M&S–ENABLED TESTING OF DISTRIBUTED SYSTEMS: BEYOND INTEROPERABILITY TO COMBAT EFFECTIVENESS ASSESSMENT

Bernard P. Zeigler, Ph.D., Dale Fulton, Jim Nutaro, Phil Hammonds, Ph.D.
Northrop Grumman Information Technology Team
Joint Interoperability Test Command
Fort Huachuca, AZ 85613-7051
zeiglerb@fhu.disa.mil

ABSTRACT:
Department of Defense acquisition policy requires testing throughout the systems development process to ensure not only technical certification but also combat effectiveness. Complexity within each new system, as well as composition into families of systems and systems of systems, combines with the extensive use of simulation in the design phase to multiply the challenges over traditional interoperability methodologies and processes. Existing single-thread conformance certification processes, such as those conducted at the Joint Interoperability Test Command, provide a solid foundation on which to build a more robust framework for distributed interoperability testing from a holistic perspective. The framework recognizes that system development must increasingly rely on modeling and simulation (M&S) while addressing some critical impediments to the efficient use of M&S in distributed testing environments. We discuss the framework and its theoretical and practical formulation, offering an approach to the robust development of an M&S infrastructure for distributed testing. Supporting education for this approach is also described, which will ensure success within a traditional test organization.

1. Introduction

Interoperability testing, such as that conducted by the Joint Interoperability Test Command (JITC), employs a structured process in two testing categories (standards conformance and interoperability) across two different environments (synthetic and live). The process examines the entire range of systems, from single interface systems to highly complex multiple systems of systems with multiple interfaces. Like the processes of many test organizations, however, this process is not comprehensive enough to enable an overall combat effectiveness assessment that reaches beyond the individual performance of a proposed system to include its interoperability within joint systems of systems configurations. In this paper, we discuss a framework, based on modeling and simulation formalism, being introduced in JITC's context to evolve toward greater combat effectiveness testing capability. Other test organizations might adopt this formal approach in a similar way, tailoring it to their mission and long-range goals.

Cost-effective development of today's complex systems must largely, if not exclusively, rely upon modeling and simulation (M&S) principles, methodologies, and technologies. As detailed in recent Department of Defense (DOD)-sponsored reports [1,2], M&S, when properly performed through various stages of the design lifecycle, can provide effective assistance in formulating a system's capabilities, predicting and comparing the cost/benefit ratios of its various alternative architectures and evaluating its projected combat effectiveness. However, currently, there is no comprehensive approach for developing and executing simulation technology for the downstream stages of the lifecycle, including interoperability testing.

The primary objective of this paper is to propose a framework for distributed interoperability testing that can expand the perspective to support a combat effectiveness assessment. Starting from this primary objective, the paper will address issues in the following related areas:

- Identify the underlying cause that impedes the development of reusable infrastructure—the absence of a theory and formalism for M&S that, among other things, supports separation of experimentation, models, and simulators.
- Propose a solution based on the Discrete Event System Specification (DEVS) formalism and underlying dynamic systems theory, which characterizes experimental frames as distinct simulation artifacts and supports their composition with models and simulators in a modular and reusable fashion.
- Discuss how this solution, particularly the development of experimental frames, applies to testing in the interoperability Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance (C4ISR) combat effectiveness testing context.
- Offer, as an example of the solution, a discussion of the Single Link Interface Reference Specification (SLIRS) for Link-16 [8].
- Discuss a spiral methodology for the development of a test infrastructure based on the proposed approach.

- Describe training to support M&S integration into the testing process.

2. Current Methodology Assessment

Steadily increasing requirements for employing M&S in test and evaluation have stimulated the introduction of a variety of vendor-based and government-developed simulation packages. Benefits from this infusion have been unmistakable. However, as the proliferation of such simulation systems increases, pressures to reuse acquired ones inevitably follow as a consequence. Unfortunately, reuse of most of today's packages is fraught with peril since one can all too easily misapply them to new objectives that they were not intended to fulfill. In such packages, the experimentation specifications, models, and simulations are tightly coupled, even when they are conceived as objects at the software engineering level. Proprietary restrictions on access to the internals of the software aside, one would have difficulty in separating out the three key functions found in simulation software, namely, model, simulator, and experimentation specifications. This lack of transparency and modularity makes it extremely unlikely that potential users will be able to match their desired functionality with that of the off-the-shelf package in each of the three dimensions. The application of middleware, such as High Level Architecture (HLA)[4], which supports interoperating such packages, does not solve this problem; in fact, it might make it worse. It is easy to be misled into thinking that ability to exchange data, however well supported, guarantees that the data exchanges make sense for the application at hand.

The HLA Federation Development and Execution Process (FEDEP) model proposes six steps in an "iterative waterfall software process" to support development of HLA-compliant simulations. The intent of HLA is to enable the construction of compositions, called federations, of existing simulations, called federates. Although HLA and the FEDEP process provide a protocol and a methodology for such construction, the primary emphasis is on creating the right interfaces for the sharing of data among federates, rather than on assuring that the federates interact functionally as models or as test environments in the intended manner. A major problem, as suggested earlier, is that neither the HLA nor FEDEP can assure that the encapsulated models, simulators, and experimentation constituents of federates match up properly for the intended application.

Figure 1 a) illustrates the inherent difficulties encountered when trying to develop testing infrastructures in the absence of separation of experimentation, model development, and simulation execution concerns. Even assuming the most advanced middleware, such as Distributed Interactive Simulation (DIS), HLA, Common Object Request Broker Architecture (CORBA), or Test and Training Enabling Architecture (TENA), incompatibilities at any of the three levels will make it difficult for monolithic boxes in which they reside to play meaningfully with each other. Shown in the figure are the *models, simulators,* and *experimental frames*, as distinct entities within a distributed configuration for testing, simulation, and other interactions. These are described in the next section on formal methodology background.

Example: *Using interface testers for correlation algorithm testing*. The Dual Link System (DLS) [5] is a software package for testing communication interfaces to mission computers, correctness of track message parsing, and workload management. This purpose can be expressed in an underlying experimental frame that expresses the conversion of radar track inputs into messages in formats compliant with several mission computer standards. The models embedded in DLS include *dead reckoning algorithms* to correct sampled track information and a *coordinate transformation* to convert global to local coordinates.

