the evaluator does not know why or with what justification entity A’s behavior was defined.

To make a determination about a simulation, an evaluator must understand the micro-level details of the simulation. This means the evaluator must have familiarity with simulation in addition to system domain knowledge. The evaluator must be able to abstract their domain knowledge into a simulation paradigm and be able to use this abstraction to understand the conceptual model for the simulation under evaluation.

It is clear that another approach is needed for describing an ABM simulation. The approach must provide information on such things as initial conditions, all logic associated with the micro-level entities and justification behind the logic, how the entities interact and the justification, variables, parameters, probability distributions, random number generators, and terminating conditions. The tool should also emphasize the proper sanctioning required at the micro-level, aid in the development of the conceptual model at the micro-level, be a description of the simulation model that can be independently evaluated by experts of varied simulation experience levels, and be able to provide both the hows and justification of those hows at the computer programming level and at the computer novice or domain expert level. Creating a description that meets all of these needs and is still useful to the evaluator is a challenging task.

5.1 Understanding the Needs: How to Represent a Complex System

The following needs for a new ABM tool have been identified:

1. Aids in learning and conveying system knowledge
2. Incorporates proper engineering judgment
3. Aids in translating the conceptual model into a computerized model
4. Emphasizes the development and sanctioning of the micro-level behaviors

5. Displays the theories and assumptions built into the model for quantitative analysis
6. Conveys the conceptual model's logic and structure for qualitative analysis
7. Completely represents the simulation so it can be reproduced by independent evaluators
8. Provides justification for all structures and actions in the simulation
9. Reviewable by evaluators of varied simulation and domain expertise levels

Reflected in this set of needs are three reoccurring ideas. The first is the need to observe the real system and create a conceptual model translatable into a simulation. The second is that the sanctioning emphasis is on understanding the micro level behaviors of the system. Finally, the third sanctioning by a variety of evaluators. A visual model of how a ABM simulation represents a complex, nonlinear real system is one tool that has the appropriate capabilities to address all of the above needs. In the follow paragraphs, some of these capabilities are discussed.

There are capabilities from the model aspect of the tool. Generally speaking, a model is a representation and description of a real system. The visual model will be a representation of an abstracted real system for the purpose of simulating that real system. The model will convey information about the details of the simulation, information about the real system itself, and about the conceptual model used to represent that real system. This model can thus be an independent medium for evaluators to review the effectiveness of the simulation and also provide documentation independent of a computer program.

Building a visual model forces the modeler to actively gain knowledge about the real system in order to abstract system behaviors for the purposes of the study. Using a model as a tool aids the simulationist in building a conceptual model while enforcing the simulationist to consider how the model will be transformed into a simulation.

There are capabilities from the visual aspect of the new tool. A major hurdle for the simulationist to overcome is conveying knowledge about how the real system operates and how the simulation of the real system operates. Since humans are inherently visual and good visual aids can convey a

wealth of information, an appropriately designed visual modeling tool can facilitate communicating complex structures and any inter-relationships.

Finally there are capabilities from using a standardized visual model. Modeling methods need a formalism and they need to convey specific knowledge. Model method standardization helps to ensure the model provides enough knowledge for the simulation to be rebuilt (if necessary) by independent evaluators, be formal enough to possibly translate into a simulation language, and specific enough to ensure the micro-level behavior is both qualitatively and quantitatively sanctionable.

To some extent every discipline uses models to aid in understanding. A good place to begin examining visual-based modeling methods is in the fields of software and system engineering because they use many different types of visual models [29, 45]. These fields typically use Schematic or Descriptive Models that attempt to represent a system process or element [45], which accurately describes the kind of model we are looking for. However, a significant number of visual models are not capable of representing a complex, nonlinear system. Many visual models are good at describing the linear flow of information, but get increasingly confusing if the system has any interactions between entities [29], which is a common theme in the ABM paradigm. This inability of many Schematic Models to represent complex systems automatically eliminates many of the models found in systems or software engineering such as ones found in the Unified Modeling Language [11].

There are more recently developed models, such as Petri Nets [62] and Statecharts [29], that are capable of effectively representing a complex system. Petri Nets have some trouble representing hierarchies while Statecharts seem to easily convey a complex system's structure, relationships, and hierarchies. Statecharts are based on state-diagrams and "constitute a visual formalism for describing states and transitions in a modular fashion, enabling clustering, orthogonality (i.e., concurrency) and refinement, and encouraging 'zoom' capabilities for moving easily back and forth between levels of abstraction" [29]. Statecharts provide many of the capabilities that are needed for a new ABM modeling tool. In fact AnyLogic, a commercial ABM simulation language, has incorporated Statecharts into their language to describe the behavior of agents [12]. Thus, Statecharts appear formal

enough to translate behaviors into computer executable logic.

Despite the many capabilities of Statecharts, there are some shortcomings. The first, as with Schematic Models, Statecharts do a good job of describing how a process happens but does a poor job describing why it is made to happen that way. Therefore, a way to convey the ‘whys’ needs to be incorporated into the Statechart Model. Statecharts may actually be too formal and detail-oriented. While formalism and richness in detail are important, having an extremely formal and detail model can be a hindrance. Too formal and detailed the model effort shifts emphasis away from building a good simulation study and places it instead on building this abstract model. Furthermore, too formal and complicated model may make understanding the key concepts of the model more difficult. Therefore, some simplifications of the Statechart Model are needed to avoid over formalization and puts the emphasis on abstracting the real world system into micro-level behaviors that are achieved within the simulation.

5.2 Prototype Design of the New ABM Tool: Theoretical System Simulation Diagram

5.2.1 Fundamental Characteristics

A prototype design of a new ABM modeling tool was developed to have the appropriate capabilities to meet the defined needs. This new ABM tool, the Theoretical System Simulation Diagram Prototype or TSSD Prototype, is based primarily on the structure of Statecharts because of their ability to visually represent structures and relationships of complex systems [29]. However, the TSSD Prototype only borrows the fundamental structural pieces from Statecharts for visual representation purposes and does not incorporate many of the definitions that compose and define a Statechart. For example, the TSSD Prototype uses arcs to represent the movement from one block of activities to another, but unlike Statecharts arcs, TSSD Prototype arcs do not determine when to transition among blocks or states.

The TSSD Prototype also incorporates some of the basic shapes and properties of Flowcharts for the purpose of simplifying and providing flexibility to those using the TSSD Prototype. For example, within each block or state, there are a set of actions, interactions, or decisions shapes that

define the activities occurring when the modeled entity is in that block or state. Without requiring the users to specifically define events and the transitions to new states, the TSSD Prototype allows a certain flexibility to redefine the activities without needing to completely build a new model. Having simple shapes representing different activities within a block allows evaluators or builders to quickly determine what is occurring in that block without the need for the detailed logic of the model. Incorporating these simple shapes allows users to spend more time focusing on behaviors and activities and less time worrying about how best to design the logic of the computer simulation. Even with these change, the TSSD Prototype remains formal or provides enough information to translate the diagram into a simulation.

The TSSD Prototype incorporates visual aspects to aid the user in defining and understanding the hierarchical structure, relationships, and activities that occur in each block or state. The TSSD Prototype also incorporates a database of properties for each shape to provide details necessary to properly sanction and build the simulation. For example, an Action shape has five properties that define it's relationship to other shapes, the behavior of the real system it is trying to mimic, pseudo code to help to translate that real system behavior into a simulation, and a reference to the source that describes that behavior occurring in the real system. With these properties, the TSSD Prototype contains much of the 'why' information that is often lacking in other modeling techniques. This defines why the TSSD Prototype can be translated into a simulation and provide enough information for evaluators to review the sanctionability of the simulation.

5.2.2 Basic TSSD Prototype Structure

There are five shapes that are used to construct the TSSD Prototype. The first shape is the Block, visually represented as a rectangle, which defines a collection shapes. The Block is the fundamental shape that describes the hierarchical relationship and structure of the system being simulated and it can be used to describe that state of the system at any point in time. For example, a Block called Environment could describe the collection of shapes that describe the environment within which the agents in the real system operate. Activities or behaviors within a Block could be defined to occur 'synchronously' or 'concurrently,' which is a fundamental property of Statecharts. Actions within a






Block could define the creation of more instances of Blocks. For example, in a Build Agents Block there could be an action that says create 40 agents.

The TSSD Prototype has shapes to describe the micro-level behaviors that occur within the model. The Action shape describes an activity that changes an entity or variable and is visually represented with a block arrow. An example of an Action may be the consumption of food or moving to a new location. A special case of the Action shape is called the Interaction Shape, visually represented by a circle. The Interaction Shape defines the exchange of information between agents. For example, an Interaction could be the communication of the location of an important resource between two agents. There are a couple reasons for creating and highlighting the Interaction shape. First, when evaluating or building an ABM simulation it is important to understand the interactions that occur between agents; making it a special shape makes the identification of this behavior easier. Second, the interaction between agents within a simulation often requires special programming attention; explicitly capturing this behavior can aid in translating the TSSD Prototype into a simulation. The final shape that describes micro-level behavior is the Decision shape, visually represented with a diamond. A Decision shape examines a set of conditions and determines whether to transition to another block. For example, a Decision could be to transition from a Move Block to a Stop Block if an obstacle is encountered by some moving agent.

The final shape used in the TSSD Prototype is the Information Shape, visually represented with an horizontally elongated circle. The Information shape provides information referenced by the Behavior Shapes. For example, an Information shape could represent the simulation clock that a Decision shape references to see if the simulation should terminate. An Information shape could represent the schedule by which Agents are created or destroyed. The Information shape is included in the TSSD Prototype to convey key information that the simulationist does not want to represent using Blocks, Actions, Decisions, and Interactions.

