Detecting Intelligent Agent Behavior with Environment Abstraction in Complex Air Combat Systems

Saurabh Mittal, Margery J. Doyle
L3 Communications, Link Simulation & Training
711/HPW Air Force Research Lab
Wright-Patterson AFB, OH
{Saurabh.Mittal.ctr, Margery.Doyle.ctr}@wpafb.af.mil

Eric Watz
Lumir Research Institute Inc.
711/HPW Air Force Research Lab
Wright-Patterson AFB, OH
Eric.Watz.ctr@wpafb.af.mil

Abstract*—Intelligence can be defined as an emergent property in some types of complex systems and may arise as a result of an agent's interactions with the environment or with other agents either directly or indirectly through changes in the environment. Within this perspective, intelligence takes the form of an 'observer' phenomenon; externally observed at a level higher than that of agents situated in their environment. Such emergent behavior sometimes may be reduced to the fundamental components within the system and its interacting agents and sometimes it is a completely novel behavior involving a new nomenclature. When emergent behavior is reducible to its parts it is considered to be a 'weak' form of emergence and when emergent behavior cannot be reduced to its constituent parts, it is considered to be a 'strong' form of emergence. A differentiating factor between these two forms of emergent phenomena is the usage of emergent outcomes by the agents. In weak emergence there is no causality, while in strong emergence there is causation as a result of actions based on the affordances emergent phenomena support. Modeling a complex air combat system involves modeling agent behavior in a dynamic environment and because humans tend to display strong emergence, the observation of emergent phenomena has to exist within the knowledge boundaries of the domain of interest so as not to warrant any new nomenclature for the computational model at the semantic level. The emergent observed phenomenon has to be semantically tagged as 'intelligent' and such knowledge resides within the bounds of the semantic domain. Therefore, observation and recognition of emergent intelligent behavior has been undertaken by the development and use of an Environment Abstraction (EA) layer that semantically ensures that strong emergence can be modeled within an agent-platform-system, such as Live, Virtual and Constructive (LVC) training in a Distributed Mission Operations (DMO) testbed. In the present study, various modeling architectures capable of modeling/mimicking human type behavior or eliciting an expected response from a human pilot in a training environment are brought to bear at the semantic interoperability level using the EA layer. This article presents a high level description of the agent-platform-system and how formal modeling and simulation approaches such as Discrete Event Systems (DEVS) formalism can be used for modeling complex dynamical systems capturing emergent behavior at various levels of interoperability. The ideas presented in this paper successfully achieve integration at the syntactic level using the Distributed Interactive Simulation (DIS) protocol data units and semantic interoperability with the EA layer.

Keywords— systems integration, semantic interoperability, Discrete Event Systems (DEVS) formalism, Complex systems, Emergence, Complex Air Combat System, Live Virtual Constructive (LVC), Distributed Missions Operations (DMO), Environment Abstraction, Intelligent agent behavior, Artificial intelligence (AI), Modeling and Simulation

I. INTRODUCTION

Sternberg's definition of human intelligence is "(a) mental activity directed toward purposive adaptation to, selection and shaping of, real-world environments relevant to one's life" [1], which means that intelligence is how well an individual deals with environmental changes throughout their lifespan. In some types of complex systems, intelligence can be defined as an emergent property [2]. According to Brooks [3, 4], intelligent behavior can be generated without explicit representations or explicit reasoning; of the kind that a symbolic Artificial Intelligent (AI) agent possess. An implicit assumption in both of these research efforts is that the 'intelligent' behavior arises as a result of an agent's interaction both with its environment and/or agents. We also propose that Sternberg's and Brook's perspective are complementary and an intelligent goal directed behavior is "in the eye of beholder"; an observer phenomenon from a vantage point. To realize both of these definitions in artificial systems, the 'observed' emergent phenomenon has to be semantically tagged 'intelligent' and be available for exploitation by the interacting agents. Emergent phenomenon is a system's behavior, defined by outcomes that cannot be reduced to the behavior of constituent *agent-environment* interactions [5] . Put simply, emergent behavior is detectable only at a level above local interacting agents [6, 7, 8] situated in their environment. This is the case because objects and environments contain perceptual information. Information that indicates the potential actions an object or situation aligned with an active agent's goals and capability affords[9, 10, 11]; i.e., the capability to capitalize on the emergent affordances before them.

In this article, we will present a methodology to formally detect emergent intelligent behavior and capitalize upon the emergent properties/affordances of a dynamic complex system

such as an Air combat system, through Discrete Event System Specification (DEVS) formalism and System Entity Structure (SES) theory [12]. Both the DEVS formalism and SES theory are based on mathematical foundation of set theory. Detection of emergent affordance i.e., properties which allow for intelligent observer agent behavior, is facilitated by Environment Abstraction. The methodology also highlights how the concepts can be applied to both the system design as well as system Test and Evaluation (T&E).

A. Requirements for LVC training in DMO testbed

In most virtual training environments today, many behavior-modeling architectures are pre-defined rule-based, inflexible instantiations that are confined, in large part, by the type of modeling approach it depends upon.