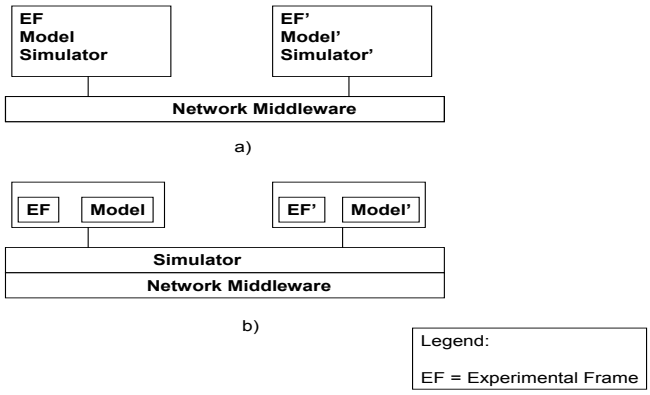

Figure 1. Separation of Frame, Model, and Simulator

The simulator can be taken as the C code in which the system is written. DLS is DIS compliant. Nevertheless, its use as a federate whose purpose is to inter-convert between tracks and messages within a platform for testing mission computer correlation algorithms is problematic. For this purpose, the dead reckoning and coordinate transformation models may **not** be needed and can even cause distortions of the tracks that the experiment cannot control. Thus, the DLS experimental frame and included models are incompatible with the frame and models of the correlation algorithm test frame, despite the fact that DIS compliance allows it to be included as a federate.

The interfacing capability provided by such middleware as DIS, HLA, and CORBA, therefore, adheres to standards but provides insufficient conditions for meaningful federation development. A more encompassing solution is needed to develop infrastructures for M&S-based testing of distributed systems.

Moreover, the conventional approach to simulation-based testing does not allow development of experimentation *commodities* in both component and composite units to enable more automated and reusable test generators and the application of more powerful analytic, summarization, and visualization capabilities. For example, while libraries of statistical subroutines are common, specific combinations of such routines required for particular tests, with input generators and visualization packages, must be developed anew for each occasion.

Within the testing context, therefore, we conclude there are two critical impediments to the efficient application of M&S–to–DOD testing environments: (1) lack of a sufficiently evolved infrastructure to support M&S-based testing of distributed systems, and (2) lack of trained personnel who are proficient in the knowledge and skills needed for efficient use of such infrastructure.

The following section provides a brief background to introduce concepts and an approach that will explore the current problem areas more closely and propose system-level comprehensive solutions.

3. Background: Framework for Modeling and Simulation

The *Framework for M&S* as described in *Theory of Modeling and Simulation* [3], establishes *entities* and their *relationships* that are central to the M&S enterprise (see figure 2). The entities of the framework are *source system, experimental frame, model,* and *simulator;* they are linked by the *modeling* and the *simulation* relationships. Each entity is formally characterized as a system at an appropriate level of specification within a generic dynamic system.

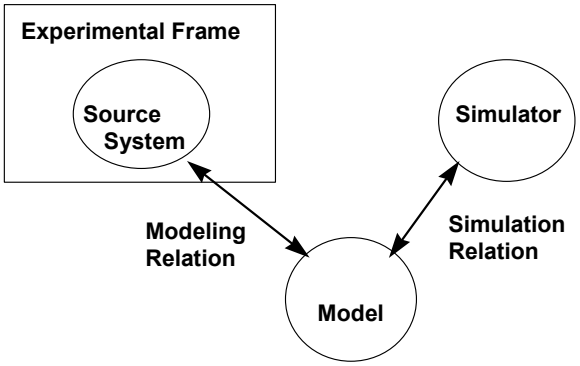

Figure 2. Framework Entities and Relationships

Source System

The source system is the real or virtual environment that we are interested in modeling. It is viewed as a source of observable data in the form of time-indexed trajectories of variables. The data that has been gathered from observing or otherwise experimenting with a system is called the system behavior database. This data is viewed or acquired through experimental frames of interest to the modeler.

Experimental Frame

An experimental frame is a specification of the conditions within which the system is observed or experimented; it is also the operational formulation of the objectives that motivate an M&S project. A frame is realized as a system that interacts with the source system, or System Under Test (SUT), to obtain the data of interest under specified conditions. An experimental frame specification consists of four major subsections:

- *input stimuli*: specification of the class of admissible input time-dependent stimuli. This is the class from which individual samples will be drawn and injected into the model or system under test for particular experiments.
- *control*: specification of the conditions under which the model or system will be initialized, continued under examination, and terminated.
- *metrics*: specification of the data summarization functions and the measures to be employed to provide quantitative or qualitative measures of the input/output behavior of the model. Examples of such metrics are performance indices, goodness-of-fit criteria, and error accuracy bound.
- *analysis*: specification of means by which the results of data collection in the frame will be analyzed to arrive at final conclusions. The data collected in a frame consists of pairs of input/output time functions.

When an experimental frame is realized as a system to interact with the SUT (or its model), the four specifications become components of the driving system. For example, a generator of output time functions implements the class of input stimuli.

Model

A model is a system specification, such as a set of instructions, rules, equations, or constraints for generating input/output behavior. Models may be expressed in a variety of formalisms that may be understood as a means for specifying subclasses of dynamic systems. The DEVS formalism delineates the subclass of discrete event systems and it can also represent the systems specified within traditional formalisms such as differential (continuous) and difference (discrete time) equations.

Simulator

A simulator is any computation system (such as a single processor, or a processor network, or, more abstractly, an algorithm), which is capable of executing a model to generate its behavior. The more general purpose a simulator is, the greater the extent to which it can be configured to execute a variety of model types. In order of increasing capability, simulators can be:

- Dedicated to a particular model or small class of similar models
- Capable of accepting all (practical) models from a wide class, such as an application domain (e.g., communication systems)
- Restricted to models expressed in a particular modeling formalism, such as continuous differential equation models
- Capable of accepting multi-formalism models (having components from several formalism classes, such as continuous and discrete event).