These shapes, also have specific properties. A Block shape has five properties. The *From* property describes which blocks transition to it. The *To* property describes blocks that can transition to it. The *Decisions*, *Actions*, and *Interactions* properties describe the respective shapes that are

Table 3: TSSD Prototype Shape Descriptions

Shape	Name	Properties	Definition	Example
	Block	From To Decisions Actions Interactions	A collection of actions, decisions, interactions, and/or information. Can also be a collection of several blocks. Describes the state of the simulation or entity at a point in time.	Initialization, Building, Searching, Crossing
	Action	Member Of Impacts Decision Behavior Pseudo Code Source	An activity describing the change of an entity or variable.	Set Coordinates, Travel Home, Destroy Agent
	Decision	Member Of Transitions To Behavior Pseudo Code Source	A condition that describes when to transition to another Block as well as what Block to transition to.	Interrupt?, Go Home?, Begin Simulation?
	Interaction	Member Of Interacts With Behavior Pseudo Code Source	A special Action describing the exchange of information between agents.	Look for Neighbor, Kill Enemy, Calculate Distance to Neighbor
	Information	Member Of Behavior Source	A parameter or constant	Clock, Initial Number of Agents, Schedule

directly contained within the block. The fundamental properties of the Block shape describes its relationships to other shapes.

The Behavior Shapes (Action, Decision, and Interaction) also have five properties that are closely related to each other. The first property is the *Member Of* property, which describes the direct block that the shape belongs to. The next property describes the activity being performed. For the Action shape, the *Impacts Decision* property describes any Decision shape that is being impacted by the action. For the Decision shape, the *Transitions To* property describes blocks that can be transitioned to as a result of the decision. For the Interaction shape, the *Interacts With* property describes the blocks or shapes that the interaction impacts. The final three properties for the Behavior Shapes are exactly the same. The *Behavior* property describes a behavior of the real system that the shape is attempting to mimic. The *Pseudo Code* property describes the pseudo code intended to facilitate translating the behavior into the simulation. Finally, the *Source* property is a reference to a source that provides the justification for that behavior.

The final shape, the Information shape, has three properties; *Member Of*, *Behavior*, and *Source*. Each of these properties are as previously defined for Behavior Shapes. A complete summary of the shapes, their visual representation, properties, definition, and some examples are shown in Table 3. Although the TSSD Prototype is still in the early stages of development, we believe that

the fundamental shapes and the structure associated with Statecharts can effectively provide rich descriptions of most ABM simulations. As a proof of concept, to further describe the functionality of the TSSD Prototype, and to show TSSD Prototypes effectiveness at sanctioning, a TSSD Prototype will be developed based on the Bay of Biscay ABM Simulation Scenario 1. Furthermore, a template of the TSSD Prototype will be developed using Microsoft Visio 2007 that will have all of the TSSD Prototype shapes and properties. These will be documented in the final deliverable.

5.3 Section Concluding Remarks

This portion of the research effort examined elements of ABM sanctioning particularly those elements of current practice that appear at odds with the requirements of viable ABM sanctioning methods. An initial ABM developmental methodology was defined that focuses on providing sanctionable ABMs. This methodology was culled from a broad examination of developmental methods used for simulation in general and, if available, ABM in particular. A key objective was to promote sanctionability from both a micro-level and macro-level ABM behavior perspective.

The last portion of this research will focus on re-creating the Bay of Biscay ABM of [16]. Champagne used the past, historical record to define the ABM. Verification techniques were employed to ensure the model was correctly built and macro-level validation efforts were designed and exploited. This research extends [16], and subsequent papers [32] and [31], by using the initial ABM developmental methodology developed in this research to define, design and realize this Bay of Biscay ABM.

6 Proof of Concept Case Study

As a proof of concept, to further describe the functionality of the TSSD prototype, and to show the TSSD Prototypes effectiveness at sanctioning, the TSSD Prototype was used to build a simulation that replicates Scenario 1 of the Bay of Biscay ABM Simulation as described by Champagne [16]. Earlier the basic rationale behind selecting this ABM Simulation to investigate with the TSSD Prototype was provided. However, another primary reason for selecting this simulation was the amount of available documentation. Because a significant portion of his dissertation was spent

describing the Bay of Biscay ABM Simulation many aspects required to reproduce the simulation are included. As discussed earlier, this amount of documentation is rarely provided for published simulation projects. With these vast amount of details, we could more effectively develop the TSSD Prototype as a proof of concept, describe the functionality of the prototype, more effectively show the usefulness and application of the TSSD Prototype, and further improve the TSSD Prototype and identify short comings. Ultimately, deciding to reproduce another simulation puts more emphasis on the TSSD Prototype and less emphasis on creating an entirely new simulation.

A secondary reason for selecting this ABM Simulation was to have a more meaningful discussion regarding the sanctionability of the reproduced ABM Simulation using the TSSD Prototype. In his dissertation, Champagne provides both conceptual and operational benchmarks that can help gauge the sanctionability of our ABM Simulation. For example, Champagne provides justification and documentation regarding the logic of how Aircraft searched for Uboats and how the Uboats responded to the Aircraft. Using this documentation we can build a better conceptual model and also make some claims about the sanctionability of our conceptual model. Also, Champagne provides key operational measures such as Number of Sightings and Number of Kills that can be used to operationally sanction our simulation. Being able to make these comparisons will further support the effectiveness and justification of the TSSD Prototype. It should be made clear that we are not directly comparing our simulation with the real system, but with another well documented and sanctioned simulation.

6.1 Scenario 1 of the Bay of Biscay ABM Simulation

In Champagne's Bay of Biscay ABM Simulation there are two scenarios that approximately correspond to the developments that occurred in WWII. For simplicity we only replicated Scenario 1. However, the changes presented in Scenario 2 could easily be incorporated in the TSSD Prototype and the resulting ABM Simulation. In the next few paragraphs, a general description of Scenario 1 will be described as given in Champagne's Dissertation. For more detailed information see [16].

Scenario 1 takes place over the six month period from October 1942 to March 1943 and is preceded by a 12 month warm-up period where only the Uboats travel between the North Atlantic

and French Ports. After the 12 month warm-up period the Aircraft take off from England and search for the Uboats in a 50 NM by 50 NM search area using the Modified Barrier Search Pattern. Altogether the Aircraft are randomly assigned to search 200 NM x 350 NM area that is not within a 100 NM of the French coast. The Aircraft are schedule to take off uniformly throughout a 24 hour period and must be at the base for a 12 hour period before taking off again. Throughout the simulation there are 19 Aircraft such that the number of sortie hours approximately correspond to the historical number of sortie hours flown during that six month period. Aircraft travel at 120 knots and search for Uboats for 7 hours and then return to the base. If at anytime the Aircraft detects a Uboat and fires, the Aircraft will immediately return to the base.

During this Scenario the Uboats observe the maximum submergence and nighttime surfacing policy. Day is defined as the time between Nautical Dawn and Nautical Dusk and Night is the the time between Nautical Dusk and Nautical Dawn. While on the surface the Uboats will scan for Aircraft and upon detection will submerge until it is nighttime. On the surface the Uboats travel at 10 knots and while submerged they travel at 2.5 knots. They can only travel 3 hours submerged before they must surface to recharge their batteries if they wish to continue moving. For every hour traveled on the surface, the batteries have one more hour of submerged travel time up to a total of 3 hours of submerged travel. When they are within a 100 NM of the French coast they will travel on the surface because they will have German Aircraft support. During the warm up period the 70 Uboats are randomly distributed across the Bay of Biscay and are set to either head towards their home port or the North Atlantic. Once the warm up period is over, at the beginning of each month a scheduled number of Uboats enter the simulation at their home port and head toward the North Atlantic. While in the North Atlantic the Uboats have 30 days worth of provisions and have a 25% chance of extending their time in the North Atlantic by another 30 days. This models the limited number of resupply ships employed by the Germans to keep the Uboats operational. Once the Uboats reach their port they will uniformly depart again after 25 to 40 days.

For detection both the Aircraft and the Uboats used the Inverse Cube Law, which incorporates the use of multiple sensors to create a positive probability of detection regardless of the distance

away from the target. Although the sensors and their range was not directly provided by Champagne for the Aircraft and the Uboats, Champagne did reference [54, 78] which gave the breakdown of the sensors. Based on those references, the sweep width and detection probability was calculated. Furthermore, when an Aircraft detects a Uboat it has a 0.02 probability of killing the Uboat regardless of the time of day.

Although many details of the simulation that were given by Champagne are beyond what is typically encountered in literature, there were still many aspects of his simulation that were not completely clear. There are several reasons for this. First, Champagne primarily relies on written descriptions to convey the logic and activities of the simulation. This means of communication can be fairly confusing if the sentence is not worded in just the right way. For example, saying Uboats submerge after detecting an Aircraft and then resurface at night could be interpreted several different ways. Do they resurface immediately if it is night time? Do they resurface after 3 hours and it is still night time? Unless the written description is in a clear structure and formalism there can be many misinterpretations of the logic. This problem with written descriptions further indicates that some sort of formal description of the model needs to be established if someone is going to attempt to reproduce a model or even interpret the model's sanctionability.