Traditional scripts (i.e., rule-based methods) are generally static and predictable, leading to an inability to scale and adapt to changes in the environment (i.e. die when shot down), or deal with novel situations. In addition, many rule-based scripts contain weaknesses. Weaknesses often 'gamed' by operators who learn very quickly how to exploit and defeat rule-based models, creating a negative learning situation. These problems, which are common even in state-of-the-art game AI technologies, hamper the quality of training human operators can receive.

At the Air Force Research Lab (AFRL), the combat simulation training testbeds run on a Distributed Interactive Simulation (DIS) network. Recently, AFRL initiated a project to evaluate the current generation of commercially available modeling architectures to determine their viability to execute LVC DMO training with accurate, believable, rapidly developed models that measure or mimic human behavior. Through collaboration with industry participants, viz., Aptima, Charles River Analytics, CHI Systems, SoarTech, Alion and Stottler-Henke, the following activities are being pursued that define the scope of the larger problem set and address the integration and interoperability issues required for the use of rapid adaptive believable agents in LVC DMO training environments.

1. Identify criteria to evaluate the current generation of modeling architectures for designing Computer Generated Forces (CGFs).
2. Define data parameters and thresholds to develop an agent model that can be objectively evaluated to determine the behavioral fidelity of such an agent
3. Determine viability and utility for use of these models in adaptive constructive training environments.

While the above defines the overall scope, at the fundamental level, each collaborator is trying to:

1. Specify and develop agent models that mimic human behavior, i.e., render an agent model believable
2. Integrate their modeling architecture with the existing infrastructure at AFRL.
3. Conduct performance evaluation to determine the agent's or modeling architecture's viability for use in training testbeds

This article will describe:

1. The technical architecture and the approach used to achieve semantic interoperability between the collaborators and the AFRL infrastructure in a complex air combat system.
2. The conceptual ideas put forth in [8] towards the development of a configurable system capable of detecting certain emergent properties in an Air Combat System by means of an Environment Abstraction (EA) component. The EA component is geared to achieve complete semantic interoperability at the *agent-platform-system* level.

Ahead, we will describe the technical challenges associated with the development and evaluation of such architectures for use with the existing AFRL infrastructure. Section 2 provides background and an overview on systems theory. It also provides an overview on linguistic levels of interoperability. Section 3 provides the *agent-platform-system* architecture concept that is required for detecting intelligent behavior within the observer paradigm. Section 4 elaborates on the EA component and its various design aspects. It also describes the methodology using DEVS and SES to engineer an EA component. Section 5 concludes the paper.

II. BACKGROUND AND SCOPE

We begin this section with an overview of various definitions so as not to leave any confusion with respect to 'similar' concepts.

A. Working definitions

- **System**: A System is defined as an area of interest with a defined boundary. Logically, this boundary is knowledge-based and technically, it takes the form of an interface. An Interface through which a given system communicates with another system or an environment either directly or indirectly. In component-based modeling paradigm, a component is a representation of a system.
- **Dynamical system**: A system has a state-space and how it moves from its current state to the next state in a given time interval is described formally. Such systems are usually modeled with difference equations, differential equations, or discrete event formalisms [12] .
- **Complex system**: A system that displays emergent behavior from a set of interacting systems/sub-components. In such systems the components are hierarchically organized to manage complexity. The structure and behavioral function of a complex system is *closed under composition* (i.e., a black-box) and can display the complete behavior of its sub-systems taken together.
- **Closed System**: A system defined within a boundary that is impervious to the intake of new knowledge. A complex system can be classified as a closed system: *closed under composition*.
- **Open system**: A system that has no boundary and its behavior cannot be *closed under composition*. New

knowledge can be generated, or synthesized within the system or externally injected such that the system evolves over time. Such a system may have a self-similar nature at different levels of hierarchy.

- **Complex Adaptive Systems** (CAS): A CAS is a complex system constituting a persistent environment inclusive of persistent agents which adapt/learn/evolve. CAS is an open system. Any complex system with a human in-the-loop is a CAS.
- **Complex Air Combat System (CACS):** A complex Air Combat System is a complex dynamical system that displays emergent 'intelligent' (meaningful/purposeful) behavior in an air combat exercise between pilot agents in a dynamic environment. Taken together it is also referred in this article as 'agent-platform-system'. It continues to be a closed system if there is no human-in-the-loop. A Live-Virtual-Constructive (LVC) with a human-in-the-loop system becomes an open system and a CAS.
- **DEVS formalism**: Discrete Event Systems specification (DEVS) formalism is mathematical formalism that formally describes complex dynamical systems and can be used to engineer complex adaptive systems.

B. *Linguistic Levels of Interoperability*

To address the system interoperability challenge in Modeling & Simulation (M&S), three levels of interoperability have been identified and can serve as guidelines to discuss information exchange. Two systems can be configured to work together i.e. they can be *integrated*. In order to interoperate, two systems must have a shared understanding of data at higher levels of abstraction. System integration facilitates interoperability and is the prime driver to achieve interoperability at various levels (Figure 1).

