Multi-Level Testing In a Net-Centric Environment

Interoperability or lack thereof, is the characteristic that highlights the fact that systems that perform well individually, typically do not work well when brought together. Inability to exchange information is often at the heart of interoperability problems. For example in geospatial sensing, an enormous number of single purpose "stove-piped" processing chains have emerged in which very little of the available information from all the systems can be fused together. In military and business organizations, lack of interoperability between stove-piped systems prevents rapid, seamless, and collaborative exchange of information. Such stove-piping is supposed to be eliminated by the transition to the Global Information Grid and its Service Oriented Architecture (GIG/SOA), in which services are designed to be accessed without knowledge of their internals through well-defined interfaces that are readily discoverable and composable. However, such a transition is easier said than done. Rigorous testing of GIG/SOA capabilities as they are developed is needed to assure that such capabilities truly support the needs of all DoD agency and system applications. The complexity of GIG/SOA enterprise services under development requires that testing methodology become more rigorous, in-depth and thorough. At the same time, to keep up with the rapid change and short development life cycles expected from the system builders, tests have to be ready to conduct in time scales compatible with the agile development strategies of new systems.

This paper presents a layered architecture for information exchange that blends together existing concepts of Levels of Information Systems Interoperability and Levels of Conceptual Interoperability. We find it useful to employ the distinction between syntactic, semantic, and pragmatic concepts of traditional linguistics to develop a unified structure that can be used to design and test for improved interoperability. Reviewing the linguistic concepts, we have that

- Syntax focuses on a structure and adherence to the rules that govern that structure
- Semantics: Low level semantics focuses on definitions and attributes of terms; high level semantics focuses on the combined meaning of multiple terms
- Pragmatics: considers data use in relation to its structure and the context of application

Based on stratification along the lines of the linguistic levels of pragmatics, semantics and syntax, we discuss an approach to multilevel testing in net-centric environments including that of the SOA over the Global Information Grid (GIG). We show how formal data engineering, modeling and simulation can address the polar requirements of increased rigor and faster test development in a net-centric environment.

Levels of Information Systems Interoperability

Introduced in 1998, the Levels of Information Systems Interoperability, (LISI [LISI]) is a set of models and associated processes based on five levels of interoperability described in Table 1.

Level of Information System Interoperability	**Characteristic**	**Information Exchanges at the level**
Enterprise	Shared data and applications	Global Information Grid (GIG) web-services, service-oriented architecture, advanced collaboration
Domain	Shared data, separate applications	Access to common data bases, sophisticated collaboration
Functional	minimal common functions, separate data and applications	annotated images, maps with overlays
Connected	electronic connection	tactical data links, email, file transfer
Isolated		

Table 1. Levels of Information Systems Interoperability

Levels of Conceptual Interoperability Model

Although the LISI models are used successfully to determine the degree of interoperability between information technology systems, they do not provide a systematic formulation of the underlying properties of information exchange. To remedy this situation, [TOLM] introduced the Levels of Conceptual Interoperability Model (LCIM) in which there are various levels of interoperability between participating systems . The current version of LCIM, outlined in Table 2 was developed to become a bridge between conceptual and technical design for implementation, integration, or federation [TURN, MUGU].

Level of Conceptual Interoperability	**Characteristic**	**Key Condition**
Conceptual	The assumptions and constraints underlying the meaningful abstraction of reality are aligned	Requires that conceptual models be documented based on engineering methods enabling their interpretation and evaluation by other engineers.
Dynamic	Participants are able to comprehend	Requires common understanding of system dynamics

	changes in system state and assumptions and constraints that each is making over time, and are able to take advantage of those changes.	
Pragmatic	Participants are aware of the methods and procedures that each is employing	Requires that the use of the data – or the context of their application – is understood by the participating systems.
Semantic	The meaning of the data is shared	Requires a common information exchange reference model
Syntactic	Introduces a common structure to exchange information,	Requires that a common data format is used
Technical	Data can be exchanged between participants	Requires that a communication protocol exists
Stand alone	No interoperability	

Table 2. Levels of Conceptual Interoperability

The last column lists key conditions that are required to reach an interoperability level from the one below. Of course, the conditions accumulate as the level increases. We note that the conditions given in the LCIM for pragmatic interoperability require that the *use* of data be mutually understood, where the term "use" is interpreted as the context of its application. The definition of the semantic level requires the use of a single reference semantic model as a hub for information exchange among participants in collaboration. However, such a hub and spokes approach, while desirable, is not always feasible[1]. In the stratification to be introduced below, we propose a more streamlined and extended account of information exchange levels.

Linguistic Levels

The authors of LCIM associate the lower layers with the problems of simulation interoperation while the upper layers relate to the problems of reuse and composition of models [HOFF,CHAU]. They conclude "simulation systems are based on models and their assumptions and constraints. If two simulation systems are combined, these

[1] [TURN1, TURN2] evaluated a common information exchange model, C2IEDM, as an interoperability-enabling ontology for command and control. The conclusion is that even if there is room for improvements, the model supports almost all basic needs for such a semantic bridge . [LASS] claim that in its current form, the model is unbalanced in its levels of detail and too large to be practical.

assumptions and constraints must be aligned accordingly to ensure meaningful results."[MUGU]. This suggests that levels of interoperability that have been identified in the area of modeling and simulation (M&S) can serve as guidelines to discussion of information exchange in general. Therefore, we consider an earlier developed conceptual layered architecture for M&S [ZEIG]. We'll correlate the above linguistic definitions with the layers outlined below and shown in Figure 1.