Closely related to the lack of formal description is that the written description provides no structure forcing the author to describe all of the necessary details in order for others to reproduce their work. After spending a significant amount of time building a simulation it is likely that small, yet critical, pieces of the simulation will be left out of written descriptions because they are assumed by the simulationist. There are many examples in Champagne's work where we believe that this occurred. For example, Champagne did not include critical information regarding the detailed sensor data in his descriptions of the model. In this particular case we were able to find the sensor data that we *believe* he used, however we cannot be 100% certain. It could be conjectured that if a formal description was required, then some of these unintentional omissions would be averted.

Overall, the description of the simulation given in Champagne's Dissertation provided most of the information needed in order to reproduce both the conceptual and operational aspects of

the simulation. When we encountered something that was unclear we attempted to review any references given for clarification, however if no reference were given we made reasonable assumptions and recorded it as such in the TSSD Prototype that was developed.

6.2 Bay of Biscay TSSD Prototype Description

Based on Champagne's Dissertation, a TSSD Prototype was developed that serves as a medium between the real world system and the simulation model. In essence, the TSSD Prototype is the descriptive and definitional abstraction of the real system that serves as a translator between the infinitely complex real world and the finitely defined simulation. The main objective of this section is to describe the TSSD Prototype's functionality and further explain how the TSSD Prototype can aid in building better simulation models.

Using Microsoft Visio 2007 a template of the fundamental shapes and their properties described earlier was created. From this basic template the TSSD Prototype of the Bay of Biscay ABM Simulation was built. Both the template file and the TSSD Prototype file of the Bay of Biscay are included in this technical report for further examination beyond the brief descriptions provided here. The reason for selecting Visio to develop the TSSD Prototype was the ease of use, the ability to run reports to obtain the properties of all the shapes, and the general public knowledge of Visio.

Within the TSSD Prototype file there are four different sheets or views of the simulation. The first view is the 'Bay of Biscay Model' shown in Figure 7. This view shows the model level abstraction of the simulation and includes initialization blocks and execution blocks. In this view the major actions taken to initialize and run the simulation are shown. The second view called 'Environment' shows the the environment level of abstraction and can be seen in Figure 8. This includes the Uboats and the Aircraft that exist in the environment, their basic interaction, and the progression between night and day. It should be noted that the 'Environment' view is part of the 'Bay of Biscay Model' view and that the only reason for having separate views is to aid the user in viewing different levels of abstraction in the TSSD Prototype. The final two views are the 'Uboat' (Figure 9) and 'Aircraft' (Figure 10) views, which display the detailed behaviors of each agent.

What can be observed from an initial look at each of these views is the hierarchical structure

Figure 7: TSSD Prototype - Bay of Biscay Model View

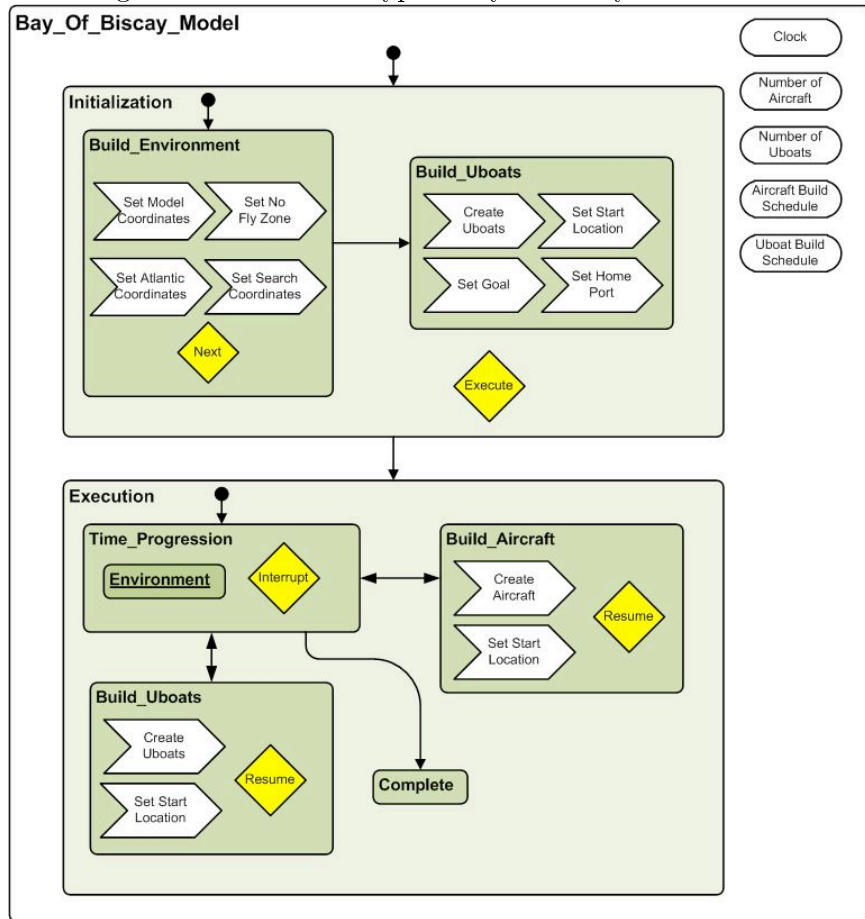
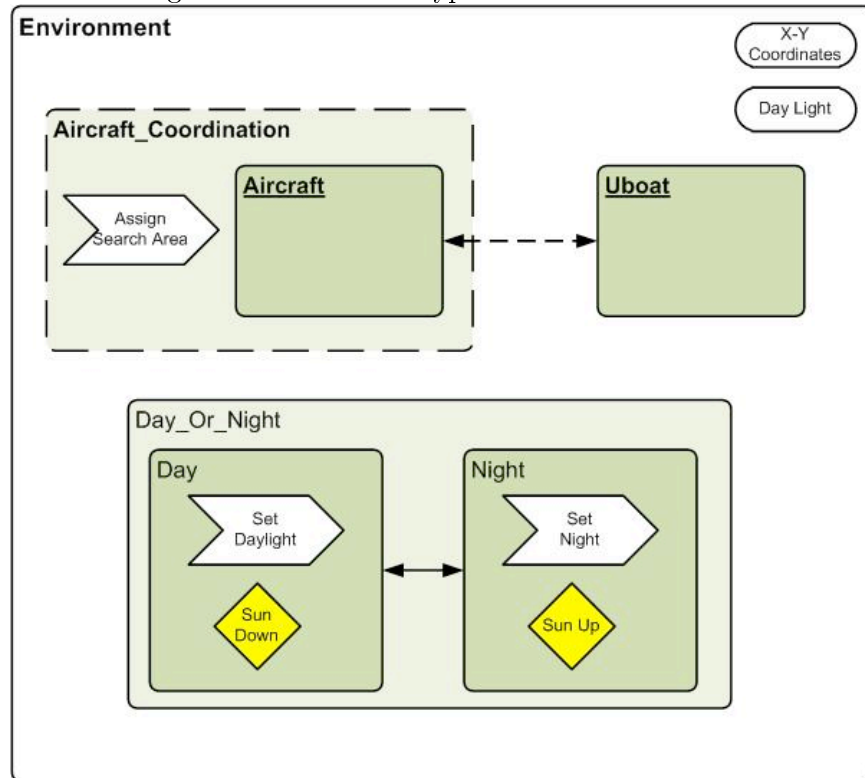


Figure 8: TSSD Prototype - Environment View



of the actions that take place in the real world and the simulation. For example, in the Uboat view (Figure 9) there are many actions that can happen while the Uboat is crossing the bay such as they can be on the surface, submerged and moving, submerged and stopped, or be destroyed. Furthermore, the boundaries of the real world abstraction and the simulation can clearly be seen. We can observe in the Environment view (Figure 8) that the agents involved are the Aircraft and Uboats and that the time of day plays a role. In the Model view (Figure 7) we can see the simulation details, the building of the environment, the creation of new agents, and other important simulation aspects that define when the simulation begins and ends.

Although the visual aspects of the TSSD prototype provide a lot of information for the reader, it does not provide enough information for the purposes of sanctioning this conceptual model or building the simulation. This is where the database of information/properties related to each model shape fills in the gaps. For each shape in the TSSD Prototype there are a series of properties that further define the details of that shape. In the Visio file of the TSSD Prototype, whenever a shape