Interoperability occurs at three primary levels: syntactic, semantic and pragmatic

- **Syntactic**: At this level, the data structures are exchanged between two systems without the understanding of meaning of the data. At this level, system integration is realized.
- **Semantic**: At this level, the two integrated systems share a common meaning of the data. The data is transformed into knowledge (i.e. data with context) and new abstractions can be introduced. Technically, data is specified using metamodels and integration occurs at the metamodeling level leading to shared meaning required for semantic interoperability.
- **Pragmatic**: At this level, the integrated systems share a common context and purpose.

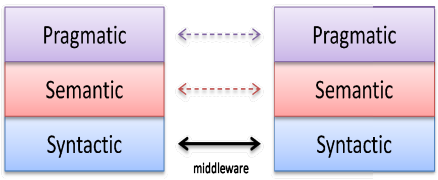

Fig. 1. Interoperability Levels

In order to have interoperability at multiple levels, the modeling and simulation (M&S) conceptual framework also consists of the following layers [12, 13]. Starting from the bottom:

- *Network Layer* contains the actual computers (including workstations and high performance systems) and the connecting networks (both LAN and WAN, their hardware and software) that support all aspects of the M&S lifecycle.
- *Execution Layer* is the software that executes the models in simulation time and/or real time to generate behavior. Included in this layer are the protocols that provide the basis for distributed simulation (such as those that are standardized in the HLA/DIS). Also included in this layer are database management systems, software systems to support control of simulation executions, visualization and visualization/animation of the generated behaviors.
- *Modeling Layer* supports the development of models in formalisms that are independent of any given simulation layer implementation. HLA/DIS provides object-oriented templates for model description aimed at supporting confederations formed by associations between globally dispersed models acting together toward a set of global goals. However, beyond this, the formalisms for model behavior, (whether continuous, discrete, or discrete event in nature) as well as structure change, are also included in this layer. Model construction and especially the key processes of model abstraction and model-continuity over the lifecycle are also included. In addition, ontologies can be defined and added to this layer. Ontologies are understood as models of the world from a particular vantage point for purposes of conceptualization to support information exchange.
- *Design and Search Layer* supports the design of systems and/or system of systems, such as in the Department of Defense Architecture Framework (DoDAF) where the design is based on specifying desired behaviors through models and implementing these behaviors through interconnection of system components. Model-driven Engineering along with formal systems M&S practices is the chosen paradigm that transitions models to real systems [14]. It also includes investigation of large families of alternative models, whether in the form of spaces set up by parameters or by way of a more powerful means by specifying alternate model structures, such as those provided by the Systems Entity Structure (SES) methodology [14, 15]. In addition, AI and simulated natural intelligence (evolutionary programming) may be brought in to help deal with combinatorial explosions occasioned by powerful model synthesizing processes.
- *Decision Layer* applies the capability to search and simulate large model sets at a layer below to make decisions when solving real-world problems. Included are course-of-action planning, selection of design alternatives and other choices where the outcomes may be supported by concept explorations, "what-if" investigations, and optimizations of the models constructed in the modeling

layer using the simulation layer below it. Cognitive decision making rests here.

- *Collaboration Layer* enables people (human agents) or intelligent agents with partial knowledge about a system, whether based on discipline, location, task, responsibility, or specialization, to bring to bear individual perspectives and contributions, often for purposes of to achieving a shared goal. This layer requires the pragmatic use of various cognitive capacities integrated semantically in a multi-agent system leading to group, team and collaborative actions.

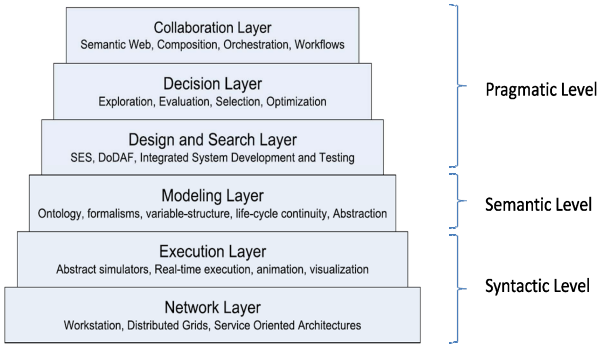

Fig. 2. Architecture for Modeling and Simulation mapped to linguistic levels of interoperability

TABLE 1. INTEROPERABILITY LEVELS AND THEIR APPLICATION.

Interoperability Level	**A collaboration or service interoperate, if:**	**Example**
Pragmatic: How the knowledge is used. Context identification. Purpose	The receiver reacts to the message as the sender intends.	An order from a commander/instructor is obeyed by the pilot in the field as the commander intended. A necessary condition is that the information arrives in a timely manner and that its meaning has been preserved (semantic interoperability at the lower level)
Semantic: Shared understanding of meaning of a message	The receiver assigns the same meaning as the sender did to the message	An order from a commander to multi-national participants in a coalition operation is understood in a common manner despite translation into different languages.
Syntactic: Common rules governing transmission and composition of messages and or information	The receiver is able to parse the message and or information	A common network and transmission protocol (e.g. DIS, TCP/IP) is employed ensuring that all nodes on the network can send and receive data bit arrays adhering to a prescribed format (e.g., PDUs, XML).