- *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 do the work of supporting 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 their behavior. Included in this layer are the protocols that provide the basis for distributed simulation (such as those that are standardized in the High Level Architecture (HLA). Also included are database management systems, software systems to support control of simulation executions, visualization and animation of the generated behaviors.

- *Modeling Layer* supports the development of models in formalisms that are independent of any given simulation layer implementation. HLA just mentioned also provides object-oriented templates for model description aimed at supporting confederations of globally dispersed models. 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 continuity over the lifecycle are also included. We add ontologies to this layer, where ontologies are understood as models of the world intended to support information exchange.

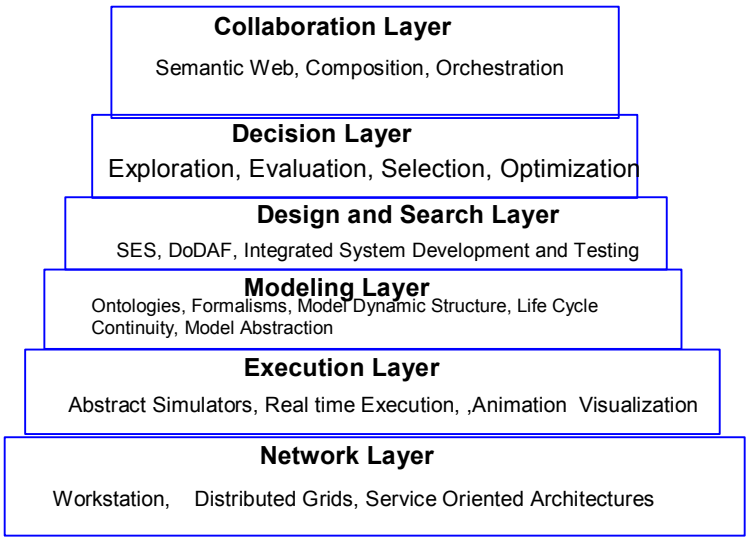

Figure 1 Architecture for Modeling and Simulation

- *Design and Search Layer* supports the design of systems, such as in the DoDAF where the design is based on specifying desired behaviors through models and implementing these behaviors through interconnection of system components. It also includes investigation of large families of alternative models, whether in the form of spaces set up by parameters or more powerful means of specifying alternative model structures such as provided by the SES methodology [COU1, COU2, COU3]. Artificial intelligence and simulated natural intelligence (evolutionary programming) may be brought in to help deal with combinatorial explosions occasioned by powerful model synthesizing capabilities.

- *Decision Layer* applies the capability to search and simulate large model sets at the layer below to make decisions in 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.

- *Collaboration Layer* enables people or intelligent agents with partial knowledge about a system, whether based on discipline, location, task, or responsibility specialization, to bring to bear individual perspectives and contributions to achieve an overall goal.

Using the definitions for linguistic levels above, we correlate such levels with the layers just discussed. 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 identified in the upper layers of the M&S architecture. This use occurs for example, in design and search, making decisions and collaborating to achieve common goals. Indeed, such mental activities, along with real-world physical actions that they lead to, provide the basis for enumerating the kinds of pragmatic frames that might be of interest in particular applications – the context of use.

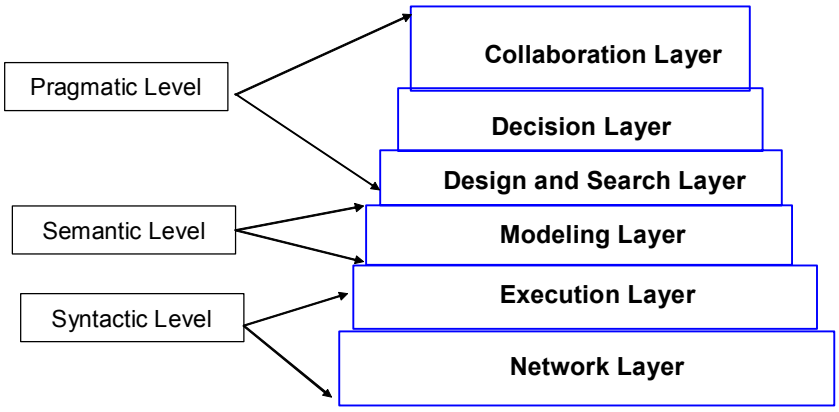

Figure 2 Associating Linguistic Levels with Layers of Modeling and Simulation

The resulting stratification leads us to propose Table 3 for defining effective interoperation of collaborating systems or services at the identified linguistic levels (first and second columns).