The DEVS formalism implements a well-defined concept of a simulation engine to execute models and generate their behavior. A *coupled model* in DEVS consists of component models and a coupling specification that tells how outputs of components are routed as inputs to other components. The simulator for a coupled model is illustrated in figure 3. It consists of a coordinator that has access to the coupled model specification as well as simulators for each of the model components. The coordinator performs the time management and controls the message exchange among simulators in accordance with the coupled model specification. The simulators respond to commands and queries from the coordinator by referencing the specifications of their assigned models. The simulation protocol works for any model expressed in the DEVS formalism. It is an algorithm that has different realizations that allow models to be executed on a single host and on networked computers where the coordinator and component simulators are distributed among hosts.

Validation and Verification Relationships

The entities *system, experimental frame, model,* and *simulator* take on real importance only when properly related to each other. For example, we build a model of a particular system for some objective; only some models, and not others, are suitable. Thus, it is critical to the success of a simulation modeling effort that certain relationships hold. Two of the most important are *validity* and *simulator correctness*.

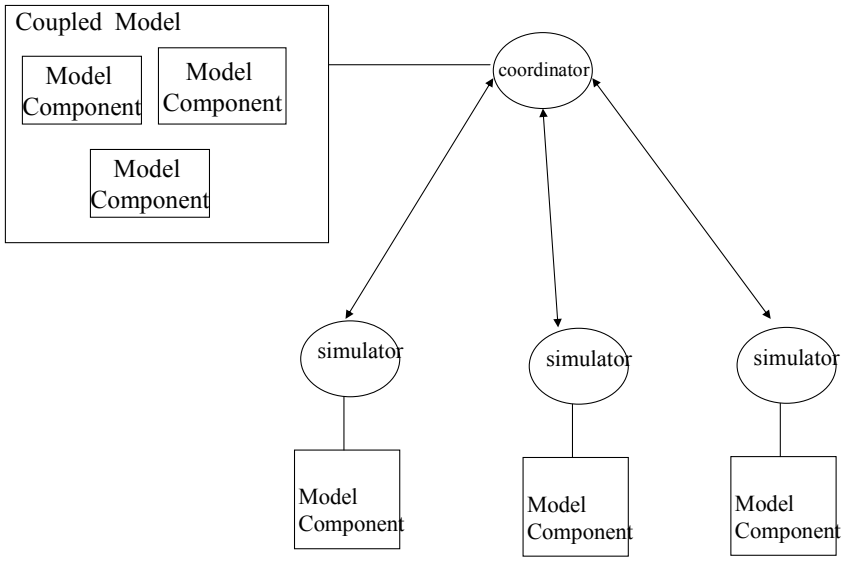

Figure 3. The DEVS Simulation Protocol

The basic modeling relation, *validity,* refers to the relation between a model, a source system, and an *experimental frame*. The most basic concept, *replicative validity*, is affirmed if, for all the experiments possible within the experimental frame, the behavior of the model and system agree within acceptable tolerance. The term *accuracy* is often used in place of validity. Another term, *fidelity*, is often used for a combination of both validity and detail. Thus, a high-fidelity model may refer to a model that is both highly detailed and valid (in some understood experimental frame). However when used this way, the assumption seems to be that high detail alone is needed for high fidelity, as if validity is a necessary consequence of high detail. In fact, it is possible to have a very detailed model that is nevertheless very much in error, simply because some of the highly resolved components function in a different manner than their real system counterparts.

The basic *simulation* relation, simulator correctness, is a relation between a *simulator* and a *model.* A *simulator correctly simulates a model* if it is guaranteed to faithfully generate the model's output trajectory, given its initial state and its input trajectory. In practice, as suggested above, simulators are constructed to execute not just one model but also a family of possible models. In such cases, we must establish that a simulator will correctly execute a particular class of models. Since the structures of both the simulator and the model are at hand, it may be possible to prove correctness by showing that a *homomorphism* relation holds.

Implementation of the above concepts and the M&S framework are key to the correct integration of a systems approach and formalism for DOD testing. By clearly defining and discriminating at fundamental levels the models, simulators, and experimental frames, we can better organize and apply M&S resources to the testing process. The subsequent description illustrates the use of DEVS formalism applied to actual test requirements.

4. An Example of DEVS Formalism Application

To help clarify the concept of experimental frame, we offer an example drawn from a current project to demonstrate the proposed methodology. In contrast to the monolithic federates of figure 1 a), part b) suggests an approach that allows more expeditious plug-and-play. Here, both models and frames are expressed as dynamic elements within a single formalism. The DEVS formalism supports composition of such elements into model-frame pairs with a standard simulation protocol, resulting in more efficient execution over existing middleware. Common use of the formalism in the simulator therefore obviates simulation incompatibility as an issue. Further, modularity provides ease of disassembly of model-frame pairs, so incompatibilities at the frame and model levels can be reduced to the smaller, more manageable problem sets that facilitate a "separation of concerns" strategy.

Separation of frames from models and simulators allows development of experimental frame *commodities* in both component and composite units to enable more automated and reusable test generators and the application of more powerful analytic, summarization, and visualization capabilities.

Example: *Single Integrated Air Picture (SIAP) Pilot Event Data Registration Frame.* Figure 4 illustrates the basic experimental frame for a project to develop a virtual environment for conducting SIAP Pilot Events. The frame specifies the components needed for data registration experiments and consists of sections for input stimuli, control, metrics, and analysis as illustrated in figure 4 a). The implementation of this specification in terms of dynamic elements in formalism is suggested by the block diagram in figure 4 b).

Development in this manner clearly separates the experimental frame as a module that can be coupled with models or system to be tested. Moreover, this frame becomes a reusable artifact in that it can be coupled with many different models or systems of a nature similar to the ones for which it was developed. Indeed, the frame, when implemented as a test platform, becomes a reusable commodity of economic value to the organization with known attributes and known application domains.

This example can be distributed over a wide area, with functions at different locations, especially in the DOD test environment where resources are rarely collocated at any given time and place. A distributed environment consists of networked sites, each with their military service-specific M&S assets connected through a common communications network. As depicted in figure 5, this connects multiple, geographically distributed sites to provide system-of-systems battlefield representative environments in support of developers, testers, and warfighter requirements using DOD and contractor test resources.