As illustrated in Figure 2, at the syntactic level we associate network and execution layers. The semantic level corresponds with the modeling layer – where we have included ontology frameworks as well as dynamic system formalisms as models. Finally, the pragmatic level includes use of the information such as experiment scenarios, objectives and practical use. This use occurs, for example, in design and search, decision making, and collaborating to achieve common goals. Indeed, such computational/ cognitive type activities, along with the associated actions in the world, provide the basis for enumerating pragmatic frames that are potentially of interest, as matched/applied to the context of use.

The resulting stratification leads us to propose the use of the defined linguistic levels, identified in Table 1, for the development of effective systems, or services interoperability. In addition, Figure 2 shows the correlation between the linguistic definitions and the levels of system interoperability.

C. Strong and Weak Emergence for Intelligent behavior

Complex systems are characterized by the presence of an emergent behavior. Complex adaptive systems (CAS) are characterized by usage of properties resulting from emergent system behaviors/outcomes back into the complex systems. The emergent behavior has been classified as *strong* and *weak*, based on the behavior of the agent in the complex system [16]. In *weak* emergence, an agent situated in an environment is simply a reactive agent, not a proactive agent. That is, such an agent has no apparatus to perceive to the utility of emergent behavior, because doing so would require knowledge of new affordances. Affordances can be seen as opportunities for action. They refer to the perceived and actual properties of objects and surrounding environments by animals or humans [17]. While in *weak* emergence affordances brought to bear through emergent phenomena are not available to the agent, in *strong* emergence, a situated agent displays a proactive role by taking advantage of emergent affordances. Affordances that match to an agent's capabilities and limitations, given the goals, will often guide an agent's actions and future goals. Consequently, an agent can act as both an observer of the emergent phenomena while also being imbued with the capability to capitalize on an awareness of what the environment might afford an agent; affordances that may be aligned with an agent's goals. Additionally, in *strong* emergence, the notion of an information boundary becomes critical as the proactive agent, through weak links with other agents or objects in the environment, or its own inherent capability or limitation, continues to evolve, learn and adapt its behavior [8]. Even more critical is the affordance of new information through a global observer that is at a higher level of abstraction than the interacting agent. This new knowledge make the proactive agent more competitive in the same environment, as the agent is more 'aware' of its environment and can process information in a hierarchical way.

In a recent article [8], DEVS levels of system specifications addressed the emergence aspect in CAS. Here we shall implement the conceptual ideas put forth in [8] towards the development of a configurable system capable of detecting certain emergent properties/affordances in an Air Combat System by means of an EA component realized as an Observer Agent (to display *weak* emergence). In later phases of this project, we shall enhance this agent with causal powers (to display *strong* emergence).

Having described the various concepts, let us now look at the agent-platform-system that is capable of displaying the intelligent behavior as a strong emergence phenomenon. The next section describes the technical solution for integrating various modeling architectures including the EA component.

III. AGENT-PLATFORM-SYSTEM ARCHITECTURE WITH ENVIRONMENT ABSTRACTION

The first area of inquiry in this collaborative effort i.e. to create believable behavior in Computer Generated Forces (CGFs) is typically being handled by each collaborator through use of premier, often proprietary architecture or human behavior modeling/mimicking approach. Most of the solutions can be denoted as fairly mature architectures, models, and methods. However, in order to collaborate effectively and have the capacity to evaluate any architecture and the specified agent model, in an adaptive constructive training environment (such as LVC), each collaborator and the evaluator must account for integration and interoperability at three levels (i.e. the syntactic, semantic, and pragmatic). Because the critical path to success and the problem set is first primarily defined as a technical integration task, a task to integrate with DIS and the proprietary AFRL infrastructure, it has been assessed that, currently, almost all collaborators comply or are working to comply with syntactic interoperability by using the DIS Protocol Data Units (PDUs). Additionally, almost all of the collaborators posses or are capable of developing proprietary DIS adapters and have a proprietary, uniquely engineered semantic layer as well. Rendering semantic *interoperability* between the architectures rather questionable, primarily because semantic tokens (shared meanings, terms), required for agent consumption and use, or the domain these models are being applied to, do not currently exist.

Therefore, in order to facilitate semantic interoperability with the AFRL infrastructure, a model to DIS (m2DIS) application programming interface (API) was developed. The m2DIS API provides a set of common terminology/nomenclature. The objective of m2DIS API is twofold. First, it serves as a starting point to introduce semantics and additional abstractions over the DIS PDUs. Second, while each collaborator is free to publish and subscribe to DIS entities through their proprietary middleware, publishing to the threat infrastructure, such as the Network Integrated Control Environment (NICE) over the DIS network mandates the use of the m2DIS publishing API, it the only mechanism by which modelers can leverage the AFRL's LVC DMO testbed, inherently ensuring semantic interoperability at a partial (publish) level.

