Strengthening OV-6a Semantics with Rule-Based Meta-models in DEVS/DoDAF based Life-cycle Development for Enterprise-Service Architectures

Saurabh Mittal, Amit Mitra, Amar Gupta, Bernard P. Zeigler, *Fellow IEEE*

Abstract — The development of a distributed testing environment would have to comply with recent DoD mandates requiring that the DoD Architectural Framework (DoDAF) be adopted to express high level system and operational requirements and architectures Unfortunately, DoDAF and DoD net-centric mandates pose significant challenges to testing and evaluation since DoDAF specifications must be evaluated to see if they meet requirements and objectives, yet they are not expressed in a form that is amenable to such evaluation. DoDAF is the basis for integrated architectures and provides broad levels of specification related to operational, system, and technical views. In our earlier work, we described an approach to support specification of DoDAF architectures within a development environment based on DEVS (Discrete Event System Specification) for semi-automated construction of the needed simulation models. The result is an enhanced system lifecycle development process that includes both development and testing in an integral manner. We also developed automated model generation using XML which paves the way for OVs to become service-providing components in the Web Services architecture. In this paper we present the semantic structure for one of the Operational View documents OV-6a that would aid the development of these semi-automated models. We will describe how OV-6a can be structured in a more generalized meta-model framework such that every rule is reducible to meaningful code which is automatedly constructed through Natural Language Processing (NLP) methods and further be reduced to DEVS based models. The paper also presents an overview of the Life-cycle development methodology for these enterprise architectures and how a common enterprise domain-model can be used in customized business/domain-specific rules and policy structures**.**

I. INTRODUCTION

The development of a distributed testing environment would have to comply with recent DoD mandates requiring that the DoD Architectural Framework (DoDAF) be adopted to express high level system and operational requirements and architectures [4,5,6,7] Unfortunately, DoDAF and DoD net-centric [8] mandates pose significant challenges to testing and evaluation since DoDAF specifications must be evaluated to see if they meet requirements and objectives, yet they are not expressed in a form that is amenable to such evaluation.

This paper begins by providing an overview of the current DoDAF descriptions and how DEVS is positioned to address the need for a new DoDAF-based and net-centric paradigm for test and evaluation at the system-of-systems and enterprise systems levels. Our earlier work [9] enhanced DoDAF by proposing a methodology to map DoDAF descriptions to DEVS specifications, i.e., DoDAF-to-DEVS mapping. Since DEVS environments, such as DEVSJAVA, DEVS.C++, and others [10] are embedded in object-oriented implementations, they support the goal of representing executable model architectures in an object-oriented representational language. As a mathematical formalism, DEVS, is platform independent, and its implementations adhere to the DEVS protocol so that DEVS models easily translate from one form (e.g., C++) to another (e.g., Java) [11]. DEVS environments are typically open architectures that have been extended to execute on various middleware such as DoD's HLA standard, CORBA, SOAP, and others [12,13,14,15] and can be readily interfaced to other engineering and simulation and modeling tools [16]. Furthermore, DEVS operation over a web-middleware (SOAP) enables it to fully participate in the net-centric environment of the Global Information Grid [8]. As a result of recent advances, DEVS can support model continuity through a simulation-based development and testing life-cycle [17]. This means that the mapping of high-level DoDAF specifications into lower-level DEVS formalizations enables such specifications to be thoroughly tested in virtual simulation environments before being easily and consistently transitioned to operate in real environment for further testing and fielding.

In [18], we proposed extensions to DoDAF by introducing two new Operational View documents, OV-8 and OV-9, that allow modeling and simulation be a critical part in the design process. We demonstrated how DoDAF-DEVS mapping can actually take place from the existing DoDAF UML specifications and how standardized Model Repositories can be created.

The present work aims to refine another DoDAF document, namely OV-6 document. We are particularly focused towards OV-6a specifications that incorporate various rule-based constraints that would allow selective capabilities and multiple designs from a single architecture specified within DoDAF framework. We will demonstrate how the applications of a defined rule-based meta-model provides structure to the current OV-6a document and expedites the construction of semi-automated DEVS models. We propose a DoDAF/DEVS based developmental methodology that includes formal Modeling and Simulation as a part of design, test and evaluation strategy. In addition to this overall development methodology, our focus is to produce a semantically strong OV-6a document that would aid creation

of semi-automated Model development. The procedures that would bring about the translation from a rule-based structure to DEVS Model definitions pave way to creation of run-time models through Natural Language Processing (NLP) methods, as shown in [18].