Figure 9: TSSD Prototype - Uboat View

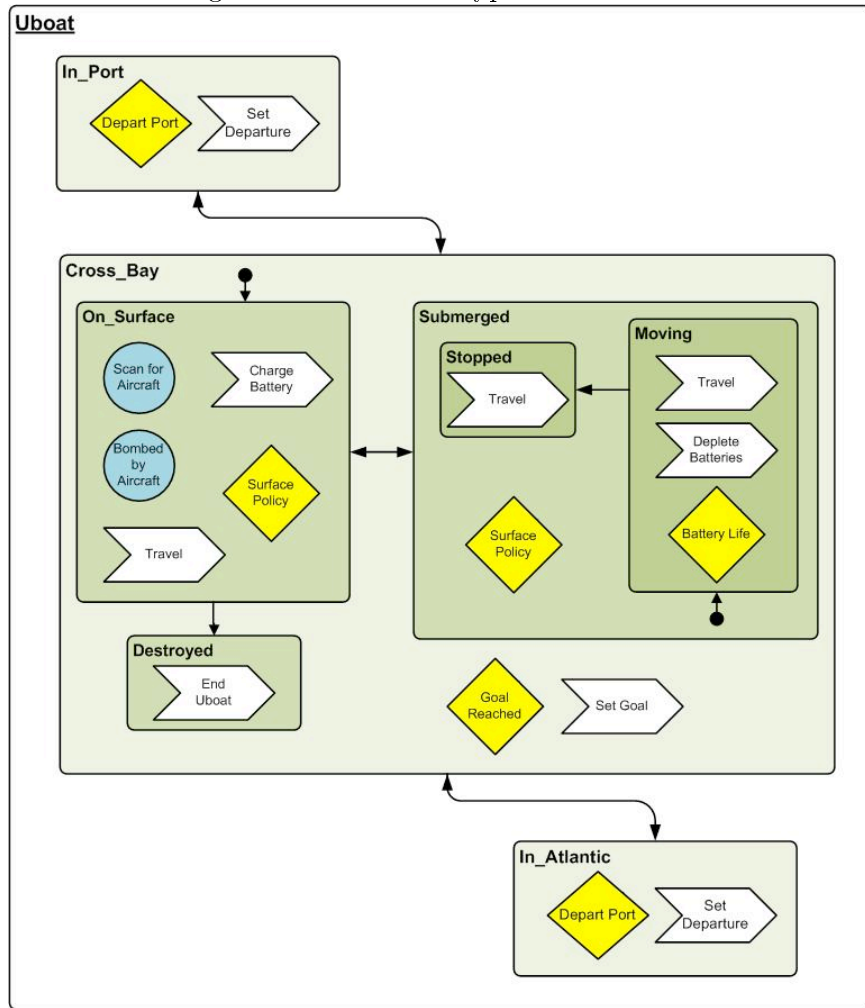


Figure 10: TSSD Prototype - Aircraft View

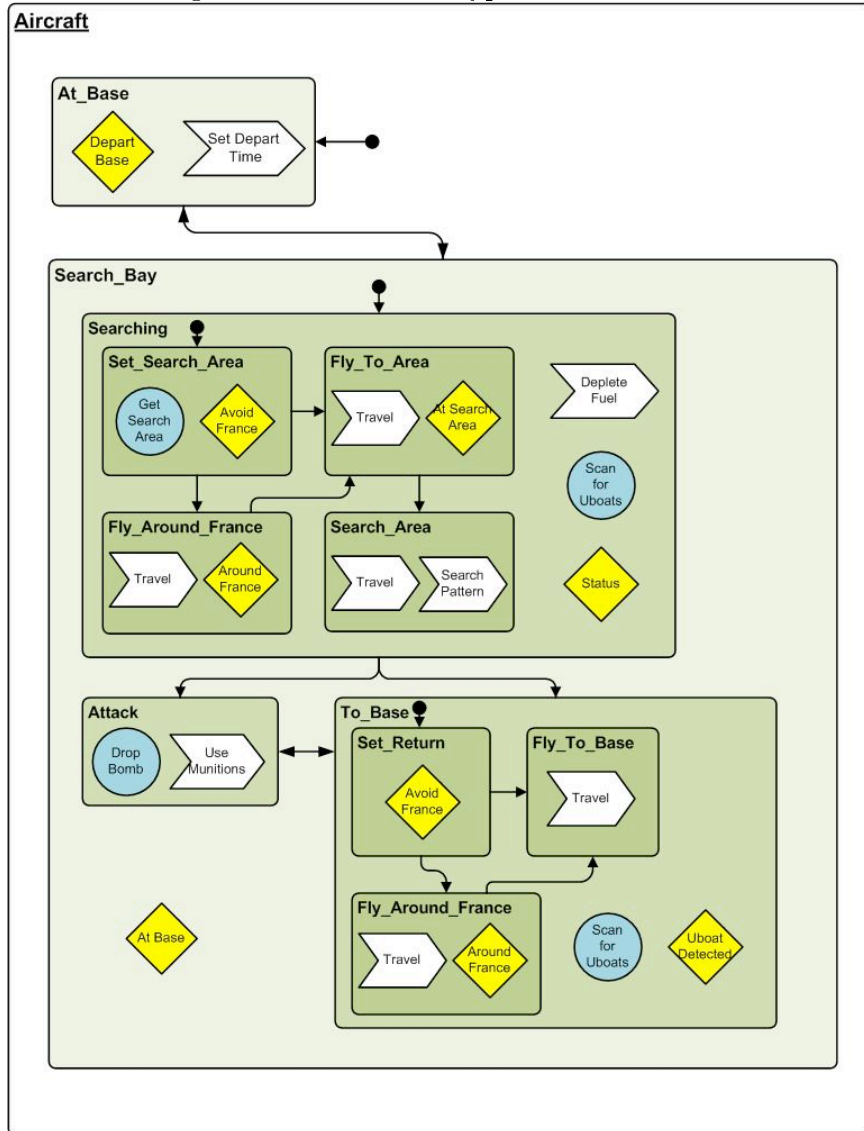
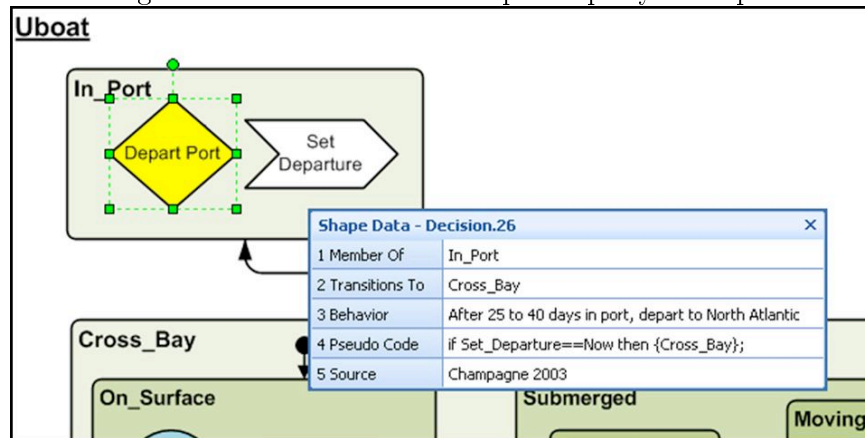


Figure 11: Uboat Decision Shape Property Example



is selected a Shape Data Window opens and prompts the user to fill out the properties associated with that shape. For example, a screen shot of the TSSD Prototype in Visio shows the Shape Data Window for the ‘Depart Port’ decision shape in the Uboat block in Figure 11. In the ‘Depart Port’ shape there are five associated properties: *Member Of*, *Transitions To*, *Behavior*, *Pseudo Code*, and *Source*. The associated value of the *Member Of* properties is In_Port because this describes the block that the decision shape is residing in. Furthermore, the *Transitions To* property is set to Cross_Bay because this describes the block that is transitioned to as the result of the decision. The *Behavior* property describes in plain English the behavior that the decision shape executes and in turn the *Pseudo Code* property describes the pseudo code to execute the described behavior. Finally, the *Source* property describes the source of the behavior, and in this case, Champagne’s 2003 Dissertation [16] is referenced. For more details on all of the properties associated with each shape, see the earlier section discussing the TSSD Prototype design.

To further define the functionality of the TSSD Prototype we will spend the remainder of this section describing the Uboat block of the TSSD Prototype. By the end of this section the reader should have a good understanding of how to read and interpret the TSSD Prototype. Therefore, if the reader wishes to learn more about the details of the simulation, then please review the TSSD Prototype file. Before proceeding to the detailed description of the Uboat block it may be helpful to have both the TSSD Prototype visual of the Uboat block as well as the properties of all of the shapes included in the Uboat block. Although this can be done dynamically with the Visio

file, all of the necessary information to aid in understanding the Uboat block have been included in this report. The visual of the Uboat block is shown in Figure 9. The list of action properties, block properties, decisions properties, and interaction properties of the shapes within the Uboat block are shown in Figures 12, 13, 14, and 15, respectively.

In the previous sections, the real world description of the Bay of Biscay and the behavior of the Uboats was provided. From this real world description, we used the principles of the TSSD Prototype to develop a conceptual model of the Uboat behavior that captures the desired level of abstraction, provides documentation of that conceptual model, and is designed for simulation. The first thing to notice is that in the description we stated that there are 70 Uboats. However in the TSSD Prototype there is only one Uboat block. This does not mean that all of the Uboats are homogeneous or that there is only one Uboat. In fact, the Uboats are heterogeneous because each Uboat is assigned to have different home ports and destinations in the North Atlantic (this aspect is defined in the Initialization block). However, while some of the parameters of the Uboats are heterogeneous, their behavior logic is homogeneous. Therefore, we can represent all Uboats with a single conceptual block with generic parameters that can be adjusted for each individual Uboat. It is important to remember here that we are not building the TSSD Prototype as if it were a machine executable simulation. Instead, the TSSD Prototype provides a medium between the real world and the simulation, so we leave the actual construction of the simulation to the simulationist. The goal of the TSSD Prototype is to lead to better simulations and simulation projects; not to be a simulation itself. Thus, we are concerned with representing Uboat behavior in a conceptual sense that can eventually be programmed into a computer simulation.