Level	**A collaboration of systems or services interoperates at this level if:**
Pragmatic (includes LCIM Dynamic level)	The systems or services use the information received in a manner that is consistent with the shared agreements about the use of information exchanged. Roughly, the receiver re-acts to the message in a manner that the sender intends (assuming non-hostility in the collaboration).
Semantic	The systems or services employ common

(includes LCIM Conceptual level)	ontologies and other models to interpret/generate the messages received/sent, or if not, then translators exist to correctly re-interpret messages on the basis of one ontology/model to another. Roughly, the receiver assigns the same meaning as the sender did to the message.
Syntactic (includes LCIM Technical level)	The systems or services employ common formats and protocols for communicating message data frames, or if not, gateways and adapters exist to correctly map from one protocol to another. For example. they employ TCP/IP, HTTP, and SOAP at successively higher level.

Table 3 Interoperation of services at identified linguistic levels

Note that the stratification employs the traditional three levels of linguistics to represent much of the same ground as the seven levels of the LCIM. This happens because we have given the levels broader interpretation than in traditional linguistics. This allows the stratification to collapse the technical and syntactic levels into the syntactic level, the semantic and conceptual levels into the semantic level, and the pragmatic and dynamic levels into the pragmatic level. More specifically, the claim is that:

- Technical interoperation can be considered syntactic in the sense that rules of one sort or another underlie the operation of computer and network devices, but such rules are blind the larger context in which they operate. They provide the computational and communicational substrate for, and under the command of, higher level activities.

- The conceptual level concerns shared model features such as abstractions and assumptions that can be incorporated in the model-based semantic level as we have defined it.

- The dynamic level, with its concern for evolution over time, refers to the more advanced form of pragmatics called dynamic pragmatics [REF].

Simultaneous Testing at Multiple Levels

Implications for simultaneous testing at the enhanced linguistic levels are shown in Table 4. Here we use the locution “testing that everything is as it should be” as a place holder to be filled in detail in subsequent analysis. The formulation recognizes the fact that we can’t properly test a level without first gaining assurance that everything is as it should be at lower levels. Due to combinatorial complexity, it is extremely difficult to diagnose the source of a deviance at a higher level without first being assured that everything below is

working as it should. We call the approach to testing that takes account of this fact, "testing at all levels simultaneously."

Level	**Testing that everything is as it should be at this level involves:**	**Testing Details**
Pragmatic	1) Establishing that everything is as it should be at the semantic and syntactic levels 2) Examining whether there are shared agreements about the use of information exchanged. 3) Observing whether or not the messages exchanged among the systems or services result in the uses that are specified in the pragmatic agreements.	Mission threads are given for testing of collaboration among participants. Pair-wise translations between participant ontologies or to common information exchange model are evaluated for equivalence within pragmatic frames that are involved in end-to-end mission threads. Test agent are deployed to observe end-to-end mission thread performance and evaluation of measures of effectiveness of information exchanges Tests whether mission threads are executed to completion within allowed timings
Semantic	1) Establishing that everything is as it should be at the syntactic level 2) Examining whether there are common or harmonized ontologies and models to support consistent interpretation and use 3) Observing whether or not the messages	Employ ontological framework tools to assess compatibility of participant ontologies in the context of the mission threads to be executed. Determine whether correct translations exist between participants

	sent/received are interpreted or re-interpreted consistently with the ontologies/models.	ontologies or to a common information exchange models.
Syntactic	1) Examining whether there are common formats and protocols are being used or gateways if needed. 2) Observing whether the computers and communications networks are functioning as they should be.	XML Schema are well-formed XML Schema are valid XML Instance documents are well-formed XML Instance documents are valid wrt to Schema

Table 4 Testing at identified linguistic levels

At the Pragmatic level, mission threads are given for testing of collaboration among participants and to establish use contexts for information exchange. Such mission threads system are intended to capture how effectively the net-centric capabilities are being used in real world situations. Examples of such mission threads are:

- Direct action in an urban environment
- Joint close air support
- Noncombatant evacuation operations
- Force augmentation

We must evaluate the existence, and adequacy, of pragmatic frame agreements about how information that is exchanged will be used in the end-to-end mission threads to be tested. On a pairwise system-to-system basis, we can test information exchange and interoperability. To do this, we must determine whether translations exist between participants' ontologies or to a common information exchange models. If such translations exist, we must ascertain whether they result in pragmatic equivalence within the uses contexts that arise during the course of mission threads execution [REF].

Pair-wise evaluation is followed by test agent-mediated observation and end-to-end evaluation of measures for effectiveness of information exchanges among the systems or services as specified in the pragmatic frame agreements. In a combat context, such measures evaluate how effectively participants can receive role-appropriate command and control information and share a current and accurate understanding of the overall operational situation. In the overall context, we can evaluate

- measures of ability to exchange role-based data-in-context
- measures of specificity to which the information can be routed to intended receivers and no others

The ultimate test is whether mission threads are executed to successful completion within allowed time and resource constraints.

At the Semantic level, we can examine whether there are common or harmonized ontologies and models to support consistent interpretation and use. We can also, observe whether or not the messages sent/received are interpreted or re-interpreted consistently with the ontologies/models.

At the Syntactic level, we can test whether the XML Schema are well-formed, and whether the XML Schema are valid. Likewise, we check that the XML Instance documents are well-formed are valid with respect to Schema.