The next section presents an overview of DoDAF documents, and the Rule-Based Meta-model Framework. Section III describes the integrated developmental methodology using DEVS Testing and Evaluation procedures as a part of design process. Section IV explores the OV-6a semantics with more details on rule-based meta-models and DEVS model constructions. Section V concludes with some discussion on the proposed methodologies and its advantages on the development of Enterprise Architectures

Impact

In an editorial [1]. Carstairs asserts an acute need for a new testing paradigm that could provide answers to several challenges described in a three tier structure. The lowest level, containing the individual systems or programs, does not present a problem. The second tier, consisting of systems of systems in which interoperability is critical, has not been addressed in a systematic manner. The third tier, the enterprise level, where joint and coalition operations are conducted, is even more problematic. Although current test and evaluation (T&E) systems are approaching adequacy for tier two challenges, they are not sufficiently well integrated with defined architectures focusing on interoperability to meet those of tier three. To address mission thread testing at the second and third tiers, Carstairs advocates a Collaborative Distributed Environment (CDE) which is a federation of new and existing facilities from commercial, military and not-for-profit organizations. In such an environment, Modeling and Simulation (M&S) technologies can be exploited to support Model-continuity [2] and Model-Driven Design development [3], making test and evaluation an integral part of the design and operations life-cycle.

The present work employs formal M&S, semantically accurate rule-based structure that is built on an underlying Meta-model, and NLP based methods that would translate these semantic rule structures to automated models, thereby exploiting recent DEVS advancements towards an integrated life cycle development methodology that entails a formal Test and Evaluation strategy for enterprise systems.

II. Background and Earlier Work

A. DoDAF documents (enhanced)

The DoDAF is mandated for expressing high level system and operational requirements and architectures that cross organizational and national boundaries [20]. Its objective is to provide a common denominator of understanding, comparing and integrating these Family of Systems (FoSs), System of Systems (SoSs) and interoperating and interacting architectures. It comprises of 3 major Views:

1) Operational View (OV):
This view provides information on what needs to be accomplished and who should be doing it. It deals with the functional capabilities of the architecture

2) Systems View (SV):
This view provides information on which systems are employed to provide the functionalities expressed in OV. It provides the bridge between the conceptual functionalities and real systems that would provide them.

3) Technical View (TV):
This view provides information on what standards are being used to employ the systems required in SV and what standards are under development to address the future needs of the current architecture.

The interaction between these three views is shown in figure below.

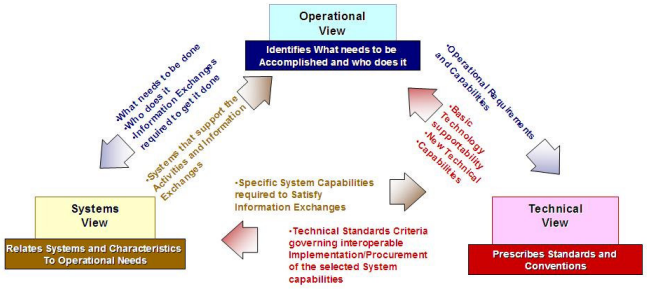

Figure 1: DoDAF Views and their inter-relationships

The primary focus of this paper is within the Operational View documents. Listing all of them, in order of their development sequence:

1) OV-1: Contains the overall functional objective
2) OV-5: Contains the hierarchical functional descriptions of the central capabilities and how different functional elements are integrated in a top-down approach.
3) OV-6: It is further divided into three sub-documents:
    a. OV-6a: Contains the rule-based constraints that would define the boundaries and operational limits.
    b. OV-6b: Contains the sequencing information of various activities listed in OV-5. It also involves decomposing of OV-5 activities into smaller activities. Links various activities to provide a composite 'capability'
    c. OV-6c: Contains information about the statechart (finite state machine) descriptions for any activity/capability.
4) OV-2: Contains the logical Operational node definitions and how different capabilities are grouped together to be performed at one logical node and their mutual connectivity.
5) OV-3: Contains information about various data exchanged that happen between logical nodes in OV-2.