For simplicity we will describe the behavior of a Uboat as if it were starting at it's home port and is about to cross the Bay of Biscay in route to the North Atlantic. Therefore, we will begin in the In_Port block (the Uboat is in it's home port). Within the In_Port block there is one action shape, Set Departure, and one decision shape, Depart Port. While in this block the Uboat executes the Set Departure action, which sets a departure time to be 25 to 40 days into the future, and the Uboat evaluates whether it is time to depart based on the Depart Port decision shape. As described

Figure 12: Uboat Action Shape Properties

Master Name	Displayed Text	1 Member Of	2 Impacts Decision	3 Behavior	4 Pseudo Code	5 Source
Action	Set Departure	In_Port	Depart Port	Uboats depart 25 to 40 days after arriving in port.	Set_Departure=Now+Uniform(25,40) days	Champagne 2003
Action	Travel	On_Surface	Surface Policy	Move towards the goal heading at speed=10 NM/hr allowed while On_Surface	Move To goal(speed=10 NM/hr)	Champagne 2003
Action	Charge Battery	On_Surface	Surface Policy	The batteries take 3 hours to charge while On_Surface	If battery_charge<fully_charged then {if hour_has_passed==true then {battery_charge++}};	Champagne 2003
Action	Travel	Stopped		Move towards the goal heading at speed=0 NM/hr allowed while Stopped	Move To goal(speed=0 NM/hr)	Champagne 2003
Action	Travel	Moving		Move towards the goal heading at speed=3 NM/hr allowed while Submerged	Move To goal(speed=3 NM/hr)	Champagne 2003
Action	Deplete Batteries	Moving	Battery Life	Batteries only last 100 NM before being depleted	If 100 - distance_traveled<=0 then {battery_depleted==true};	Champagne 2003
Action	End Uboat	Destroyed		When a Uboat is hit with a bomb it is destroyed	End Uboat Instance	Champagne 2003
Action	Set Goal	Cross_Bay	Goal Reached	If coming from the Atlantic, then the Uboat will head towards their home port. Otherwise, they will head somewhere towards the North Atlantic	If from_port==true then {goal=North_Atlantic}; Else {goal=Home_Port};	Champagne 2003
Action	Set Departure	In_Atlantic	Depart Port	Uboats stay in Atlantic for 30 days with 25% chance of staying 60 days	If Uniform(0,1)<0.25 then {set_departure=now + 60 days}; Else {set_departure=now + 30 days};	Champagne 2003

Figure 13: Uboat Block Shape Properties

Master Name	Displayed Text†	1 From	2 To	3 Decisions	4 Actions	5 Interactions
Block	In_Port	Cross_Bay	Cross_Bay	Set Departure	Depart Port	
Block	Cross_Bay	In_Port, In_Atlantic	In_Port, In_Atlantic	Goal Reached	Set Goal	
Block	On_Surface	Cross_bay	Submerged	Surface Policy	Travel, Charge Battery	Scan for Aircraft, Bombed by Aircraft
Block	Submerged	On_Surface	On_Surface	Surface Policy		
Block	Stopped	Moving			Travel	
Block	Moving	Submerged	Stopped	Battery Life	Travel, Deplete Batteries	
Block	Destroyed	On_Surface			End Uboat	
Block	In_Atlantic	Cross_Bay	Cross_Bay	Set Departure	Depart Atlantic	

Figure 14: Uboat Decision Shape Properties

Master Name	Displayed Text	1 Member Of	2 Transitions To	3 Behavior	4 Pseudo Code	5 Source
Decision	Depart Port	In_Port	Cross_Bay	After 25 to 40 days in port, depart to North Atlantic	if set_departure==now then {Cross_Bay};	Champagne 2003
Decision	Surface Policy	On_Surface	Submerged or Destroyed	If aircraft is detected then submerge. If bombed by aircraft then the Uboat is destroyed. If there is daylight then submerge.	if aircraft==detected or daylight==true then {Submerged}; if destroyed==true then {Destroyed};	Champagne 2003
Decision	Battery Life	Moving	Stopped	If the Uboat's batteries run out, then the Uboat must stop moving but remained Submerged	if batteries_depleted==true then {Stopped};	Champagne 2003
Decision	Surface Policy	Submerged	On_Surface	If Uboats is submerged because aircraft was detected, then the Uboat will not surface until the next days night time. Otherwise, the Uboat will surface when there is no daylight.	if aircraft_detected==false & daylight==false then {On_Surface}; if next_day==true & daylight==false then {On_Surface};	Champagne 2003
Decision	Goal Reached	Cross_Bay	In_Atlantic	Once the goal location is reached, the Uboat waits for a period of time in either the Atlantic or in their Home Port	if from_port==true & current_location==goal then {In_Atlantic}; if from_port==false & current_location==goal then {In_Port};	Champagne 2003
Decision	Depart Port	In_Atlantic	Cross_Bay	Uboats leave port with approximately 30 days of supplies and there is a 25% chance that they will be able to extend their stay to 60 days.	if set_departure==now then {Cross_Bay};	Champagne 2003

Figure 15: Uboat Interaction Shape Properties

Master Name	Displayed Text	1 Member Of	2 Interacts With	3 Behavior	4 Pseudo Code	5 Source
Interaction	Scan for Aircraft	On_Surface	Aircraft	Scan for aircraft using the Inverse Cube Rule with 3 devices (Metax-15mi, Naxos=15mi, Vision=2mi). W=swsweep width=sqrt(15^2+15^2+2^2)	For (each Aircraft within W) {prob_detect=(1-e^(- (W^2)/(4*P*(dist_to_aircraft)^2)); if Uniform(0,1)<prob_detect then {aircraft_detected=true}};	Champagne 2003, MicCue 1990
Interaction	Bombed by Aircraft	On_Surface	Aircraft	If a Uboat is hit with a bomb from an aircraft, then it is destroyed	If bombed==true then {destroyed=true};	Champagne 2003

in the Depart Port decision shape properties, the Uboat will transition to the Cross_Bay block when the current simulation time is equal to the departure time.

When the Uboat transitions to the Cross_Bay block, it will immediately transition to the On_Surface block within the Cross_Bay block. This behavior is denoted by the circle with the arc pointing to the On_Surface block. Before describing the behaviors that occur in the On_Surface block, it is important to note that there are two shapes that are members of the Cross_Bay block which have precedence over all of the shapes within the On_Surface, Submerged, and Destroyed blocks because they are at a higher level of aggregation. In other words, the Uboat can be crossing the bay in many different states, but once they reach their goal they will automatically transition to the In_Atlantic block regardless of whether they were submerged or not. This idea of hierarchy is very important in the TSSD Prototype because it helps capture and convey complex behaviors. Within the Cross_Bay block the Set Goal action shape sets a goal location in the North Atlantic and the Goal Reached decision shape evaluates when that goal is reached and to then transition to the In_Atlantic block.

If the goal has not been reached, then the internal blocks of the Cross_Bay continues to execute. Within the On_Surface block there are several different shapes. The Charge Battery action shape is a reoccurring action shape that charges the batteries used in submerged travel. The Travel action shape is also a reoccurring shape that moves the Uboat towards the goal at 10 knots. In essence, the Travel action shape is updating the location, which is being evaluated by the higher level Goal Reached decision shape. The Scan for Aircraft interaction shape is a reoccurring action that accesses all of the Aircraft in the simulation location and calculates whether they can detect any aircraft in their area. For more details, see the Uboat Interaction Shape Properties in Figure 15. The Bombed by Aircraft interaction shape checks to see if any Aircraft have recorded hitting a Uboat. Finally, the Surface Policy decision shape determines whether it is time to transition to the Destroyed block of the Submerged Block. If the Uboat has been hit, then it will transition to the Destroyed block. If the Uboat has spotted an Aircraft, or if the sun is up, then the Uboat will transition to the Submerged block.

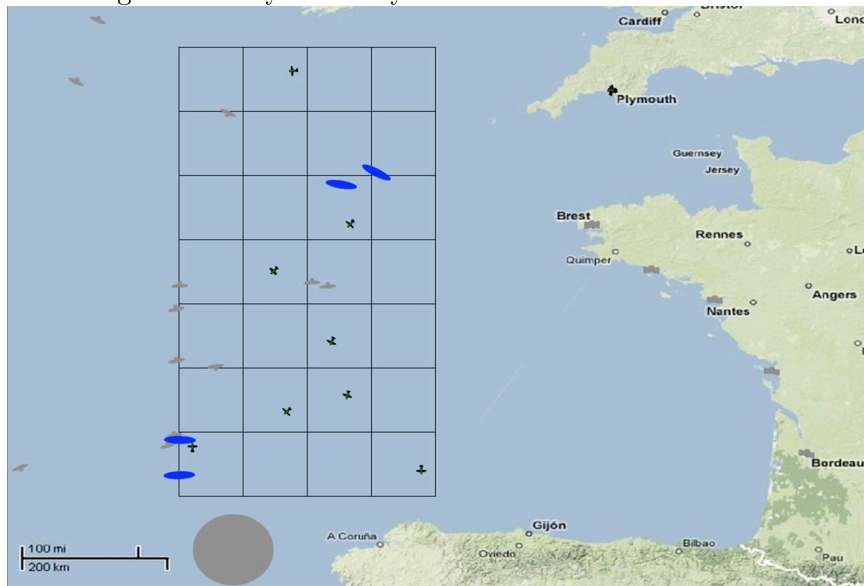
Within the Destroyed block there is only one action block: End Uboat. The End Uboat action shape removes the Uboat from the simulation. It is important to reiterate that while the TSSD Prototype is building a conceptual model, it is also building the foundation of the simulation. Therefore, including actions such as removing the Uboat agent is important to include in the TSSD Prototype.

Inside the Submerged block, there is another level of blocks as well as the different Surface Policy decision shape. This Surface Policy decision shape evaluates when it is time to transition to the On_Surface block based on the surfacing policy set forth in Scenario 1. Once the Uboat has transitioned to the Submerged block it will immediately transition to the internal Moving block. Thus, the Uboat can be crossing the bay, submerged, and moving under water. Within the Moving block there are two action shapes and one decision shape. The Travel action shape is a reoccurring shape that moves the Uboat towards the goal at 2.5 knots. The Deplete Batteries action shape is also a reoccurring shape that depletes the battery charge. The Battery Life decision shape determines if the batteries have been completely depleted and then transitions the Uboat to the Stopped block. Within the Stopped block the Travel action shape moves the Uboat towards the goal at 0 knots.