As shown in the map of distributed testing, the architecture based on the formalism provides a robust method of synchronization, the required real-time stimuli (using discrete event specifications), and real-time test monitoring and control. There are many advantages and high value to the discipline of a formal systems methodology over traditional *ad hoc* real-time system design.

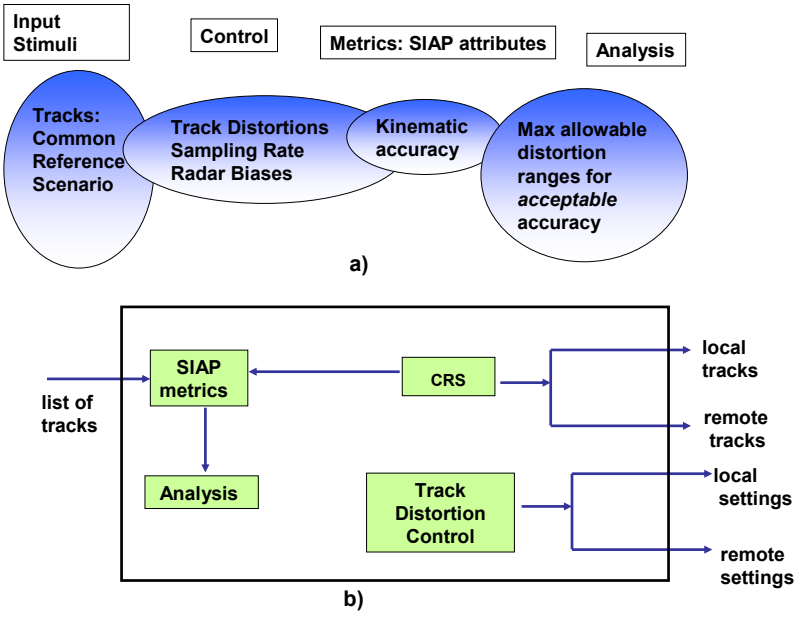

Figure 4. SIAP Pilot Event Data Registration Frame

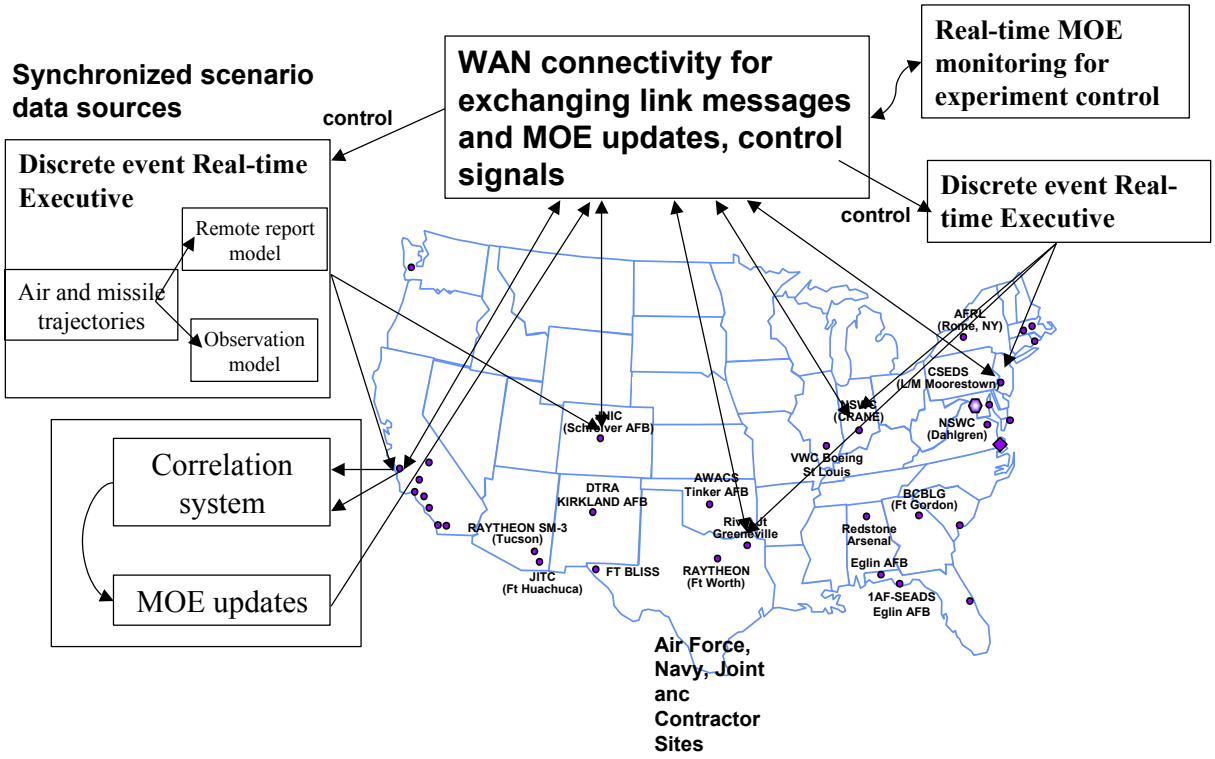

Figure 5. Distributed Testing Formalism Application

5. Applying the Formalism: Experimental Frames for DOD Testing

A military system may operate in multiple environments under a wide variety of conditions; therefore, a methodology is needed to guide the development of families of experimental frames that can cover the vast space of potential test cases. Such a methodology might employ well-known and commonly employed categorizations as a top-level starting point for frame development. For example, the Command, Control, Communications, Computers, Intelligence, Surveillance, and Reconnaissance (C4ISR) Architecture Framework provides broad levels of systems related to operational, system, and technical views. These may be crossed with the categories of context and perspective to generate matrix of initial categories as in table 1.

Note that while context may be most strongly associated with the operational view, it is relevant to the system and technical views as well. The overall context of eventual use will help to focus testing at the constituent levels to situations that are likely to be encountered. Within the context category, enumerating the environments in which the system is designed to function will generate associated frames.