6) OV-7: Contains information about the logical data model developed from OV-2. It inherits the logical connectivity description from OV-2.
7) OV-4: Contains information about the organizational structure (and their associated constraints) of various Operational nodes identified in OV-2 and OV-7.
8) OV-8: Contains information relation to functional capability as 'components' and their interface descriptions needed for component M&S
9) OV-9: Contains information on mapping the Activity components to Operational nodes (defined in OV-2) for functional composablitiy and enhanced M&S.

B. Rule-Based Meta-model Structure

The Rule Meta-model is based on the meta-model of Knowledge as put forward by Dr. Mitra and Dr. Gupta in their work [28]. Figure 2 below shows the layered architecture of Knowledge meta-model. At the top layer is the domain information. The data flows from top-bottom and is analogous to the OSI 7 layer logical structure. The difference here is that the layered meta-models in actuality are rules, classified on the basis of their functionalities and according to their application-domain.

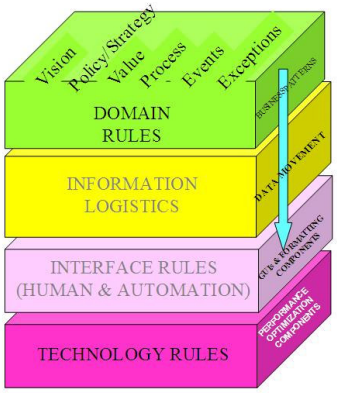

Figure 2: Layered Information Stack in Knowledge Meta-model

The rules in the below 3 layers are common to most of the enterprise architecture designs with little changes but the rules in the topmost layer are truly the rules and constraints that define the performance and behavior of any architecture. They are derived from 'Meaning', a term coined in the Knowledge meta-model that basically signify an abstract term reducible to a logical object capable of some resulting effect as the available domain rules apply to this term. Alternatively, considering a term A having certain meaning, on application of some rule/constraint, transforms to term B with some different meaning in real world. These transformations are also defined at application-domain level and are know as 'Relationship' constraints in this Knowledge meta-model. Consequently, an architecture when reduced to a specific design, ready for being tested (through M&S) or before deployment has set 'terms', meanings and various 'relationships' through which these rules get manifested. The Relationship set along with Meanings may be called upon as jargon of that particular architectural design. This is beneficial for information reuse and component redesign as same logical entities in a generalized architecture can be called upon by different names in dissimilar domains e.g. Business-domain and Military-domain. When such top-down domain-specific rules are applied to any generalized information architecture, the resulting design is application specific and is heavily developed through component reuse.

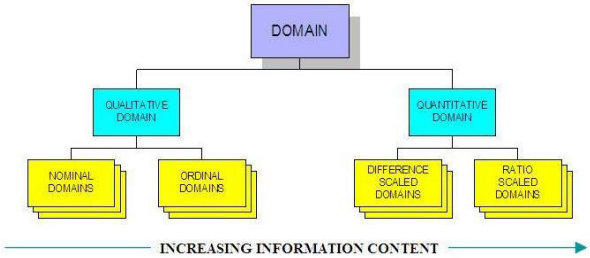

Figure 3: Broad classification of Domain-meaning

Looking deeper into the structure of 'Domain', we have it classified into two broad categories: Qualitative and Quantitative domains. The Domain by itself is actually a domain of Meanings. There are 4 types of domains:

1) Nominal Domains

It contains only classification information. i.e. what category does this Meaning belong to. Are these two meanings the same? They have no information on sequencing, or ratio of properties of objects. This piece of information is expressed in atleast one, but possibly, many formats.

2) Ordinal Domains

It contains both classification and sequencing. They have no information on magnitude of ratios between two meanings. This piece of information is expressed in atleast one, but possibly, many formats and can be used to compare various objects to arrange them in a sequence.