If at anytime the current location of the Uboat in the environment equal the goal coordinates of the Uboat, then the Uboat in this case will transition to the In_Atlantic block. Within the In_Atlantic block the Set Departure action shape sets the time at which the Uboat will be begin heading toward it's home port. The Depart Port decision shape evaluates whether the current time equals the departure time at which point the Uboat will transition to the Cross_Bay block. Also, it is important to note that the goal coordinates this time will be the Uboat's home port and not the North Atlantic.

Although this description of the Uboat's behavior occurring to the TSSD Prototype has been at a relatively high level, we believe that it has provided most the knowledge needed in order to better understand how to read and interpret the TSSD Prototype. However, much more detail about the Uboat and the entire Bay of Biscay simulation as well as the 'whys' for each behavior is available in the TSSD Prototype File. The ability of the TSSD Prototype to provide both high and low levels

Figure 16: Bay of Biscay ABM Simulation Screen Shot



of details further fills the need for evaluators of various level of expertise to be able to understand the conceptual model that the simulation was based upon. Thus, the TSSD Prototype can be used as a tool for sanctioning and as a tool for verifying that the simulation performs as intended.

6.3 Sanctioning the Reproduced Bay of Biscay ABM Simulation

After constructing the TSSD Prototype of the Bay of Biscay simulation to build and document the conceptual model, the next step was to build the actual Bay of Biscay ABM Simulation. Using the TSSD Prototype as a guide and verification tool, the reproduced Bay of Biscay ABM Simulation was constructed. Figure 16 shows a screen shot of the simulation in action (the dark ellipses are submerged Uboats). From this screen shot, it can be seen that Uboats are crossing the Bay of Biscay and Aircraft are searching 50 NM by 50 NM areas denoted by the squares in the bay. Furthermore, because it is night (indicated by the dark circle at the bottom of the figure) most of the Uboats are surfaced and some are submerged because they have spotted a plane.

Once we verified with the help of the TSSD prototype that the simulation was running as intended the next step was to determine the sanctionability of the simulation. In this case, we are sanctioning our simulation against the results of Champagne's Dissertation and not directly with the real world. As a result of the discussion in the previous chapters, we both conceptually and

Figure 17: Bay of Biscay ABM Simulation Results

	Champ.	Our	Champ.	Our	Champ.	Our
Iteration 1	2	5	108	144	20224	22
Iteration 2	5	2	129	146	21938	22
Iteration 3	3	2	129	139	24760	22
Iteration 4	3	5	150	145	24423	21
Iteration 5	4	4	128	146	22404	21
Iteration 6	2	4	143	147	22432	21
Iteration 7	5	4	147	130	24587	21
Iteration 8	3	2	130	146	24981	22
Iteration 9	5	1	184	138	25045	22
Iteration 10	6	4	168	151	24961	21
Iteration 11	4	1	102	147	19932	22
Iteration 12	3	2	159	123	22729	22
Iteration 13	2	2	107	132	20493	22
Iteration 14	2	4	116	138	21943	22
Iteration 15	4	2	131	150	22408	22
Iteration 16	4	4	120	174	23203	21
Iteration 17	3	4	120	117	25021	22
Iteration 18	5	2	149	132	25100	21
Iteration 19	5	1	156	144	25086	22
Iteration 20	4	4	130	141	22107	22
Mean	3.70	2.95	135.30	141.50	23188.85	2200!
Stan. Dev.	1.22	1.36	21.44	11.81	1766.87	13!

operationally sanctioned our simulation in order to ensure that the entire simulation is sanctionable. To conceptually sanction our simulation, we used the TSSD Prototype to document and describe how we abstracted the conceptual model from Champagne’s Dissertation. For behavior in the TSSD Prototype of the Bay of Biscay, and therefore executed by the simulation, we have included a source that provides the ‘why’ justification for each of those behaviors. For the few behaviors where Champagne’s description was unclear we made sure to document the behavior was based on another reference or was an assumption on our part. Therefore, based on the documentation that is required by and is within the TSSD Prototype we believe that we created a sanctionable conceptual model of the Bay of Biscay simulation that closely mimics Champagne’s 2003 Dissertation.

To operationally sanction our simulation three key statistics were collected and compared to the results published in Champagne’s Dissertation. These statistics were the total number of Uboat kills, the total number of Uboat sightings, and the total number of sortie hours flown at the end of the simulation. A total of 20 replications of the simulation were performed and the results are shown in Figure 17. From this data, several two-sided Two-Sample t-Tests assuming unequal variances

Figure 18: Two-sided Two-Sample t-Test Results

	H_0	H_a	α	p-value	Conclusion
# of Kills	$\mu_1 = \mu_2$	$\mu_1 \neq \mu_2$	0.05	0.074	Fail to reject H_0
# of Sightings	$\mu_1 = \mu_2$	$\mu_1 \neq \mu_2$	0.05	0.266	Fail to reject H_0
# of Sortie Hours	$\mu_1 = \mu_2$	$\mu_1 \neq \mu_2$	0.05	0.008	Reject H_0

were performed at 95% level of significance. The results from these tests are displayed in Figure 18.

The conclusions that can be drawn from the statistical tests are that our reproduced simulation is not significantly different in terms of the number of kills and the number of sightings. However, it is significantly different in terms of the number of sortie hours flown. There are several possible reasons for this discrepancy. The first is that it is unclear exactly how Champagne modeled the Aircraft schedules. In his dissertation he mentions that there are weather delays, however he does not mention the frequency of these delays. Also, there are several different ways one could interpret his description of the Aircraft scheduling procedure and length of flight time. For example, he says that Aircraft searched the Bay until 70% of the fuel was depleted. Based on a few sentences we interpreted this to mean that the Aircraft were searching for 7 hours until they began to return to the base, but one could interpret this aspect of the Aircraft flight time differently.

Another potential reason for this discrepancy is that Champagne does not discuss how he collected the number of sorties hour flown. Different interpretations of when the Aircraft are taking part in searching for Uboats could result in different number of sortie hours flown. This point brings up an unforeseen need that a future version of the TSSD Prototype needs to incorporate and clearly define how measures of performance are captured.

A third potential reason for this discrepancy is the way in which Champagne arrived at getting his number of sortie hours to match the historical results. The number of Aircraft in his simulation was set to 19 because it resulted in his simulation obtaining results that were close to the historical figures. This modeling fitting aspect of his simulation could present further difficulties and complexities to our problem of reproducing his simulation because we are not 100% certain how he modeled every single aspect of the airplane and furthermore we cannot be 100% certain that his given descriptions were accurately being executed by his simulation. In other words, because this aspect

of his simulation was fitted to reality, we have a hard time knowing whether our interpretation of his simulation is wrong or whether his simulation did not execute as he describes since he adjusted the parameters of the model until it fit accordingly. Certainly we are not accusing Champagne of presenting an unverified simulation (in this case Champagne had no choice but to fit the model to match the historical figures because there was no detailed documentation available), instead we are pointing out one of the many troubles that might be encountered when attempting to reproduce a simulation. Reproducing a simulation when the original simulation used model fitting may be a more challenging task than reproducing a simulation that is built using model testing.

Although one of the three measures of operational performance was significantly different, it was one of the least critical measures in terms of the objective of the original simulation which was to evaluate strategies of the Aircraft and Uboats. As a result, we can reasonably say that our simulation is both conceptually and operationally sanctioned. Therefore, we have shown that putting emphasis on developing a conceptual model using the TSSD Prototype can lead to both a conceptually and operationally sanctioned simulation.

7 Concluding Remarks and the Next Steps Towards Improving and Extending the TSSD Prototype

By using the TSSD Prototype to build this simulation many opportunities for improving the tool were identified. Although some of these needed improvements have been alluded too earlier, a compiled list of the next steps and action items to improve the TSSD Prototype is given below:

1. A more complete review of ABM Simulation articles is needed to better understand how the authors build and sanction their simulations. By having this data:
 - (a) A better justification for the need of the TSSD Prototype can be made;
 - (b) The TSSD Prototype can be better geared for what is needed; and
 - (c) Those who use ABM can be better aware of the current status of the field today.
2. A thorough review of modeling tools and techniques in the fields of Systems and Software

Engineering needs to be conducted such that:

- (a) The TSSD Prototype can be better defined within the wider modeling and simulation community; and
- (b) Other modeling techniques and practices can be incorporated into the TSSD Prototype.

3. Several additions to the TSSD Prototype need to be made. Including:

- (a) An additional property for the action and interaction shapes is needed that defines time and frequency of occurrence. While attempting to reproduce the Bay of Biscay simulation, there was a glaring need to better define the timing aspect of the actions within TSSD Prototype for simulation. For example, the notion of travel is easy to conceptually comprehend, but abstracting this continuous action into a discrete action is a critical part of building the simulation. Therefore, adding another property that better defines continuous actions and time is needed.
- (b) An additional shape or property is needed that completely defines the collection of a key statistics used to evaluate the operational effectiveness of the simulation. As discussed earlier, not clearly defining how statistics from the simulation are define can be just as troublesome as not clearly defining behaviors of the simulation.
- (c) A naming or numbering convention for the shapes is needed to better show the hierarchy of the shapes, relationships between them, and to allow for easier identification of unique shapes. Through using and explaining the TSSD Prototype it become clear that a naming or numbering convention such as the one in IDEF0 is needed.
- (d) The representation of the interactions and coordination between agents needs to be improved and better defined. Currently, only a dotted line represents the passing of information between the Aircraft block and the Uboat block (see Figure 8). Also, the coordination of the Aircraft search areas is represented by a dotted block. Whether this is the best way to handle this is debatable, however the reoccurring theme here is that

actions that take place in parallel on the Agent level are not well represented by the TSSD Prototype to date.