Building on the use of test agents for web-services [REF], Figure 3 depicts a concept of how such testing might be structured. Test agents at each level interact with the system or services under test at their assigned levels. This implies access to the message traffic and ability to interpret message contents within the assigned level. In addition agents communicate and coordinate with other agents at other levels. This is where the term "simultaneous" applies. For example, an agent that detects an anomaly at the syntactic level should inform colleagues in the semantic level that all is not well at the lower layer. This should cause agents at the semantic level to stop attempting to conduct tests.

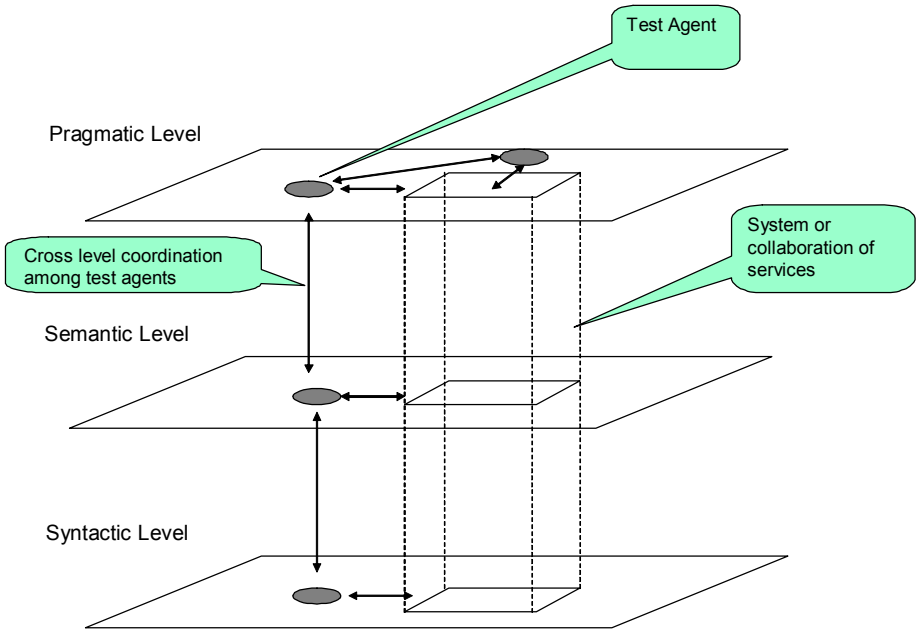

Figure 3 Simultaneous testing at multiple levels

Instrumentation of Testing: Test Federations

The information needed for testing at the various levels can be derived from an architectural system design specification, such as the DoD Architectural Framework (DoDAF). Indeed, in principle, there should be a common specification that drives both development and testing [MITT]. In the following sections, we discuss the implementation of test federations at the pragmatic, semantic, and syntactic levels.

Pragmatic Level – Mission Thread Testing

A test federation observes an orchestration of web-services to verify the message flow among participants adheres to information exchange requirements. A mission thread is a series of activities executed by operational nodes and employing the information processing functions of web-services. Test agents watch messages sent and received by the services that host the participating operational nodes. Depending on the mode of testing, the test architecture may, or may not, have knowledge of the driving mission thread under test. If thread knowledge is available, DEVS test agents can be aware of the current activity of the operational nodes it is observing. This enables it to focus more efficiently on a smaller set of messages that are likely to provide test opportunities. A DEVS distributed federation is a DEVS coupled model whose components reside on different network nodes and whose coupling is implemented through middleware connectivity characteristic of the environment, e.g., SOAP for GIG/SOA, The federation models are executed by DEVS simulator nodes that provide the time and data exchange coordination as specified in the DEVS abstract simulator protocol.

To help automate set-up of the test we use a capability to inter-covert between DEVS and XML. DEVSML allows distributing DEVS models in the form of XML documents to remote nodes where they can be coupled with local service components to compose a federation [MITJ]. The layered middleware architecture capability is shown in Figure 4.

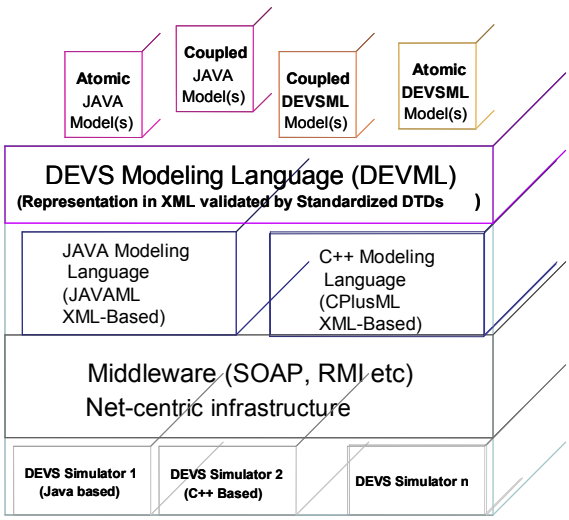

Figure 4. DEVSML layered architecture providing simulator transparency

At the top is the application layer that contains model in DEVS/JAVA or DEVSML. The second layer is the DEVSML layer itself that provides seamless integration, composition and dynamic scenario construction resulting in portable models in DEVSML that are complete in every respect. These DEVSML models can be ported to any remote location using the web-service infrastructure and be executed at any remote location.