3) Difference Scaled Domains

This domain allows classification in natural sequence based on the measure of their point-to-point differences in the sequence. They carry no information on ratios. It needs atleast one physical format for its expression, know as Unit of Measure (UOM) in Knowledge meta-model. Each UOM must be expressed in atleast one, but possibly several formats.

4) Ratio Scaled Domains

This domain allows classification in natural sequence based on the measure of their point-to-point differences, and takes their ratios. They always have a natural zero.

More details on these domains can be seen in [28].

Mapping to DoDAF Views

This Knowledge meta-model can be very readily mapped to the DoDAF framework. The lowest layer is analogous to Technical View. The second layer from below, Interface rules can be mapped to System Views where different system components have their own presentation and interface definitions. The third layer from below, can be mapped to the

Operational View where logistics and other operational constraints defined how 'Operational nodes' be defined. The top layer is the domain specific rule structure that drives the whole 3-layer set below. As we will see in the next section how application of Domain rules transform a generalized architecture into a specific design, the Knowledge meta-model is very much in line with the basic development methodology of any information related system.

C. *Brief Overview of DEVS M&S Capabilities*

Recent advancement in DEVS technology has enabled the field of M&S to be applied to the system design process. Earlier M&S was viewed as an analysis tool but currently it is very much a part of Design search process. DEVS with its Experimental Frame scenario construction separates the behavior of model against definite controlled conditions thru user interface of Experimental Frame. To provide a brief overview of the current capabilities provided by DEVS, let's look how it could provide solutions to the challenges in net-centric design and evaluation (Table 1).

Desired M&S Capability	**Solutions provided by DEVS technology**
Requirement Coherence and Prioritization	1. Control simulation on-the-fly [23]. 2. Reconfigure simulation on-the-fly [24] 3. Provide dynamic variable-structure component modeling [24][25] 4. Separate model from the act of simulation itself which can be executed on single or multiple distributed platforms [11] 5. Simulation Architecture is layered to accomplish the technology migration or run different technological scenarios [16][26] 6. With its Bifurcated test and development process, automated test generation is integral to this methodology [27] 7. Dynamic simulation tuning, interoperability testing and benchmarking [24]. 8. Provide rapid means of deployment using Model-continuity principles and concepts like 'simulation becomes the reality' [12].
MIL-Worth Analysis (M&S Executable Architectures)	
Enhanced user capabilities	
Execution Roadmaps	
Source Selection	
Technology Application /Transition	
Test Support including Vulnerability analysis	
Interoperability and Integration Assurance	
Hierarchical modular construction of models aiding Systems of system testing	
Provide collaborative distributed environment for M&S	