Further well-known categories that cross with these are *Operational Testing (OT)* and *Developmental Testing* (DT), according to the phase of the lifecycle. Combat effectiveness testing is most closely associated with OT and the C4ISR operational view. However, within the Simulation-Based Acquisition evolution, the pressure to support earlier and continued testing that combines aspects of both OT and DT has implications for formalized, reusable development of experimental frames.

The analysis specification of a development frame might specify certain *measures of performance* (MOP) that typically judge how well parts of a system are operating. In contrast, the analysis portion of an operational frame might roll up MOPs into outcome measures, called *measures of effectiveness* (MOE). MOEs measure how well the overall system goals, for example, combat effectiveness, are being achieved. Other objectives of testing such as *behavior validation*, *diagnostic,* and *conformance* to standards will also generate particular experimental frames. *Interoperability* testing and testing for *conformance* to standards, goals that are central to the mission of JITC, likewise generate frames of importance to this organization. Basic frames for testing in the joint context must accommodate variants that relate to specific *military service-related variants* of the same (or similar) underlying functionality.

However objectives giving rise to experimental frames are characterized, the methodology must support their appropriate translation into operational frame specifications. For example, figure 6 depicts the process of transforming outcome measures (such as MOEs or MOPs) into analysis parts of experimental frames. In order to compute such measures, a model must include *output variables*, whose values are computed during execution runs of the model. The mapping of the output variables into outcome measures is performed by the analysis component of the experimental frame. Help with choosing the boundary between output variables computed in a model and outcome measures computed by a frame should be provided by the methodology, in that the more computation is transferred to the frame, the greater the separation of concerns, and the greater the reuse potential of both frame and model.

Table 1. Taxonomy of Objectives That Motivate Experimental Frame Development

C4ISR Architecture Framework	**Context—**Understanding how the system will actually function in its environment. If this is a system of systems, this understanding consists of how the intended joint configuration will function.	**Perspective—**The clear expression of the tester hypothesis, taking into account the policy, standards, and testing practices required by the testing organization
Operational View–Identifies Warfighter Relationships and Needs	Environments OT, MOE	Interoperability Combat effectiveness
System View—Relates Capabilities and Characteristics to Operational Requirements	DT, MOP	Diagnostic Upgrades Interoperability
Technical View–Prescribes Standards and Conventions	Interoperability	Conformance MIL-STDS, Protocols, Performance

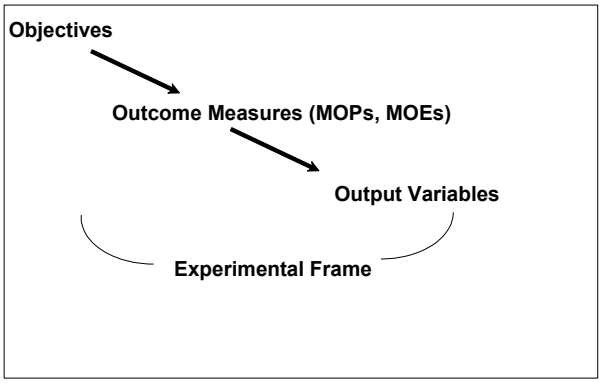

Figure 6. Transforming Objectives to Experimental Frames

6. The Formalism Approach to SLIRS (MIL-STD-6016) Conformance Testing

Vexing challenges confront the development of an M&S-based approach to SLIRS conformance testing. The specification document states requirements in natural language and at the behavioral level, employing directives such as the "system shall…," which often tend to be given ambiguous interpretations. The document is voluminous, with many hyperlinked chapters and appendixes, thus rendering its interpretation labor intensive and prone to error. Because of its size and complexity, the specification as a whole is potentially incomplete and self-contradictory (inconsistent). As a consequence, it is a major challenge to ensure that a certification test procedure developed from the specification document completely covers the requirements and can be consistently replicated across the numerous military service, national, and manufacturer contexts in which SLIRS standard certification testing will be executed.

A solution begins with the recognition that the SLIRS specification is, in fact, a description of a system that tells how it should respond to stimuli in various situations. In contrast to the interpretation in the context of system design, the objective in the context of conformance testing is to transform this description into a large family of test procedures and to specify the required outcomes of tests and sequence of tests. Further, sequences of tests can be regarded as the injection of stimuli by experimental frames that will realize the test procedures. To execute a test plan requires developing experimental frames that do the generation of input to the SUT and observe the SUT's output. For conformance at the technical level, a set of positive and negative test cases must be selected to provide the desired coverage. From an operational view, a set of tests can be designed to cover the normally expected interactions between the SUT and the world, as well as interactions that would not normally occur but cannot be excluded from consideration. The inputs come from either an external source, such as human operator or sensors, or as a result of an internal event in the system, such as a timeout or anomaly detection, which results in an issuance of an alarm. *A system that has discrete event input and output characteristics, such as this, can be characterized by a DEVS model.* This suggests the approach illustrated in figure 7, in which the SLIRS specification is formalized as a DEVS model. The latter represents, in operational form, the abstraction common to all systems that will be tested for conformance to the standard.

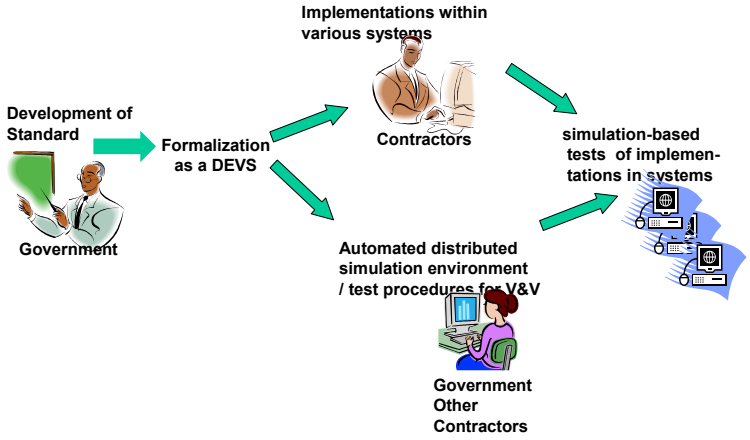

Figure 7. DEVS-based Independent Verification and Validation

On one hand, the DEVS model can then facilitate the development of test procedures; on the other hand, it could be employed by systems developers to account for the SLIRS tactical communication interoperability requirements in their more encompassing system designs. In contrast to current practice, such formalization, lays the basis for *independent verification and validation* (IV&V) in which the government determines its testing protocol early in the development of a new standard, thereby reducing its dependence on downstream system development that lies in the hands of contractors with their own proprietary interests. Moreover, this formalization supports the development of more automated generation of test procedures in the form of experimental frames synthesized from components resident in a repository.