4. The execution of the Microsoft Visio 2007 template needs to be reviewed and updated accordingly when changes are made to the TSSD Prototype.
5. The TSSD Prototype needs to be thoroughly tested to ensure that it is capable of building conceptual models of any type of agent-based simulation without the addition of special additions. Furthermore, these tests should be able to show the effectiveness and advantages of using the TSSD Prototype to help build a simulation in terms of documentation and sanctioning.
6. A guide for the using and interpreting the TSSD Prototype needs to be developed to encourage the use of the TSSD Prototype and to provide a baseline documentation of the proper use of the tool.
7. The feasibility of using the TSSD Prototype for more applications other than ABM should be explored. For example, is the TSSD Prototype capable of being used to build generic simulations and/or can the TSSD Prototype be used as a teaching tool to aid students in building better simulations?

Completing these next steps will further improve the TSSD Prototype and the concepts discussed throughout this technical report.

Acknowledgements

The authors wish to thank the support of the Marine Corps Development Command support within their AbVal program under contract with Northrop Grumman. The authors also wish to thank Mr Victor Middleton and Dr. Michael Bailey for their support on this project.

References

- [1] W. Ross Ashby. *An Introduction to Cybernetics*. John Wiley and Sons, Inc., New York, 1956.
- [2] W. Ross Ashby. Analysis of the system to be modeled. In *The Process of Model-Building in the Behavioral Sciences*, chapter 6, pages 94–114. Ohio State University Press, 1999.
- [3] Robert Axelrod. Advancing the art of simulation in the social sciences. *Simulating Social Phenomena*, pages 21–40, 1997.
- [4] Robert Axelrod and Michael D. Cohen. *Harnessing Complexity: Organizational Implications of a Scientific Frontier*. Basic Books, New York, 2000.
- [5] Robert Axtell, Robert Axelrod, Joshua M. Epstein, and Michael D. Cohen. Aligning simulation models: A case study and results. *Computational and Mathematical Organization Theory*, (1):123–141, 1996.
- [6] Osman Balci. Principles and techniques of simulation validation, verification, and testing. In W.R. Lilegdon C. Alexopoulos, K. Kang and D. Goldsman, editors, *Proceedings of the 1995 Winter Simulation Conference*, pages 147–154, 1995.
- [7] Osman Balci. Verification, validation, and accreditation. In J.S. Carson D.J. Medeiros, E.F. Watson and M.S. Manivannan, editors, *Proceedings of the 1998 Winter Simulation Conference*, pages 41–48, 1998.
- [8] Steven C. Banks. Agent-based modeling: A revolution? In *Proceedings of the National Academy of Sciences of the United States of America*, volume 99, pages 7199–7200. Sackler Colloquium on Adaptive Agents, Intelligence, and Emergent Human Organizations: Capturing Complexity through Agent-Based Modeling, May 2002. Supplement 3: Arthur M. Sackler Colloquium of the National Academy of Sciences.
- [9] Jerry Banks, John S. Carson, Barry L. Nelson, and David M. Nicol. *Discrete-Event System Simulation*. Prentice-Hall, Upper Saddle River, NJ, 3rd edition, 2001.
- [10] Yaman Barlas and Stanley Carpenter. Philosophical roots of model validation: two paradigms. *System Dynamics Review*, 6(2):148–166, 1990.
- [11] Grady Booch, James Rumbaugh, and Ivar Jacobson. *The Unified Modeling Language User Guide*. Addison-Wesley, 1999.
- [12] Andrei Borshchev and Alexei Filippov. From system dynamics and discrete event to practical agent based modeling: Reasons, techniques, tools. In *Proceedings of the 22nd International Conference of the System Dynamics Society*, 2004.
- [13] Marcel Boumans. The difference between answering a 'why' question and answering a 'how much' question. In Gunter Koppers Johannes Lenhard and Terry Shinn, editors, *Simulation: Pragmatic Construction of Reality*, volume 25 of *Sociology of the Sciences Yearbook*, chapter 7, pages 107–124. Springer, Dordrecht, The Netherlands, 2006.
- [14] H.J. Bremermann. Optimization through evolution and recombination. In F.T. Jacobi M.C. Yovits and G.D. Goldstein, editors, *Self-Organizing Systems*, chapter 7, pages 93–106. Spartan Books, Washington D.C., 1962.

- [15] John L. Casti. *Complexification: Explaining a Paradoxical World Through the Science of Surprise*. HarperPerennial, 1 edition, 1995.
- [16] Lance E. Champagne. *Development Approaches Coupled with Verification and Validation Methodologies for Agent Based Mission-Level Analytical Combat Simulations*. PhD thesis, Air Force Institute of Technology, Wright-Patterson Air Force Base, OH, November 2003.
- [17] Paul K. Davis. Generalizing concepts and methods of verification, validation, and accreditation for military simulations. Technical report, RAND, Santa Monica, CA, December 1992.
- [18] Paul K. Davis and Donald Blumenthal. The base of sand problem: A white paper on the state of military combat modeling. Technical report, RAND, Santa Monica, CA, 1991.
- [19] Joshua M. Epstein. Agent-based computation models and generative social science. *Complexity*, 4(5):41–60, May 1999.
- [20] Joshua M. Epstein and Robert Axtell. *Growing Artificial Societies: Social Science from the Bottom Up*. Brookings Institution Press, Washington, D.C., 1996.
- [21] Adrew H. Feinstein and Hugh M. Cannon. Constructs of simulation evaluation. *Simulation and Gaming*, 33(4):425–440, December 2002.
- [22] Andrew Hale Feinstein and Hugh M. Cannon. A hermeneutical approach to external validation of simulation models. *Simulation and Gaming*, 34(2):186–197, June 2003.
- [23] Jacques Ferber. *Multi-Agent Systems: An Introduction to Distributed Artificial Intelligence*. Addison-Wesley, Harlow, England, 1999.
- [24] G.S. Fishman and P.J. Kiviat. The statistics of discrete-event simulation. *Simulation*, 10:185–195, 1968.
- [25] Roman Frigg and Julian Reiss. The philosophy of simulation: Hot new issues or same old stew? Forthcoming in *Synthese*, Jan 2008.
- [26] Carlos Gershenson. Complex philosophy. In Pedro Sotolongo, editor, *The First Biennial Seminar on the Philosophical, Methodological, and Epistemologic Implications of Complexity Theory*, La Habana, Cuba, 2002.
- [27] James Gleick. *Chaos: Making a New Science*. Viking, New York, NY, 1987.
- [28] Kurt Gödel. Über formal unentscheidbare sätze der principia mathematica und verwandter systeme. *Monatshefte für Mathematik und Physik*, (38):173–198, 1931.
- [29] David Harel. Statecharts: A visual formalism for complex systems. *Science of Computer Programming*, 8(3):231–274, 1987.
- [30] David Harel. On visual formalisms. *Communications of the ACM*, 31(5):514–530, 1988.
- [31] R.R. Hill, R.G. Carl, and L.E. Champagne. Using agent simulation methods to examine and investigate search theory agaisnt a historical case study. *Journal of Simulation*, 1(1):29–38, 2006.

- [32] R.R Hill, L.E. Champagne, and J.C. Price. Using agent-based simulation and game theory to examine the wwii bay of biscay u-boat campaign. *Journal of Defense Modeling and Simulation*, 1(2):99–109, 2004.
- [33] James S. Hodges. Six or so things you can do with a bad model. *Operations Research*, 39(3):355–365, May 1991.
- [34] John H. Holland. *Hidden Order: How Adaptation Builds Complexity*. Helix Books, Cambridge, MA, 1995.
- [35] J.N. Hooker. Needed: An emprical science of algorithms. *Operations Research*, 42(2):201–212, 1996.
- [36] Don Ihde. Models, models everywhere. In Gunter Koppers Johannes Lenhard and Terry Shinn, editors, *Simulation: Pragmatic Construction of Reality*, volume 25 of *Sociology of the Sciences Yearbook*, chapter 5, pages 79–86. Springer, Dordrecht, The Netherlands, 2006.
- [37] Andrew Ilachinski. Irreducible semi-autonomous adaptive combat (isaac): An artificial-life approach to land warfare. *Military Operations Research*, 5(3):29–47, 2000.
- [38] P.A. Jensen and J.F. Bard. *Operations Research Models and Methods*. John Wiley & Sons, 2003.
- [39] Ann Johnson. The shape of molecules to come. In Gunter Koppers Johannes Lenhard and Terry Shinn, editors, *Simulation: Pragmatic Construction of Reality*, volume 25 of *Sociology of the Sciences Yearbook*, chapter 2, pages 25–39. Springer, Dordrecht, The Netherlands, 2006.
- [40] Harold Kincaid. *Philosophy: Then and Now*, chapter 5, pages 321–338. Blackwell Publishers Ltd, Malden, MA, 1998. Mill and the Nature of Science.
- [41] Esther E. Klein and Paul J. Herskovitz. Philosophical foundations of computer simulation validation. *Simulation & Gaming*, 36(3):303–329, 2005.
- [42] George B. Kleindorfer and Ram Ganeshan. The philosophy of science and validation in simulation. In E.C. Russel W.E. Biles G.W. Evans, M. Mollaghasemi, editor, *Proceedings of the 1993 Winter Simulation Conference*, pages 50–57, 1993.
- [43] George B. Kleindorfer, Liam O’Neill, and Ram Ganeshan. Validation in simulation: Various positions in the philosophy of science. *Management Science*, 44(8):1087–1099, Aug 1998.
- [44] Tarja Knuuttila. From representation to production: Parsers and parsing in language technology. In Gunter Koppers Johannes Lenhard and Terry Shinn, editors, *Simulation: Pragmatic Construction of Reality*, volume 25 of *Sociology of the Sciences Yearbook*, chapter 3, pages 41–55. Springer, Dordrecht, The Netherlands, 2006.
- [45] Alexander Kossiakoff and William N. Sweet. *Systems Engineering: Principles and Practice*. Wiley, 2003.
- [46] Günter Küppers and Johannes Lenhard. Validation of simulation: Patterns in the social and natural sciences. *Journal of Artificial Societies and Social Simulation*, 8(4):3, 2005.