At the functional level, the m2DIS layer does not restrict the DIS PDUs available to any model, allowing access to all world/situation knowledge (omni-data), data that can be used for an agent's decision-making, in every situation, at every instant. However, access to omni-data could and likely would result in unrealistic behavior by the agent. Therefore, to gain/retain some semblance of realistic behavior, each model has to determine what DIS information is relevant and what information it should act on, acting "as if" it resides within real world's physical and tactical limitations and capabilities. That is to say, if a human pilot cannot normally see, or act as if they can see over-the-horizon than an agent expected to emulate a human type response or a response a human operator can believe, should not act as if it can see over-the-horizon when properties of the environment/agent/situation do not afford such an action. An agent could appear super-human if it utilizes omni-data (world data) rather a limited perspective-view of information for its decision-making and actions. Although more information and knowledge can sometimes be a good thing, having access to, or acting to omni-data most certainly would degrades an agent's realism. To alleviate this problem, we engaged in the development of an EA component, realized as a modular agent system component that limits the amount of semantic information available to any agent, based on the airframe (vehicle platform), LVC environmental context, and the nomenclature/taxonomy of the domain.

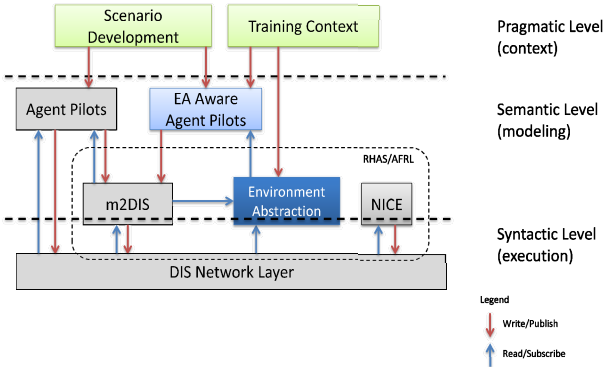

Fig. 3. Technical architecture, integration, and interoperability levels

The EA component is geared to achieve complete semantic interoperability at the agent-platform-system level. Once semantic interoperability is displayed by each modeling team for the current activities, our efforts will then turn towards achieving pragmatic interoperability in a training context. Figure 3 shows the interoperability at the modeling *architecture* level. A knowledge-based agent is architected with pragmatic, semantic, and syntactic levels in terms of knowledge processing for a particular domain. Figure 4 shows various levels of interoperability in an agent-platform system. An agent's perception and action capacity is bounded by the capabilities and limitations of the platform it inhabits (i.e., its body in the Air Combat system), a body that typically is rule-based and obeys the laws of real world physics. The platform (an agents' acting body) in this case, is provided by the NICE infrastructure that allows various types of air vehicles (platforms) to act in this LVC world. Consequently, an agent's perception is bounded by the information that is visible to the agent within this platform's cockpit or through external radio communication with other team members. Similarly, the agent's actions are bounded within domain knowledge and by the actions that can currently be taken through the proprietary AFRL infrastructure.

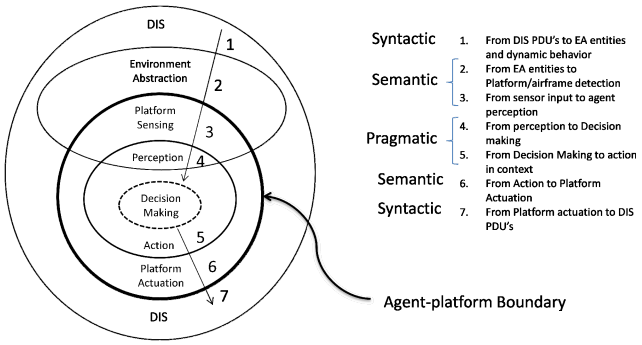

Fig. 4. Agent-Platform architecture and Interoperability levels

IV. Environment Abstraction (EA)

The EA component sits between the platform and the DIS network layer. Based on the structural-functional capabilities, limitations of a platform and its geo-location in a combat exercise, the EA component while aware of the entire combat exercise, makes available only the relevant information to the requesting (subscribing) agent at its platform level. The agent's actions (realized in the AFRL infrastructure) are published by an EA aware agent through the m2DIS API as shown in Figure 3.

The EA component deals with sensing/perception and does not contribute to any actuation capabilities for the specific platform in this phase of the project and will be addressed in the future work. In order to understand the 'sensing' capability of emergent affordances made available through use of the EA component, consider the following example. Assume that a 4 X 4 scenario is in execution. Four blue pilots are flying in formation and in response four agent-based constructive entity red forces start executing a maneuver in a coordinated fashion; responding to blue force tactics. In the current architecture, the entire worldview of information about any actions taken by any entity is available to all other entities via the DIS network. However, this is not realistic, because human agents and or technology based sensors do not typically have the capacity to function in such a manner. Meaning, it would be erroneous to assume that real world agent's (i.e., humans) can see all world data). In minimalist terms, red agents show up on a particular blue agent's radar (visual or sensor based) in a sequential time-based manner so a red force can be 'perceived' through the unfolding scenario or through communication between blue force agents based on their platform's sensing capabilities or their own visuals (i.e., a pilot's line-of-site capacity).