A fundamental systems theory concept is that of a minimal realization of a given input/output behavior. In this case, the SLIRS document supplies the input/output behavior and the minimal realization will take the form of a DEVS model. A core element in the construction of a minimal realization is the discovery of an appropriate state space. Here, since the behavior is not presented in a rigorous manner, the state vector needs to be inferred by an analysis of the SLIRS document.

A common practice is to examine such requirements documents from the point of view of threads of linked paragraphs corresponding to a sequence of test actions. For example, a thread may start with the attempt of an operator to drop a radar track; if the track has a moderate level of importance, the SUT is required to generate a high-level alert requiring the operator to reassert the intention to drop the track. This thread may continue to consider issuance of a drop-track confirmation, a test for its subsequent nonexistence. In the systems-based approach, examining such a thread serves to uncover relevant elements of the state vector. In this case, such elements include status and importance of tracks, whether a drop-track request has been received, etc. Indeed, the state vector concept is a digital state representation in which every requirement thread can be described by changes in the state vector and outputs that result from such transition. The state vector summarizes the effect of testing at any time and we need to capture relevant parts of the state vector to determine the required outcome of testing. Although the complete state vector is not known at the start of analysis, we can build up the state vector as each thread is analyzed by determining the information needed to establish the required outcome of a test case and the items that will be changed as a result of the test. Indeed, the SLIRS document can be decomposed in a manner that relates to the state vector whose elements emerge as more and more threads are analyzed. This relation can be used to identify the set of requirements that apply to a given vector element and therefore that need to be tested to completely cover the requirements. By constructing a repository of experimental frames, we can provide a basis for computer assistance in synthesizing test procedures from repository-extractable components as illustrated in figure 8. A methodology for developing the repository will be discussed below, but first, we continue with an architectural framework for model development in this context.

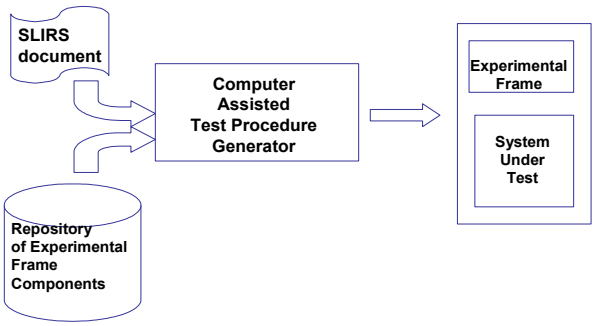

Figure 8. Computer Assisted Test Procedure Generation

Architecture for SLIRS Reference Implementation

Thus far, we have approached characterization of the SLIRS standard from a black box point of view. However, the documentation also suggests a structural decomposition into models. Figure 9 sketches a hierarchical architecture for development of SLIRS reference implementation. At the highest level are the major components, including communications, sensors, Identification Friend or Foe (IFF), and so on. The core component, Track Behavior, is shown as further decomposed into such functional modules as Data Registration, Track Number Management, and Surveillance Track Reporting. The decomposition continues with modules such as Data Registration further divided into finer level components.

Having the model architecture in hand allows us to consider how experimental frames are employed to test models in this architecture. Figure 10 illustrates three kinds of frame-model pairs in which a SLIRS reference implementation might participate. Figure 10 a) concerns the development of the SLIRS reference implementation in standalone fashion. Figure 10 b) illustrates how the SLIRS reference implementation, once verified and validated, could be used as a component in an experimental frame to assess the conformance of another SLIRS implementation.