The simulation engine is totally transparent to model execution over the net-centric infrastructure. The DEVSML model description files in XML contains meta-data information about its compliance with various simulation 'builds' or versions to provide true interoperability between various simulator engine implementations. Such run-time interoperability provides great advantage when models from different repositories are used to compose models using DEVSML seamless integration capabilities. Finally, the test federation is illustrated in Figure 5 where different models (federates) in DEVSML collaborate for a simulation exercise over GIG/SOA.

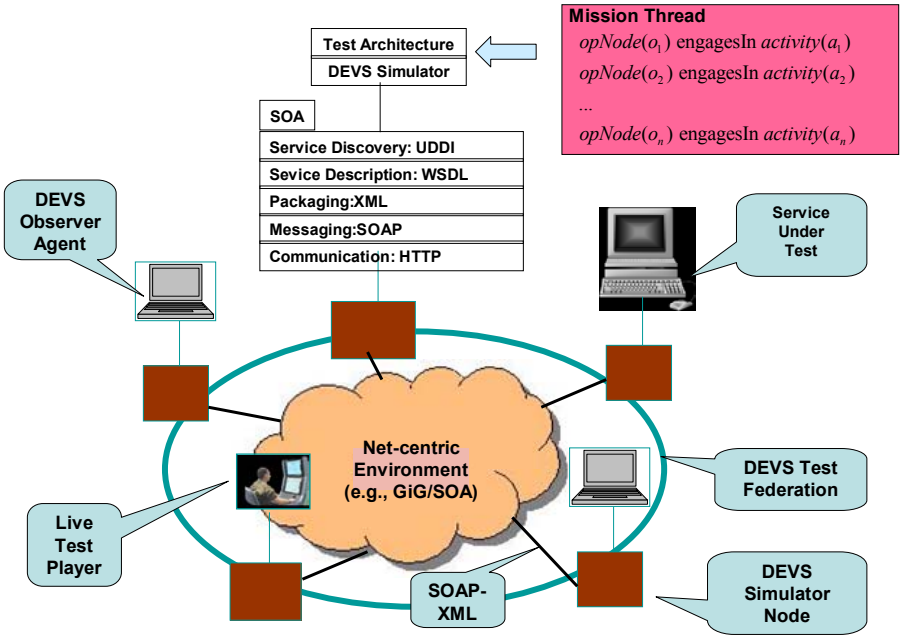

Figure 5. Prototypical DEVS Test Federation

Semantic Level – Information Exchange in Collaborations

Mission threads consist of sequences of discrete information exchanges. A collaboration service supports such exchanges by enabling collaborators to employ a variety of media, such as text, audio, and video, in various combinations. For example, a drawing accompanied by a voice explanation involves both graphical and audio media data. Further, the service supports establishing producer/consumer relationships. For example, the graphical/audio combination might be directed to one or more participants interested in that particular item. From a multilevel perspective, testing of such exchanges involves pragmatic, semantic, and syntactic aspects. From the pragmatic point-of-view, the ultimate worth of an exchange is how well it contributes to the successful and timely completion of a mission thread. From the semantic perspective, the measures of performance involve the speed and accuracy with which an information item, such as a graphical/audio combination, is sent from producer to consumer. Accuracy may be measured by comparing the received item to the sent item using appropriate metrics. For example, is the received graphic/audio combination within an acceptable "distance" from the transmitted combination, where distance might be measured by pixel matching in the case of graphics, and frequency matching in the case of audio. To automate this kind of comparison, metrics must be chosen that are both discriminative and quick to compute. Further, if translation is involved, as discussed above, the "meaning" of the item must be preserved as discussed above. Also, the delay involved in sending an item from sender to receiver, must be within limits set by human psychology and physiology. Such limits are more stringent where exchanges are contingent on immediately prior ones as in a conversation. Instrumentation of such tests is similar to that at the syntactic level to be

discussed next, with the understanding that the complexity of testing for accuracy and speed is of a higher order at the semantic level.

Syntactic Level – Network Health Monitoring

From the syntactic perspective, testing involves assessing whether the infrastructure can support the speed and accuracy needed for higher level exchange of information carried by multimedia data types, individually and in combination. We now consider this as a requirement to continually assess whether the network is sufficiently "healthy" to support the ongoing collaboration.

In a standard concept of experimental frame (EF), the generator sends inputs to the SUT, the transducer collects SUT outputs and develops statistical summaries, and the acceptor monitors SUT observables making decisions continuation or termination of the experiment [REF]. In an EF for real-time evaluation of network health, the SUT is the network infrastructure (OSI layers 1-5) that supports higher session and application layers. As illustrated in Figure 6 a)**, t**he EF is distributed among network nodes, so that each node has an EF consisting of generator, acceptor, and transducer components. The network inputs sent by nodal generators are packets with other nodal EFs as destinations; transducers observe the arrival of packets and the data in their fields to gather transit time and other statistics, providing quality of service (QOS) measurements, while acceptors make decisions on whether the QOS measures are at the levels required for meaningful testing at the higher layers Figure 6 b).

Note that:

- An EF co-exists on a node with a web server or client that is participating in a concurrent test at higher levels, but does not otherwise interact with it.
- The collection of nodal EFs has the objective of assessing the health of the network relative to the QOS that it is providing for the concurrent higher level tests. Thus such a distributed EF is informed by the nature of the concurrent test for which it monitoring network health. For example, if a higher level test involves exchanges of a limited subset of media data types (e.g., text and audio), then the lower layer distributed EF need only monitor the subset of types.