In the proposed architecture, the EA component interfaces between the DIS traffic and the air vehicle platform to limit the information available to any pilot agent. It is worth noting that information is defined as 'semantically rich' data grounded in a domain specific formal ontology and is provided by the EA component. That is, domain-based information about tactics techniques and procedures previously elicited from subject matter experts (SMEs) through a task analysis. This procedure ensures that an agent will appear to behave in a rational manner in response to human/agent actions supporting any collaborating team to freely access and leverage the EA provided semantic to develop pragmatic context for their agents. We shall now look at goals behind the EA conceptualization, various research issues that need to be addressed and requirements that lead the development of the EA component.

A. EA Goals

The EA is designed to achieve the following objectives:

1. Detect emergent affordances supporting emergent 'Intelligent' behavior in a multi-agent system constituting air combat agents. This portrays *weak* emergence.
2. To incorporate the results of the EA component dynamically in an LVC agent to display intelligence and behaviors based on *strong* emergence.
3. To formalize interoperability at syntactic and semantic levels in an event-driven architecture (EDA) performing as a complex adaptive system (CAS).
4. To develop a domain ontology for Air Combat System using System Entity Structure (SES) theory and computationally realized in a formal systems M&S framework such as DEVS.

B. EA Research Issues

In order to develop environment abstraction embodied in an external system (EA component), the following issues have to be formally addressed:

1. Behavior of agents based on their dynamic semantic knowledge.(e.g. role-playing)
2. Mode of communication (direct and indirect) between agents and environment at syntactic, semantic and pragmatic levels
3. Injection/intake of observed emergent properties both at the syntactic and semantic levels into LVC agents
4. Network topology of agents (P2P or hierarchical) and the environment (blackboard, persistent, stigmergic, or P2P)
5. Structural-functional affordances at syntactic, semantic and pragmatic levels of constituent components in an *agent-platform-system* in an LVC setting.
6. Knowledge representation at syntactic, semantic and pragmatic levels using a formal ontology frameworks, grounded in the domain of interest.
7. Integration and interoperability of the observer agent (EA emergent properties) in an LVC system for *strong* emergent behavior.

C. EA Requirements

In order to achieve the goals above, the EA system component (acting as an *observer*) must be able to satisfy the following requirements (adapted from [18]).

1) Operational Requirements

1. Spaciotemporal trajectories: The observer must be able to track various mobile agents and their relative positions in space and time.
2. Levels of abstraction: The observer must organize the information in a hierarchical manner to manage complexity. Goals shall have sub-goals, and tasks

shall have sub-tasks, Actions may have sub-actions and contingencies may have sub-contingencies etc.
3. Structured plan recognition: The observer shall be able to determine what plan is being executed. In the case of incomplete information, the system must converge on a possible set of plans.
4. Team and team-plans: The observer must be cognizant of the team and the allowed team-behavior to better manage complexity and context.
5. Preemptive recognition: The observer shall be able to differentiate possible plans at the earliest possible moment based on the situation in a prioritized manner.
6. Behavioral prediction: The observer shall be able to display predictive validity of the *agent-platform* system
7. Information availability: The observer shall constrain the information that is available to a particular agent based on its field-of-view and airframe capabilities.
8. Information variability: The observer agent in conjunction with the Training Instructor shall be able to control, repertoire of declarative knowledge or information in use, based on allowable actions and the training environment state, to better guide the training process. This is the dynamic knowledge-base with semantic interoperability that adjusts to a pragmatic context.
9. Interoperability: The observer should have both the syntactic and the semantic interoperability with various agents communicating through the DIS network.

2) Technical Requirements

1. Real-time performance: The EA component shall be able to semantically process the DIS network in real-time to effectively display *strong* emergence phenomena. Information that can then be subscribed by a higher level/other agent for purposes of decision making and generating causal actions in the world. The lack of such capability will render a system capable of only displaying *weak* emergence.
2. Formal systems M&S framework: The observer agent shall leverage formal systems modeling and simulation practices according to complex dynamical systems theory as implemented in DEVS formalism to capture emergent properties in the CAS as described in [8]
3. Integration: The observer shall be specified in a component-based modular structure for integration at the syntactic level, to begin with, in a platform independent manner.
4. Interoperability: The observer shall be interoperable at the syntactic level using XML and at the semantic level using Events [14].
5. Netcentricity: The methodology shall be scalable in any data-rich system such as a Service-oriented netcentric system of system.

D. EA Development Methodology

The EA component is a hierarchical component comprised of many sub-components. The design and hierarchical organization of these sub-components is performed as follows:

1. Identify entities, messages and events that constitute inputs and outputs to various entities. Fundamentally, an entity is defined as a *strong-typed* computational data structure that can be physically realized in a real-world component. A message is a type of entity used for data-exchange. An event is realized as a type of message that may have a complex data-type. In addition, an event is defined as a quantized change in the entity state and is marked by either the generation of message or update of entity state. The identification of these tokens/primitives is guided by the semantic domain knowledge and taxonomy developed with the help of SMEs. *These tokens also constitute the emergent behaviors that are detectable*.
2. Develop minimal Input/Output (I/O) pairs that characterize 'state' of a particular entity. Such pairs define various other component 'primitives' at a higher level of abstraction that can be assembled together using DEVS levels of systems specifications [12, 15, 19]. This also paves way for a T&E framework [19].
3. Develop dynamic behavior of DEVS 'observer' agent utilizing these primitives at multiple levels of hierarchy.
4. Develop the domain metamodel of CACS utilizing the constructed primitives with SES theory [14, 15].
5. Instantiate CACS metamodel as a CACS instance with respect to the subscribing pilot agent with respect to its geo-location and air-vehicle platform specifications. In SES parlance, an SES-instance is also called a Pruned Entity Structure (PES) [12, 15, 14].
6. Push CACS event notifications to the pilot agent model making it an EA aware agent.