Table 1: DEVS on addressing M&S issues

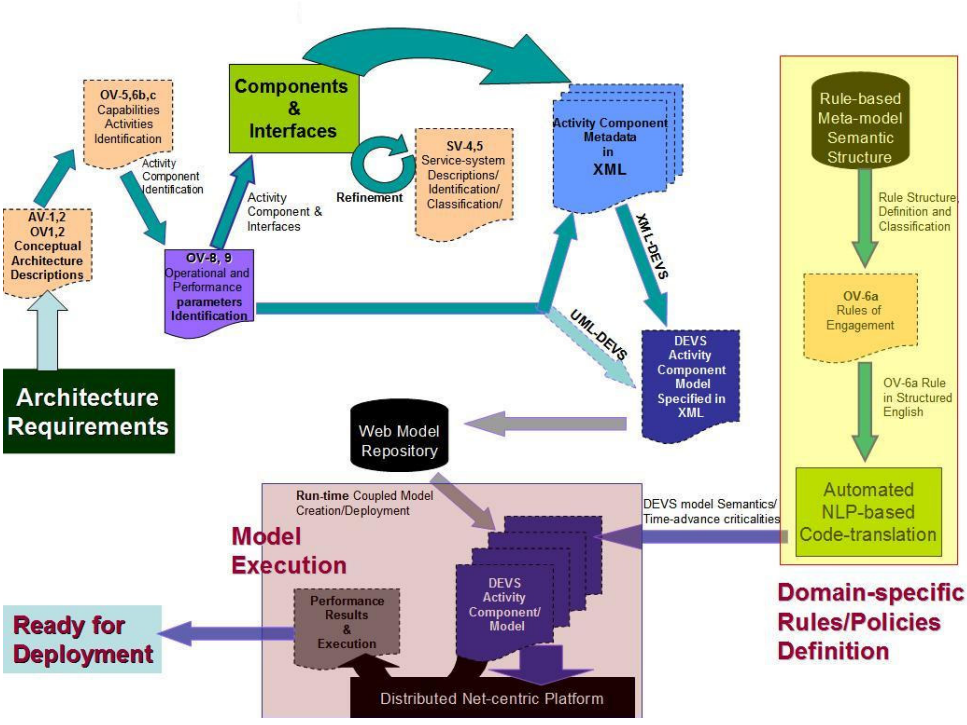

Figure 4: DEVS/DoDAF as the basis for development of Enterprise Architectures incorporating formal M&S

III. INTEGRATED DEVELOPMENT METHODOLOGY

This section presents the integrated design methodology based on DEVS/DoDAF system design principles that incorporates formal M&S test and evaluation procedures. It has 3 major sections. The first is the encoding of information in XML. The second being the development of OV-6a document based on Rule-based meta-model structure. The third being incorporating and merging the above two sections using semi-automated DEVS modeling and distributed simulation. The overall process is shown in Figure 4 above.

The integrated methodology is executed in the following sequential manner:

1. Develop architecture requirements and define DoDAF All View AV-1 and conceptual Operational view OV-1 showing the key capabilities.
2. Define the hierarchical capability functional description document OV-5 and provide more details in OV-6b,c leading to components identification.
3. Develop OV-8 and OV-9 documents that are dedicated to M&S. More details on their development can be seen in [18].
4. Gather component and interface definition information and develop System View SV-4 and SV-5 documents that deals with identification of systems (COTS) that could provide the required capabilities. SV-4 deals with new proposed system identifications. SV-5 deals with COTS. Their identification is continually refined as development to deployment time is extended over long durations.
5. Specify the components, interface, Nodes, and connectivity information from OV-8,9 documents into XML.
6. Put these XML DEVS component models in Web Model Repository
7. Develop OV-6a rules of engagement document description based on underlying meta-model and translated them into meaningful code using NLP methods as done in [19]. More details about this step is provided in Section IV.
8. Gather generalized behavior DEVS model from Web repository and apply the Domain-specific rules/constraints specified in previous step and develop run-time models ready for DEVS distributed simulation, automatedly.
9. Gather performance results (and tune models if need be [9,18]) and transform code to actual system components using Model Continuity principles [12].

IV. DEVELOPMENT OF DOMAIN-SPECIFIC RULES/POLICIES DEFINITIONS

This section presents more details regarding Step 8 of previous section (yellow shaded box in Figure 4). Going further in the details of any Domain term/meaning (Figure 3), we have in Figure 5, another two objects in Knowledge Meta-model known as Unit of Measure (UOM) and Formats. Since this is the topmost level of 'relationships', the lines shown in the diagram are called Meta-relationships, and for the similar reason, these two new objects are called Meta-objects. Figure 5 below shows that a Quantitative domain-meaning is expressed in atleast one or possibly many UOMs (e.g. Difference Scale domains and Ratio Scaled domains). Similarly, Qualitative domains are expressed in atleast one or possibly many formats. Now this figure brings new information in connecting and converting domain related information from one meaning to another:

1. One UOM is expressed in atleast one format or possibly many formats.
2. one UOM converts to none or atmost one UOM
3. one Format converts to none or atmost one Format

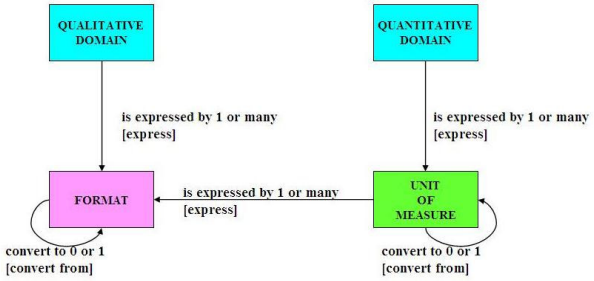

Figure 5: Meta-model of Domain

The inherent formatting information or any 'meanings' measurability is very beneficial in specifying and classifying the behavior of any domain-meaning. Having such underlying framework associated with every 'term' being used in an architecture design aids the automated conversion of various types of Formats and UOMs, if their exists a definition of it, coupled with a domain-meaning. Errors like Mars Rover conversion would not have happened if such Formatting information had been coupled with the meaning of 'Rover speed'. Only the Format was associated with it. If UOM had been associated along with Format, the unit meters/sec could have been automatedly transformed to miles/sec.