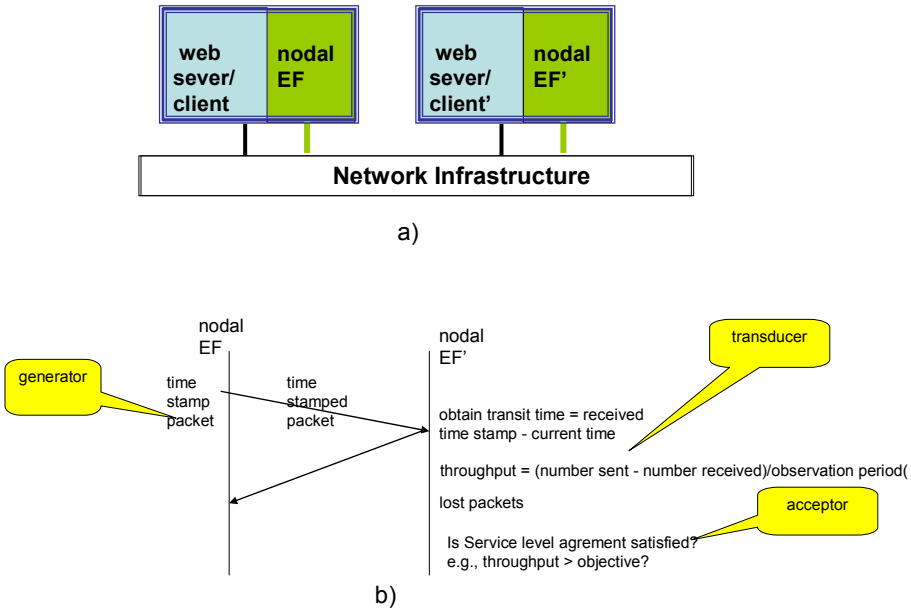

Figure 6 Network Health Monitoring Nodal Experimental Frame Components

Implementation of Distributed Experimental Frames

Since EFs are implemented as DEVS models, distributed EFs are implemented as DEVS models, or agents as we have called them, residing on network nodes. Such a federation, illustrated in Figure 5, consists of DEVS simulators executing as web servers on the nodes exchanging messages and obeying time relationships under the rules contained within their hosted DEVS models. For messages expressed in XML and carried by SOAP middleware such messages are directly generated by the DEVS generators and consumed by the DEVS transducers/acceptors. Such messages experience the network latencies and congestion conditions experienced by messages exchanged by the higher level web servers/clients. Under certain QOS conditions however, video streamed and other data typed packets may experience different conditions than the SOAP-borne messages. For these we need to execute lower layer monitoring under the control of the nodal EFs.

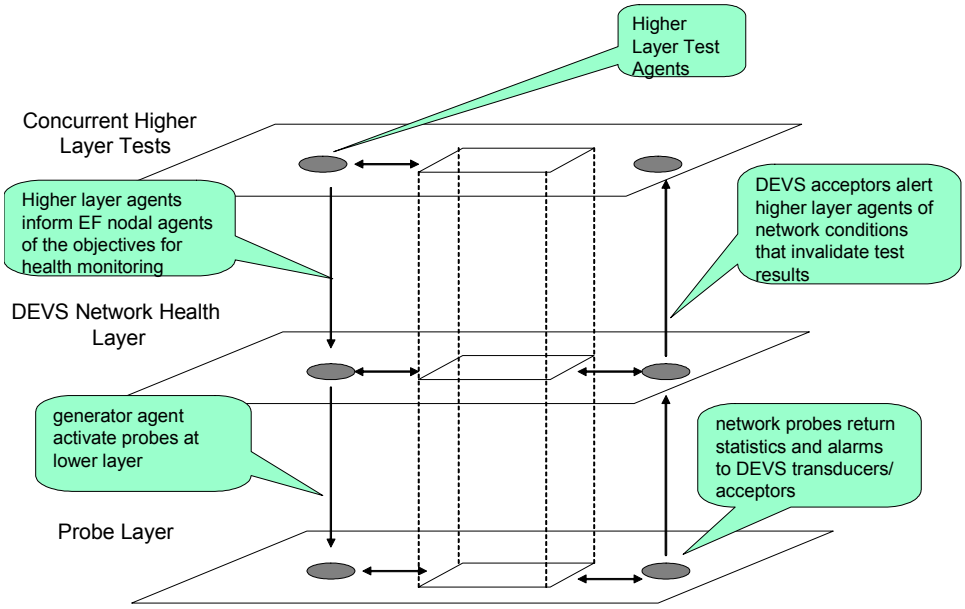

Figure 7 Multi-layer testing with Network Health Monitoring

Figure 7 illustrates the architecture that is implied by the use of subordinate probes. Nodal generator agents activate probes to meet the health monitoring QOS thresholds determined from information supplied by the higher layer test agents, viz., the objectives of the higher layer tests. Probes return statistics and alarm information to the transducers/acceptors at the DEVS health layer which in turn may recommend termination of the experiment at the test layer when QOS thresholds are violated.