The above process can be best summarized in Figure 5. The detailed implementation and design will be reported in an extended article.

V. CONCLUSIONS AND LESSONS LEARNED

Intelligent behavior, according to Sternberg is about adapting to one's local environment. According to Brooks, intelligent behavior is an emergent phenomenon and arises as a result of agent's interaction with an environment. In both the cases, there is an implicit assumption that an agent interacts with its environment (that includes other agents) based on its goal structure and available affordances. The fundamental question is a Catch-22 situation: Does the environment provide structural-functional affordance/s for an agent OR is the agent looking-for that affordance to begin with? In the case of emergent affordance, the second question is irrelevant because the agent hits its own knowledge boundary and has no knowledge to recognize new affordance/s. In *weak* emergence these two questions are independent and can co-exist because the knowledge resides in the system and the agent is not looking for emergent affordances. In *strong* emergence, the

emergent affordance has to be recognized first at the semantic level by both the environment and the agent. This semantic recognition at the environment level is being undertaken by the EA component and at the agent level, it is addressed by adding new behavior in response to the newly recognized affordance in an LVC setting (*open* system). This knowledge boundary is enriched in the EA component as it is designed to offer semantic interoperability for strong emergent behavior to occur in a system creating a complex dynamic environment. Further, at the syntactic level for the actual interaction to occur, both the environment and the agent are supplied with the needed syntax in a top-down manner. We addressed new affordance recognition as a systems interoperability problem. An *agent-platform-system* displaying strong emergence semantically interoperates with the environment provided the knowledge about new affordances enriches the existing knowledge-base.

We applied the above concepts to the domain of complex air combat systems and developed primitives towards a metamodel incorporating the semantics of the agent behavior, the platform structure, the emergent phenomena and the situation dynamics. The dynamic interplay of these elements filtered by the agent's geo-location and platform specifications yields an agent with EA aware capability. An agent with such capability is an *affordance- aware* agent capable of displaying *strong* emergence. As humans readily display strong emergence, having such a framework allows development of agents that can produce emergent intelligent and believable behavior at multiple levels of system specifications.

Having this underlying framework for a LVC DMO testbed successfully provides a capability wherein multiple agent architectures seamlessly interoperate at both the syntactic and semantic levels. A detailed implementation methodology and results will be reported in an extended article. Further usage of this infrastructure at the pragmatic level will yield benefits to the pilot's training capabilities.

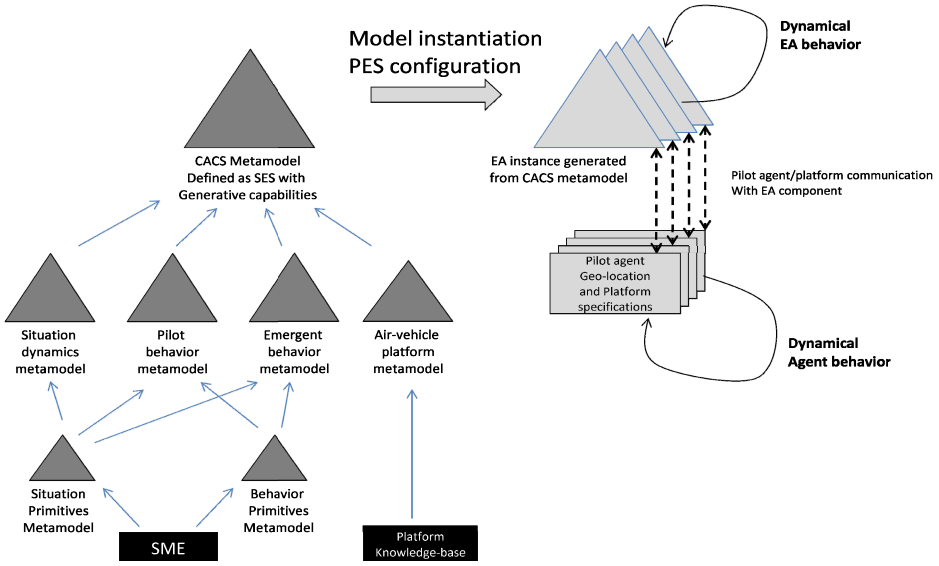

Fig. 5. EA development and Usage methodology

REFERENCES

[1] R. Stemberg, Beyond IQ: A Triarchic Theory of Intelligence, Cambridge: Cambridge University Press, 1985.

[2] R. Brooks, "Elephant's don't play chess," in *Designing Autonomous Agents*, Cambridge, MA, The MIT Press, 1990, pp. 3-15.

[3] R. Brooks, "Intelligence without reason," in *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence*, Sydney, Australia, 1991a.