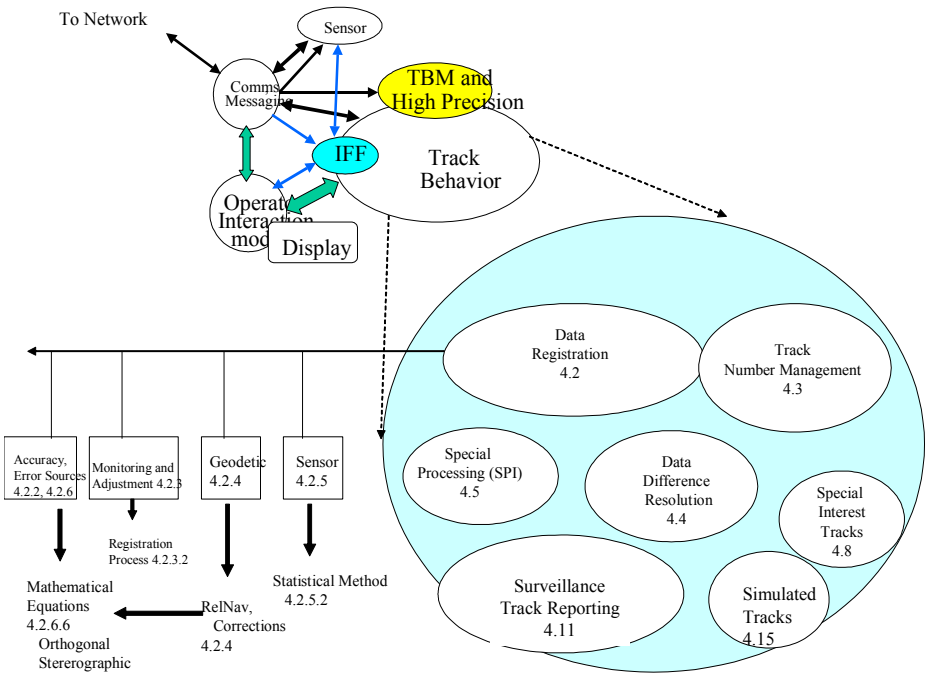

Figure 9. Model Architecture for SLIRS Reference Implementation

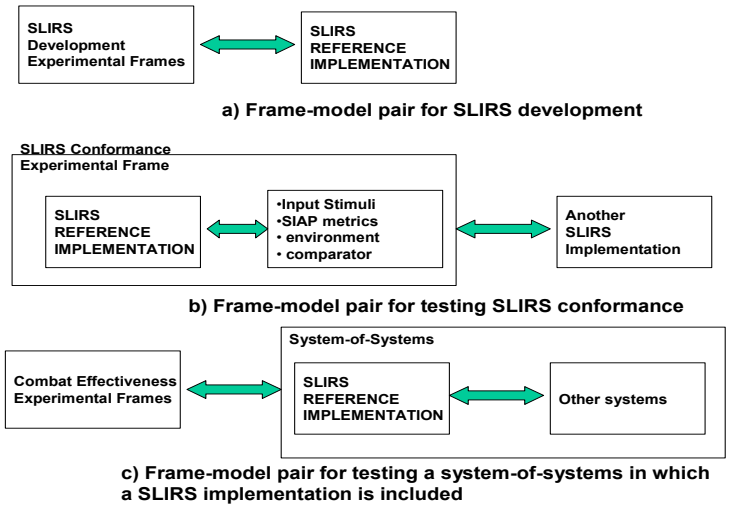

Figure 10. Model-Frame Pairs for SLIRS Reference Implementation

Finally, figure 10 c) illustrates the use of the SLIRS reference implementation as a component within a larger system of systems in which its ability to work together with other systems to create an effective combat system would be tested. In this case, one might be focusing on the SLIRS concept itself, e.g., testing the value of a single air picture within a littoral engagement or assault from the sea by US Marines in conjunction with the US Navy. Alternatively, and later in time, one might be employing an accepted SLIRS concept to test the overall projected combat effectiveness of new radar system in the same littoral scenario.

In the case of standalone development, the component models of the SLIRS architecture (figure 10 a), individually and in combination, would be tested for validation and verification. This would be followed by validation and verification of the overall implementation. Experimental frames applicable to the model components are illustrated in table 2, as well as the relative importance

of testing model components with such frames. We assume that frames are associated with broad categories of operational, system, and technical views. Typically, a model component has a natural association to a C4ISR view and the associated frame set would have high priority for use in testing that component. However, frames associated with other views may also bear some relevance to a component.

For example, Track Behavior is most naturally associated with the system view and it is most important to test it with frames derived from this view. However, the centrality of Track Behavior to the creation of a SIAP and the importance of the latter in combat effectiveness would also suggest the relatively high importance of testing it with operational view collaboration frames as suggested in table 2. On the other hand, Track Behavior is encapsulated deeply within the SLIRS implementation so that it does not directly participate in system interfaces. Thus, technical testing experimental frames of Track Behavior models would be less important than those associated with other views, although not entirely ruled out because the effects of technical factors such as geo-referencing and timing must be accounted for.

Experimental Frames for SLIRS

In this section, we suggest how the formalism and, in particular, the concept of experimental frames can be employed to develop correct and reusable test platforms for standards conformance within the C4ISR domain.

We develop the SLIRS (MIL-STD-6016) reference implementation as an example. The first step is the development of appropriate frames as described in the previous section. Some of the frames that might be considered for SLIRS testing are suggested in the following discussion.

Table 2. Experimental Frames for SLIRS Reference Implementation

C4ISR Architecture Views	Experimental Frames
Operational View	*Basic Collaborative Behavior:* This family of frames tests, and helps develop, the ability of several SLIRS-based mission computers to cooperate to produce, and share, a common air picture. Logically, such testing takes place after the Basic Individual Behavior is tested (below) but in concurrent engineering development, collaborative testing might take place once some essential precursors have been implemented and verified. *Human Operator:* The purpose of testing the interactive display and control inputs from the operator. The experimental frame would include sensors, systems, and networks in a likely engagement configuration to assess and modify the operator view and interface. Because the idea is for operational context, the platform, sensor, and networks would be military service-specific, with each instance therefore taking on a service-lead "flavor." A service-led Joint Task Force would therefore generate four different variants for these cases.
System View	*Basic Individual Behavior:* This family of frames tests, and helps develop, the SLIRS correlation algorithms and other functionality for correctness and performance in a standalone setting. An example for testing the correlation algorithm in isolation, following the methodology described here, is being developed for actual deployment [6]. *Sensors Effects:* The question addressed by the frame is how radar sensors, or their representations, having their own specific behaviors, will affect the performance of the SLIRS reference implementation. It is critical that the specification of the sensor behavior be flexible enough to respond to modifications of the environment (electromagnetic, weather, platform motion, etc.) in which sensors operate (at least to the extent that the SLIRS reference implementation behavior will be affected).
Technical View	*Network and Communication Interoperability:* This frame will allow for basic interoperability testing of the SLIRS reference implementation in large systems. It includes messaging and protocol behavior, which is done in conjunction with federates that provide network simulations, or with actual networks (e.g., Link-16), as well as timing, synchronization, and other technical factors.

7. Spiral Methodology

A methodology for developing experimental frames, models, and simulators is based on the formalism arising from dynamic systems theory. This methodology needs to be extended and refined for application to distributed systems testing so that it guides the concurrent engineering development of the required artifacts with integrated support for reuse and continuous accumulation of capability.

A notional methodology based on the spiral process [9] for software development is suggested in figure 11. At every spiral, it includes definition of objectives, appropriate frames, models and simulators, conducting experiments, and performing data analysis. Moreover, it suggests that the process can be greatly accelerated by integrating into it explicit interaction with appropriate database repositories for artifacts including frames, models, and simulators. Turns of the spiral can include further refinement of existing artifacts or addition of new ones occasioned by expansion of the development objectives. For example, at the next spiral, the SIAP Pilot Event Data Registration Frame (figure 4) may be refined to include more sources of track distortion or expanded to include tests of track number management.

It might be thought that something less than a full-scale M&S methodology might be sufficient for the testing of distributed systems. However, models and frames are dynamic elements whose use may depend on the particular objectives of the moment. Specifically,

- *A model may become a component in an experimental frame,* e.g., a radar model may become an element in an extension of the Data Registration frame in figure 4. In such an extension, the radar model might take in raw electromagnetic data and output associated track data so that distortions that originate earlier in the processing sequence can be included in the output of the frame.