Summary

When examined closely, lack of interoperability is not a trivial matter to do away with. Stove piping exists because individual applications gain efficiency by, in effect, employing data structures, models and ontologies whose scope is restricted to support the problem at hand. Trying to interoperate such systems is difficult because their idiosyncratic data assumptions have to be reconciled. Likewise, testing for effective information across such barriers must be done simultaneously at the syntax, semantics and pragmatics levels. At the pragmatics level, we have seen how systems with architectural designs and mission thread specifications can be evaluated using test agents and federations that can be rapidly generated from the design and thread specifications. At the semantics level, measures of performance can be developed for individual exchanges of multi-media data items involve that the speed and accuracy with which such an item is sent from producer to consumer. At the syntactic level, testing involves assessing whether the network infrastructure can support the speed and accuracy needed for the higher level exchanges of information in a collaborative session. We formulated this as a requirement to continually assess whether the network is sufficiently “healthy”

to support the ongoing collaboration. Together, the stratification into the basic levels of pragmatics, semantics, and syntax provides a useful basis for integrating various aspects of interoperability into a coherent, rigorous framework for GIG/SOA development and operational testing.

References

[LISI] Levels of Information Systems Interoperability (LISI), http://www.sei.cmu.edu/isis/guide/introduction/lisi.htm

[HOFF] Hoffmann, M. . Challenges of Model Interoperation in Military Simulations. SIMULATION, Vol. 80, pp. 659-667, 2004

[CHAU] Chaum, E., Hieb, M.R., and Tolk, A. "M&S and the Global Information Grid," Proceedings Interservice/Industry Training, Simulation and Education Conference (I/ITSEC), 2005.

[MUGU] Muguira, James. and Tolk., A "Applying a Methodology to identify Structural Variances in Interoperations," JDMS: The Journal of Defense Modeling and Simulation, Vol 3, No 2, 2006

[TOLM] Tolk, A., and Muguira, J.A. The Levels of Conceptual Interoperability Model (LCIM). Proceedings Fall Simulation Interoperability Workshop, 2003

[TURN] Turnitsa, C., Extending the Levels of Conceptual Interoperability Model. *Proceedings IEEE Summer Computer Simulation Conference,* 2005

[TOLK] Tolk, A.. XML Mediation Services utilizing Model Based Data Management. *Proceedings IEEE Winter Simulation Conference,* pp. 1476-1484, IEEE CS Press, 2004

[TOLD] Tolk, A. and Diallo, S.Y. 2005. Model Based Data Engineering for Web Services. *IEEE Internet* Computing*, Volume 9, Issue 4,* pp. 65-70

[TURN1] [Turnitsa & Tolk 2005] Turnitsa C., and A. Tolk, "Evaluation of the C2IEDM as an Interoperability-Enabling Ontology," Proceedings of Fall Simulation Interoperability Workshop, 2005.

[TURN2] Turnitsa C., S. Kovurri, A. Tolk, L. DeMasi, V. Dobbs and W. P. Sudnikovich, "Lessons Learned from C2IEDM Mappings within XBML," Proceedings of Fall Simulation Interoperability Workshop, 2004.

[LASS] Lasschuyt E., M. van Henken, W. Treurniet, and M. Visser, “How to Make an Effective Information Exchange Data Model,” RTO-IST-042/9,2004

[MITT] Mittal, S. "Extending DoDAF to Allow DEVS-Based Modeling and Simulation", JDMS: The Journal of Defense Modeling and Simulation, Vol 3, No 2, 2006

[MITJ] Mittal, S., Risco-Martín, J.L., Zeigler, B.P., "DEVSML: Automating DEVS Execution Over SOA Towards Transparent Simulators", DEVS Symposium, Spring Simulation Multiconference, Norfork, Virginia, March 2007

[OASD]Office of the Assistant Secretary of Defense (Networks & Information Integration, NII, “Department of Defense Architecture Framework (DODAF),” http://www.defenselink.mil/nii

[KOBR] Kobryn, C., and Sibbald, C., Modeling DoDAF Compliant Architectures: A Telelogic Approach for Complying with the DoD Architecture Framework, http://www.telelogic.com/download/paper/Modeling-DoDAF-WhitePaper.pdf

[DAND1]Dandashi, F., Ang, H.-W., and Bashioum, C. Tailoring DODAF to Support a Service Oriented Architecture. OMG Press, 2004.

[TRBO] Trbovich, S. and Reading, R., Simulation and Software Development for Capabilities Based Warfare: An Analysis of Harmonized Systems Engineering Processes. *Proceedings Spring Simulation Interoperability Workshop,* Press, 2005

[CJCS]. Joint Interoperability Directives & Instructions, CJCSI 6212.01D http://jitc.fhu.disa.mil/jitc_dri/jitc.html

[NECC] The NECC Provisional Technical Transition Architecture Specification, http://www.ditco.disa.mil/News/Documents/File/NECC_ptta_v0_5_7.pdf

[ZEIG] Zeigler, B.P., T.G. Kim, and H. Praehofer.: *Theory of Modeling and Simulation.* 2 ed. 2000, New York, NY: Academic Press

[COU1] Couretas, Jerry M., Bernard P. Zeigler, George V. Mignon, “SEAE-SES enterprise alternative evaluator: design and implementation of a manufacturing enterprise alternative evaluation tool,” Proceedings of SPIE -- Volume 3696, Enabling Technology for Simulation Science III, Alex F. Sisti, Editor, pp. 136-146
1999

[COU2] Couretas, Jerry M., Bernard P. Zeigler, U. Patel, "Automatic Generation of System Entity Structure Alternatives: Application to Initial Manufacturing Facility Design." *Transactions of the SC*S, 1999,16(4), pp. 173-185..