- [47] Gunter Koppers and Johannes Lenhard. From hierarchical to network-like integration: A revolution of modeling style in computer-simulation. In Gunter Koppers Johannes Lenhard and Terry Shinn, editors, *Simulation: Pragmatic Construction of Reality*, volume 25 of *Sociology of the Sciences Yearbook*, chapter 6, pages 89–106. Springer, Dordrecht, The Netherlands, 2006.
- [48] Gunter Koppers, Johannes Lenhard, and Terry Shinn. Computer simulation: Practice, epistemology, and social dynamics. In Gunter Koppers Johannes Lenhard and Terry Shinn, editors, *Simulation: Pragmatic Construction of Reality*, volume 25 of *Sociology of the Sciences Yearbook*, chapter 1, pages 3–22. Springer, Dordrecht, The Netherlands, 2006.
- [49] Christopher G. Langton. Artificial life. In Christopher G. Langton, editor, *Artificial Life*, pages 1–48, Redwood City, CA, 1989. Addison-Wesley Publishing Company, Inc.
- [50] B. Latane. Dynamic social impact: Robust predictions from simple theory. In U. Mueller R. Hegselmann and K. Triotzsch, editors, *Modelling and Simulation in the Social Sciences from the Philosophy of Science Point of View*, chapter 15. Springer-Verlag, New York, NY, 1996.
- [51] Averill M. Law. *Simulation, Modeling, and Analysis*. McGraw-Hill, New York, NY, 4 edition, 2007.
- [52] Steven Levy. *Artificial Life: A Report from the Frontier where Computers Meet Biology*. Vintage Books, New York, NY, 1992.
- [53] Charles M. Macal and Michael J. North. Tutorial on agent-based modeling and simulation part 2: How to model with agents. In J. Liu B.G. Lawson D.M. Nicol L.F. Perrone, F.P. Wieland and R.M. Fujimoto, editors, *Proceedings of the 2006 Winter Simulation Conference*, pages 73–83. Winter Simulation Conference, 2006.
- [54] Brian McCue. *U-Boats in the Bay of Biscay: an essay in Operations Analysis*. National Defense University Press, Washington, DC, 1990.
- [55] David Midgley, Robert Marks, and Dinesh Kunchamwar. Building and assurance of agent-based models: An example and challenge to the field. *Journal of Business Research*, 60:884–893, 2007.
- [56] John H. Miller and Scott E. Page. *Complex Adaptive Systems: An Introduction to Computational Models of Social Life*. Princeton University Press, Princeton, NJ, 2007.
- [57] William T. Morris. On the art of modeling. *Management Science*, 13(12):707–717, 1967.
- [58] Margaret Morrison and Mary S. Morgan. Models as mediating instruments. In Mary S. Morgan and Margaret Morrison, editors, *Models as Mediators*, chapter 2, pages 10–37. Cambridge University Press, Cambridge, 1999.
- [59] Scott Moss and Bruce Edmonds. Sociology and simulation: Statistical and qualitative cross-validation. *American Journal of Sociology*, 110(4):1095–1131, January 2005.
- [60] Thomas H. Naylor and J.M. Finger. Verification of computer simulation models. *Management Science*, 14(2):92–106, October 1967.

- [61] Michael J. North and Charles M. Macal. *Managing Business Complexity: Discovering Strategic Solutions with Agent-Based Modeling and Simulation*. Oxford University Press, New York, NY, 2007.
- [62] James L. Peterson. *Petri Net Theory and the Modeling of Systems*. Prentice-Hall, 1981.
- [63] Michael Pidd. *Tools for Thinking: Modelling in Management Science*, chapter 11, pages 289–312. John Wiley and Sons, 2nd edition, 2003.
- [64] Alan Pritsker. *Handbook of Simulation: Principles, Methodology, Advances, Applications, and Practice*, chapter 2: Principles of Simulation Modeling, pages 31–51. John Wiley & Sons, Inc., 1998.
- [65] L. Resnyansky. Integration of social sciences in terrorism modelling: Issues, problems, and recommendations. Technical report, Australian Government Department of Defence: DSTO Command and Control Division, Edinburgh, Australia, February 2007.
- [66] S. Robinson. Conceptual modelling for simulation part i: definition and requirements. *Journal of Operational Research Society*, 59:278–290, 2008.
- [67] Steward Robinson. Conceptual modeling for simulation: Issues and research requirements. In L.F. Perrone, F.P. Wieland, J. Liu, B.G. Lawson, D.M. Nicol, and R.M. Fujimoto, editors, *Proceedings for the 2006 Winter Simulation Conference*, pages 792–800, 2006.
- [68] Robert G. Sargent. Verification and validation of simulation models. In F. B. Armstrong M. E. Kuhl, N. M. Steiger and J. A. Joines, editors, *The Proceedings of the 2005 Winter Simulation Conference*, pages 130–143, 2005.
- [69] Thomas C. Schelling. *Micromotives and Macrobehavior*. WW Norton and Company, New York, 2 edition, 2006.
- [70] Alex Schmid. What is the truth of simulation? *Journal of Artificial Societies and Social Simulation*, 8(4):5, 2005.
- [71] Terry Shinn. When is simulation a research technology? practices, markets, and lingua franca. In Gunter Koppers Johannes Lenhard and Terry Shinn, editors, *Simulation: Pragmatic Construction of Reality*, volume 25 of *Sociology of the Sciences Yearbook*, chapter 12, pages 187–203. Springer, Dordrecht, The Netherlands, 2006.
- [72] Herbert A. Simon. The architecture of complexity. In *Proceedings of the American Philosophical Society*, volume 106, pages 467–482. American Philosophical Society, American Philosophical Society, December 1962.
- [73] Herbert A. Simon. *The Sciences of the Artificial*. The MIT Press, Cambridge, MA, 3rd edition, 1996.
- [74] Harold Stanislaw. Tests of computer simulation validation: What do they measure? *Simulation and Games*, 17(2):173–191, June 1986.
- [75] Garold Stasser. Computer simulation as a research tool: The discuss model of group decision making. *Journal of Experimental Social Psychology*, 24(5):393–422, September 1988.

- [76] John M. Usher and Lesley Strawderman. Emergent crowd behavior from the microsimulation of individual pedestrians. In J. Fowler and S. Mason, editors, *Proceedings of the 2008 Industrial Engineering Research Conference*, pages 673–678, 2008.
- [77] John von Neumann. *Theory of Self-Reproducing Automata*. University of Illinois Press, Urbana and London, 1966.
- [78] C.H. Waddington. *O.R. In World War 2: Operational research against the U-boat*. Paul Elek (Scientific Books) Ltd, 1973.
- [79] Warren Weaver. Science and complexity. *American Scientists*, 36, 1948.
- [80] Norbert Wiener. *Cybernetics, or control and communication in the animal and the machine*. The MIT Press, Cambridge, MA, 2nd edition, 1962.
- [81] Paul Windrum, Giorgia Fagiolo, and Alessio Moneta. Empirical validation of agent-based models: Alternatives and prospects. *Journal of Artificial Societies and Social Simulation*, 10(2), 2007.
- [82] Eric Winsberg. Sanctioning models: The epistemology of simulation. *Science in Context*, 12(2):275–292, 1999.
- [83] Eric Winsberg. Simulations, models, and theories: Complex physical systems and their representations. *Philosophy of Science*, 68(3):442–454, Sep 2001.
- [84] Eric Winsberg. Simulated experiments: Methodology for a virtual world. *Philosophy of Science*, 70:105–125, January 2003.
- [85] Eric Winsberg. Handshaking your way to the top: Simulation at the nanoscale. In Gunter Kupfers Johannes Lenhard and Terry Shinn, editors, *Simulation: Pragmatic Construction of Reality*, volume 25 of *Sociology of the Sciences Yearbook*, chapter 9, pages 139–151. Springer, Dordrecht, The Netherlands, 2006.
- [86] Eric Winsberg. Models of success versus the success of models: Reliability without truth. *Synthese*, 152:1–19, 2006.
- [87] Stephen Wolfram. *Cellular Automata and Complexity: Collected Papers*. Addison-Wesley Publishing Company, 1994.
- [88] Bernard P. Zeigler, Herbert Praehofer, and Tag Gon Kim. *Theory of Modeling and Simulation*. Academic Press, 2nd edition, 2000.