[4] R. Brooks, "Intelligence without Representation," *Artificial Intelligence,* vol. 47, pp. 139-159, 1991b.

[5] J. Deguet, Y. Demazeau and L. Magnin, "Elements about the emergence issue: A Survery of emergency definitions," *Complexus,* vol. 3, pp. 24-31, 2006.

[6] N. Bass, "Emergence, hiearchies and hyperstructures," *Artificial Life,* vol. 3, 1994.

[7] E. Banabeau and J. Dessalles, "Detection and Emergence," *Intellica,* vol. 2, no. 25, 1997.

[8] S. Mittal, "Emergence in Stigmergic and Complex Adaptive Systems: A Formal Discrete Event Systems perspective," *Journal of Cognitive Systems Research, Special Issue on Stigmergy,* 2012.

[9] N. Dohn, "Affordances-a Merleau-Pontian account," in *Proceedings of the Fifth International Conference on Networked Learning*, Lancaster, 2006.

[10] N. Dohn, "Affordances Revisited: Articulating a Merleau-Pontian View," *International Journal of Computer-Supported Collaborative Learning,* vol. 4, no. 2, pp. 151-170, 2009.

[11] B. Bloomfield and Y. V. T. Latham, "Bodies, Technologies, and Action Possibilities: When is an affordance?," *Sociology,* vol. 44, no. 3, pp. 415-433, 2010.

[12] B. Zeigler, H. Praehofer and T. Kim, Theory of Modeling, New York, NY: Academic Press, 2000.

[13] S. Mittal, B. Zeigler and J. Martin, "Implementation of formal standard for interoperability in M&S/System of Systems integration with DEVS/SOA," *International Command and Control C2 Journal, Special Issue: Modeling and Simulation in Support of Network-Centric*

Approaches and Capabilities, 2008.

[14] S. Mittal and J. Martin, Netcentric System of Systems Engineering with DEVS Unified Process, Boca Raton, FL: CRC Press, 2013.

[15] B. Zeigler and P. Hammonds, Modeling & Simulation-Based Data Engineering: Introducing Pragmatics into Ontologies for Net-centric Information Exchange, New York, NY: Academic Press, 2007.

[16] D. Chalmers, Strong and Weak Emergence: In The Re-emergence of emergence, Oxford: Oxford University Press, 2006.

[17] J. Gibson, The Ecological Approach to Visual Perception, New Jersey: Lawrence Erlbaum Associates, 1986.

[18] C. Heinze, S. Goss and A. Pearce, "Plan Recognition in Military Simulation: Incorporating Machine Learning with Intelligent Agents," in *Proceedings of IJCAI-99 Workshop on Team Behavior and Plan Recognition*, 1999.

[19] E. Mak, S. Mittal, M. Ho and J. Nutaro, "Automated Link 16 Testing using Discrete Event System Specification and Extensive Markup Language," *Journal of Defense Modeling and Simulation,* vol. 7, no. 1, pp. 39-62, 2010.

BIOGRAPHY

SAURABH MITTAL is a Research Scientist with L3 Communications, Link Simulation and Training supporting the Air Force Research Laboratory, Warfighter Readiness Research Division (711 HPW/RHA) at Wright-Patterson Air Force Base, OH. He earned a PhD in 2007 and MS in 2003, both in Electrical and Computer Engineering from the University of Arizona, Tucson. He is an author of a book titled "Netcentric System of Systems Engineering with DEVS Unified Process" and has authored over 30 articles in various journals, conferences and as book chapters. He was the recipient of the US DoD highest civilian contractor recognition 'Golden Eagle' award in 2006 for the project GenetScope. He has over a decade of experience in engineering solutions using formal systems modeling and simulation approaches. His current areas of interest include netcentric systems integration and interoperability, open systems, complex adaptive and dynamical systems with DEVS and SES formalisms, intelligent agents, multi-agent systems and system-of-systems architecture frameworks like DoDAF.

MARGERY J. DOYLE is a Research Consultant, Cognitive Research Scientist, and Engineer with L3 Communications, Link Simulation and Training supporting the Air Force Research Laboratory, Warfighter Readiness Research Division (711 HPW/RHA) at Wright-Patterson Air Force Base, OH. She earned her M.A. in Experimental Psychology with a certificate in Cognitive Psychology from the University of West Florida in 2007. She has also completed PhD level work and research in Cognitive Science. She has recently co-edited a special issue of Cognitive Systems Research on Stigmergy to further the use of stigmergic processes and techniques in areas such as pervasive computing, multi-aircraft control, adaptive systems, and social network analysis. Currently she is using cognitive engineering and developing research methods and systems to support development, evaluation, integration, interoperability, and utility in the use of behavior based agent architectures/models in LVC DMO research training environments.

ERIC WATZ is a Software Engineer with Lumir Research Institute supporting the Air Force Research Laboratory, Warfighter Readiness Research Division (711 HPW/RHA) at Wright-Patterson Air Force Base, OH. He holds an M.S. in Computer Information Systems from the University of Phoenix, and a B.S. in Computer Science from Arizona State University. He has extensive experience developing military simulation and training systems for DMO and LVC environments.