[COU3] [COUR] Couretas, Jerry, System Architectures: Legacy Tools / Methods, DoDAF Descriptions & Design through System Alternative Enumeration, JDMS: The Journal of Defense Modeling and Simulation: Applications, Methodology, Technology, 2006.

[SING]Single Link Interface Reference Specification for Link-16 (JSLIRS -16), 20003.

[SIAP] Representative Measures of a Single Integrated Air Picture (SIAP), Navy Study, 2001,Single Integrated Air Picture (SIAP) Attributes Version 2.0, http://www.stormingmedia.us/55/5570/A557024.html

[ZEIG] Zeigler, B.P., H. Praehofer, and T.G. Kim, 2000, Theory of Modeling and Simulation: Integrating Discrete Event and Continuous Complex Dynamic Systems, 2nd ed., 2000, Academic Press.

[DAHM] Dahmann, J.S., F. Kuhl, and R. Weatherly, *Standards for Simulation: As Simple As Possible But Not Simpler The High Level Architecture For Simulation.* Simulation, 1998. 71(6): p. 378–387.

[MODE]Modeling and Simulation in Manufacturing and Defense Acquisition: Pathwaysto Success (2002), National Academy Press.

[TECH]Technology for the United States Navy and Marine Corps, 2000-2035 Becoming a 21st-Century Force: Volume 9: Modeling and Simulation (1997), National Academy Press.

[ZEIF] Zeigler, B.P.,, Fulton,D., Hammonds P., and Nutaro, J. "Framework for M&S–Based System Development and Testing In a Net-Centric Environment", ITEA Journal of Test and Evaluation, Vol. 26, No. 3, 21-34, 2005

[NUTA] Nutaro, James and Phil Hammonds. "Combining the Model/View/Control Design Pattern with the DEVS Formalism to Achieve Rigor and Reusability in Distributed Simulation".Journal of Defense Modeling and Simulation: Applications, Methodology, Technology,pp. 19-28, Vol. 1, No. 1, 2004.

[MAKE] Mak, E, "Automated Testing Using XML And DEVS," MS Thesis, Department Of Electrical And Computer Engineering, The University Of Arizona, May, 2006.

[MANE] Manes, A.T. 2005, *VantagePoint 2005-2006 SOA Reality Check Version*: 1.0, Burton Group Publication, Jun 29

[HULL] Hull, R. and J. Su, "Tools for Composite Web Services: A Short Overview", ACM SIGMOD Record, Vol 34, No. 2, June, 2005

[GRUN] Grundy, J.C., Ding, G., and Hosking, J.G. Deployed Software Component Testing using Dynamic Validation Agents, Journal of Systems and Software: Special Issue on Automated Component-based Software Engineering, vol. 74, no. 1, January 2005, Elsevier, pp. 5-14

[YUKU] Yu Qi, David Kung, Eric Wong, "An Agent-Based Testing Approach for Web Applications," *compsac*, pp. 45-50, 29th Annual International Computer Software and Applications Conference (COMPSAC'05) Volume 2, 2005

[HUHO] Huo, Qingning, Hong Zhu, Sue Greenwood, "A Multi-Agent Software Environment for Testing Web-based Applications," compsac, p. 210, 27th Annual International Computer Software and Applications Conference, 2003

[HONG] Hong Zhu, "Cooperative Agent Approach to Quality Assurance and Testing Web Software," *compsac*, pp. 110-113, 28th Annual International Computer Software and Applications Conference - Workshops and Fast Abstracts - (COMPSAC'04), 2004

[WILL] Willmott, Steven ,Simon Thompson, David Bonnefoy, Patricia Charlton, Ion Constantinescu, Jonathan Dale, Tianning Zhang, "Agent Based Dynamic Service Synthesis in Large-Scale Open Environments: Experiences from the Agentcities Testbed," *aamas*, pp. 1318-1319, Third International Joint Conference on Autonomous Agents and Multiagent Systems - Volume 3 (AAMAS'04), 2004

[FOSF] Foster, I., Frey, J., Graham, S., Tuecke, S., Czajkowski, K., Ferguson, D., Leymann, F., Nally, M., Storey, T. and Weerawaranna, S. *Modeling Stateful Resources with Web Services*. Globus Alliance, 2004

[FOSJ] Foster, N. R. Jennings, and C. Kesselman. Brain meets brawn: Why grid and agents need each other. Autonomous Agents and Multi-Agent Systems (AAMAS'04), 2004

[HUZE] Hu, X.,B. P. Zeigler, and S. Mittal, "Variable Structure in DEVS Component-Based Modeling and Simulation", *SIMULATION: Transactions of The Society for Modeling and Simulation International*, Vol. 81, No. 2, pp. 91-102, 2005