- *An implementation of an experimental frame might be studied via M&S*, e.g., Reference [7] describes the use of an M&S package, Network Warfare Simulation (NETWARS)/OPNET, to assess the bandwidth requirements of the test network infrastructures. Such networks support interoperability test events involving live and simulated systems geographically distributed over hundreds of miles. The model of the test event identifies bottlenecks and over-purchase of bandwidth in order to resolve network bandwidth issues before the start of testing. Once validated, it can also be employed as the basis to discover, and mitigate, potential errors in configuration and setup. In this case, the implementation of the experimental frame is being examined through a NETWARS simulation model for its ability to support the conduct of test events.

Another example of the interchangeable roles of models and frames was given in the SLIRS discussion above. Because of this interchangeability, the methodology required for generic M&S becomes a part of that required for developing test environments for combat effectiveness of distributed military systems. Of course, the latter requires further extension in order to meet its particular requirements.

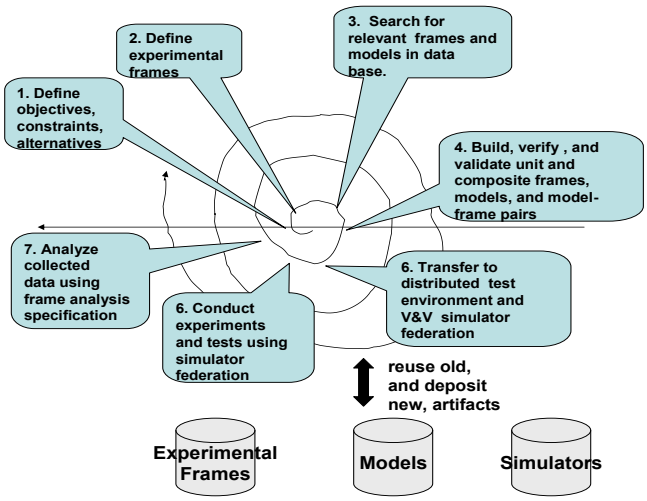

Figure 11. Spiral M&S Methodology for Distributed System Testing

Computerized Support for the Spiral Methodology

Computerized support for the development of models and simulators in the DEVS formalism is quite well established. However, such support for development of experimental frames is not as far advanced. We mention the following areas in which such support needs to be put in place to enable robust and widespread use of the spiral methodology for test infrastructure development:

- The development of experimental frame capture software to enable more automated and reusable test generators and the application of more powerful analytic/summarization/visualization capabilities.
- Support for construction of test platforms appropriate to stated test objectives and implementing associated experimental frames, where such construction eases/hides the complexities of building simulation engines by employing methods developed in the theory of modeling and simulation.
- Support for evolving test platform structures, (e.g., from constructive, to distributed, to real-time hardware-in-the-loop), in concert with system development, operational and (virtual) field test requirements, keeping complexity controlled with incremental additions.
- Support for consistency checking and error propagation analysis in frame, model, and simulator sets to increase trust and reliance on test results.

8. Development of Human Resources

A critical impediment to beneficial use of M&S in testing of distributed systems is the lack of trained personnel who are proficient in the knowledge and skills needed for efficient development and use of the infrastructure discussed above. The following are suggested as improvements and supplements to current testing. These will build the right infrastructure and provide the right education and training to overcome current barriers to progress.

- Increase the M&S competence of the existing workforce through various delivery mechanisms such as short courses and graduate degree programs. This increased competence will supplement, and complement, the multiple disciplines required in testing and engineering that are currently the strength of the test organization and its workforce.
- Build with new personnel who are educated in formal M&S programs at the university level.
- Innovate ways of injecting systems formalisms and support into DOD projects and systems lifecycles early on, so that their products are testable within M&S methodologies.

9. Conclusions and Recommendations

Military systems can be developed using a formal systems methodology that will account for functional requirements and extend to the lifecycle, including developmental and operational testing. Reliable and trusted system development requires integration of the system specification with contextual experimental frames that capture the objectives and experimentation constraints of the varied applications and uses expected of the system. Generic approaches that do not employ a formal methodology for distinguishing and evolving frames, models, and simulators cannot be trusted for robust, cost-effective system development.

The combination of traditional test methodology with rigorous M&S formal methodology provides a much richer and wider range of capabilities and options for the DOD test community. Test organizations may wish to consider a business case paradigm in which they invest in building the infrastructure to support the emerging trend toward M&S-based system development. In so doing, they will enhance their mission by introducing and integrating M&S-based testing at selected points throughout a system-of-systems lifecycle. In particular, the ultimate objective of the JITC is to improve and enrich combat effectiveness testing of operational systems-of-systems through a rigorous extension of interoperability testing that the JITC and others perform now. Instituting a formal methodology to achieve the versatility demanded by future DOD system testing requires a clear plan for evolving from the current mission and capabilities to one with increased scope. The plan must then be implemented incrementally to incorporate both process and infrastructure, allowing the test organization to increase the certainty that the interests of all stakeholders, (i.e., developers, testers, and warfighters) are consistently served. Some elements and examples of a long-range plan have been presented herein.

10. References

[1] 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.

[2] Modeling and Simulation in Manufacturing and Defense Acquisition: Pathways to Success (2002), National Academy Press.

[3] Theory of Modeling and Simulation, Academic Press, 2000.

[4] 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.

[5] MLST3 and Dual Link System (DLS) Comparison, Communication & Information Systems Division, Northrop Grumman.

[6] Virtual Environment for Patriot Pilot Event, Northrop Grumman Group.

[7] Interoperability Testing of Technology-Based Complex Systems, Phillip Hammonds, Ph.D., Northrop Grumman Information Technology, Bob Spiczka, ARINC, and Carl Agren, Northrop Grumman Information Technology.

[8] Single Link Interface Reference Specification for Link-16 (SLIRS-16), 20003.

[9] The Spiral Model as a Tool for Evolutionary Acquisition, Dr. Barry Boehm Wilfred J. Hansen, CrossTalk May 2001.