Now going further along the yellow box in Figure 4, we arrive now at the OV-6a description of the architecture descriptions. Recall that before we define our OV-6a rules of engagement, we have already developed our OV-5 hierarchical activity descriptions. We have listed numerous activities and how their sequencing occurs in OV-5, OV-6b, and OV-6c. These documents present us with the information on the mechanism of activity happening without constraints or 'security issues' in military domains. In order to develop a semantically accurate OV-6a document, we need to associate various meanings to the repository of domain-meanings as per our

Rule-based meta-model. This association will automatically entail the Meta-objects Format and UOM, removing any ambiguities in representations across any boundaries (national or organizational), which is one of the prime objectives of DoDAF.

Consider a simple OV-6a snippet translated to structured English in the form of pseudo code. This kind of structured English is done manually after understanding the operational and security procedures for any mission undertaking. In this simple example terms like "acceleration rate", "drag effect rate" can be very readily associated with domain-meanings for this particular architecture. Here 'acceleration' is a domain-meaning term, 'rate' is another domain-meaning term, and so is 'drag effect'. Construction of composite meaning like 'acceleration rate' is very well supported in the underlying rule-based Meta-model. Associating them with domain-meanings, ensures their formatting and UOMs, thereby making them semantically consistent and mathematically more accurate.

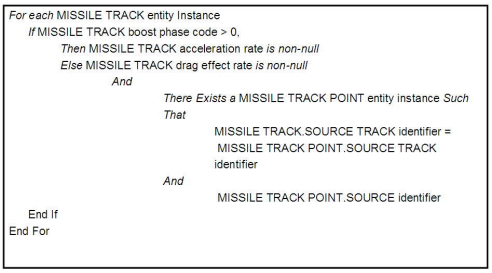

Figure 6: OV-6a pseudo code snippet for Rules of Engagement

Going to the next step involves translation of such pseudo code into dynamic DEVS model specifications. Recent work has been done in this area where structured English is translatable to DEVS models and their internal behaviors being coded thru such pseudo code. More details can be found at [18].

V. DISCUSSION AND CONCLUSION

DoDAF OV-5, 6 capture the functional capabilities of any military system architecture. OV-6a defines the rules and constraints for any mission specific exercise on a generalized architecture. Describing OV-6 documents with an underlying semantic structure, such as Rule-based Meta-model framework, enhances the usability and reuse of defined processes. The information contained therein the OV-6a is exact, semantically consistent and mathematically accurate, if terms are inherently quantifiable. Such defined structures can be used in domains other than military domains as the general mechanisms are well documented in OV-5, 6 documents. The mapping of Knowledge based Metamodel to DoDAF views gives enough evidence that DoDAF is a well constructed information oriented document. However, it is missing a rule-based structure that would allow different architectures to be used for multiple designs. Merging the Knowledge based Meta-model with DoDAF/DEVS based Life cycle development cycle makes DoDAF semantically stronger.

REFERENCES

[1] Carstairs, D.J., "Wanted: A New Test Approach for Military Net-Centric Operations", Guest Editorial, ITEA Journal, Volume 26, Number 3, October 2005

[2] X. Hu, and B.P. Zeigler, " Model Continuity in the Design of Dynamic Distributed Real-Time Systems", accepted by *IEEE Transactions On Systems, Man And Cybernetics— Part A: Systems And Humans*

[3] Wegmann, A., "Strengthening MDA by Drawing from the Living Systems Theory", Workshop in Software Model Engineering, 2002

[4] DoD Architecture Framework, Software Productivity Consortium, http://www.software.org/pub/architecture/dodaf.asp, last accessed Jan 9, 2005.

[5] DOD Instruction 5000.2 "Operation of the Defense Acquisition System," 12 May 2003.

[6] Chairman, JCS Instruction 3170.01D "Joint Capabilities Integration and Development System," 12 March 2004.

[7] Chairman, JCS Instruction 6212.01C "Interoperability and Supportability of Information Technology and National Security Systems," 20 November 2003

[8] K. Atkinson, "Modeling and Simulation Foundation for Capabilities Based Planning", Simulation Interoperability Workshop Spring 2004

[9] B.P. Zeigler, S. Mittal, "Enhancing DoDAF with DEVS-Based System Life-cycle Process", IEEE International Conference on Systems, Man and Cybernetics, Hawaii, October 2005

[10] A. Tolk, S. Solick, "Using the C4ISR Architecture Framework as a Tool to Facilitate V&V for Simulation Systems within the Military Application Domain", Simulation Interoperability Workshop, Spring 2003

[11] B. P Zeigler, H. Praehofer, T. G. Kim, "Theory of Modeling and Simulation", Academic Press, 2000

[12] X. Hu, and B.P. Zeigler, " Model Continuity in the Design of Dynamic Distributed Real-Time Systems", accepted by *IEEE Transactions On Systems, Man And Cybernetics— Part A: Systems And Humans*

[13] Discrete Event Modeling and Simulation Technologies: A Tapestry of Systems and AI-Based Theories and Methodologies Editors: Hessam S. Sarjoughian , François E. Cellier, Spring-Verlag, NY, **2001.**

[14] B. P.Zeigler, DEVS Today: Recent Advances in Discrete Event-based Information Technology, MASCOTS Conference, 2003

[15] http://www.acims.arizona.edu/SOFTWARE/software.shtml, last accessed Jan 12, 2005

[16] H. Sarjoughian, B. Zeigler, and S. Hall, "A Layered Modeling and Simulation Architecture for Agent-Based System Development", Proceedings of the IEEE 89 (2); 201-213, 2001

[17] Cho, Y., B.P. Zeigler, H.S. Sarjoughian, Design and Implementation of Distributed Real-Time DEVS/CORBA, IEEE Sys. Man. Cyber. Conf., Tucson, Oct. 2001.

[18] Mittal.S., "Extending DoDAF to Allow Integrated DEVS-Based Modeling and Simulation", submitted to special issue on DoDAF, Journal of Defense Modeling and Simulation, Oct 2005

[19] Mittal, S., Mak, E., Nutaro, J.J., "DEVS Based dynamic Model Reconfiguration and Simulation control in the Enhanced DoDAF Design Process", submitted to the special issue on DoDAF, Journal of Defense Modeling and Simulation., Feb 2006

[20] DoDAF Deskbook

[21] Bernard P. Zeigler, Herbert Praehofer, Tao G. Kim, "Theory of Modeling and Simulation," Academic Press, 2 Edition, January 2000

[22] Buss, A., Jackson, L. "Distributed Simulation Modeling: A Comparison of CORBA, HLA, and RMI". Proceedings of the 1998 Winter Simulation Conference. 1998.

[23] Mittal, S., B. P. Zeigler, "Dynamic Simulation Control with Queue Visualization", Summer Computer Simulation Conference SCSC'05, Philadelphia, July 2005

[24] Mittal,S., Zeigler, B.P:, Hammonds, P., Veena, M., "Network Simulation Environment for Evaluating and Benchmarking HLA/RTI Experiments", JITC Report, Fort Huachuca, December 2004.

[25] X. Hu, B. P. Zeigler, S. Mittal, “Dynamic Configuration in DEVS Component-based Modeling and Simulation”, SIMULATION: Transactions of SCS International, November 2003
[26] S. Mittal, B. P. Zeigler, “Modeling/Simulation Architectures for Autonomous Computing”, Autonomic Computing Workshop: The Next Era of Computing, January 2003
[27] B. P. Zeigler, D. Fulton, P. Hammonds, J. Nutaro, "Framework for M&S-Based System Development and Testing in Net-centric Environment,” , ITEA Journal, Volume 26, Number 3, October 2005
[28] A. Mitra, A. Gupta, “Agile Systems with Reusable Patterns of Business Knowledge”, Artech House, published 2005