Original Article

JDMS

Journal of Defense Modeling and Simulation: Applications, Methodology, Technology
1–13

DOI: 10.1177/1548512913506620
dms.sagepub.com
SAGE

The art and science of live, virtual, and constructive simulation for test and analysis

Douglas D Hodson[1] and Raymond R Hill[2]

Abstract

Live, virtual, and constructive (LVC) simulation technologies are well-established in the areas of technology demonstration, mission rehearsal, and exercises. A promising new role for LVC simulation technology is to facilitate weapon systems testing by producing defendable results. Requirements to test new defense systems within a system-of-systems context and within joint force scenarios have placed demands on physical test ranges that are not likely to be met. The LVC testing option promises the breadth and depth of defense systems, either as real or simulated assets, to potentially meet the new test demands. However, leveraging this technology to support testing will require a shift in the approaches used by the LVC community. In this paper, we discuss the challenges facing the LVC testing initiative, both from an experimental and from an architectural and implementation standpoint.

Keywords

Distributed simulation, experimentation, test

1. Introduction

The US Department of Defense (DoD) uses modeling and simulation (M&S) for many purposes, including:

- Analysis – the field of operations research arose to address military applications, and there is still a major emphasis on using M&S to study the extremely complex problems of planning, organization, logistics, doctrine development, assessment of proposed new systems, etc.
- Research and development – at every stage in the acquisition process for a new vehicle, weapon system, command and control system, etc., M&S is used for trade-off studies, design decisions, performance prediction, logistic support requirements, among many other uses.
- Test and evaluation (T&E) – as new systems are developed, they are often tested first in M&S before they are taken onto physical test ranges, which are very expensive to operate. Interfaces with other systems are studied using M&S, and often other systems with which the new system must interact are represented in testing by simulations of those systems.
- Training – doctrines for how to use new systems are usually developed and tested via M&S. Training simulations are developed for those who need to learn how to operate, support, maintain, or otherwise interact with these systems.

Arguably, LVC has been primarily an exercise and demonstration technology to date; however, LVC events can support analytical decision making if properly used

[1]Department of Electrical and Computer Engineering, Air Force Institute of Technology, USA
[2]Department of Operational Sciences, Air Force Institute of Technology, USA

Corresponding author:
Douglas D Hodson, Department of Electrical and Computer Engineering, Air Force Institute of Technology, Wright-Patterson Air Force Base, OH 45433, USA.
Email: doug@openeaagles.org

within an analytical framework, as noted by Parker et al.[1] Among the valid reasons offered for using these environments in test for decision making are:

- facilitate testing in a joint environment;
- help realize system of systems testing;
- provide a mechanism to include a high density and diversity of defense assets unattainable in live test;
- reduce total cost and schedule for the test event; and
- determine requirements associated with conceptual systems.

To date, much of the LVC literature and effort has focused on architecture. While a correct focus to mature the technology of LVC simulation for eventual T&E use, the actual use of the architecture within an analytically defendable framework must also evolve and mature to ensure valid analytical results emanate from LVC-centric test events. We contend there are some real challenges in this maturation process, some of which we outline in this work.

This paper is organized as follows. In Section 2 we provide a more formal definition for LVC simulations than currently exists that captures both their unique properties while avoiding the problematic issues associated with categorizing degrees of system realism into discrete classes. In Section 3 the subject of experimentation and the experimental issues unique to LVC simulations is presented. In this section, we present issues from an experimenter's perspective, not necessarily the same issues that would concern a computer scientist or simulation engineer in implementing a solution. Section 4 provides the bridge from experiment requirements to implementation – we consider the role of conceptual models as being the primary communication mechanism. Section 5 provides a discussion of interoperability from the perspective of model integration to interfacing at the simulation or application level. Section 6 highlights unique LVC issues from the computer scientist's perspective – namely, technical implementation issues that must be considered when assembling a LVC architecture for analysis. In Section 7 we conclude by highlighting some of the challenges and promises unique to LVC simulations.

2. Defining LVC

DoD has defined three distinct classes of military simulation: live simulation, virtual simulation, and constructive simulation. In a live simulation, real people operate real systems. For example, a pilot launching weapons from a real aircraft at physical targets for the purpose of training, testing, or assessing operational capability is a live simulation. In a virtual simulation, real people operate simulated systems. For example, a pilot flying a simulated aircraft, launching simulated weapons at simulated targets would be considered a virtual simulation. In a constructive simulation, simulated people operate simulated systems.

		System	
		Real	*Simulated*
Human	*Real*	Live	Virtual
	Simulated	Virtual	Constructive

Figure 1. Simulation classification framework.[3]

Categorizing simulations into discrete classes such as live, virtual, or constructive is problematic since there is no clear division between these categories. For example, the degree of human participation in a simulation is infinitely variable, as is the degree of equipment realism. The categorization of simulations also lacks a category for simulated people interacting with real equipment.[2] Simulated people interacting with real equipment may play an important role in the use of LVC simulation for test.

LVC simulations differentiate themselves from the discrete classes by including various levels of all aspects of the defined classes. A LVC event will often include real people, real systems, and simulated systems interconnected to create a simulated world or 'environment' in which to interact. Owing to this, LVC simulations are actually hybrid simulations that include a mixture of entity types (e.g. hardware/software components, real people, real systems, etc.). This is particularly true for virtual simulations which routinely include both virtual and constructive entities.

Figure 1 provides a framework to classify live, virtual and constructive simulations based upon the types of entities they include.[3] This framework includes the aspect of real systems and assigns the classification based upon included entity types. For example, entities in a live simulation include real people and real systems. Entities in a virtual simulation include simulated systems operated by real people and real system operated by simulated people. All the entities in a constructive simulation are simulated by computer models; these have arguably been the mainstay of analytical M&S in defense.

Using this framework, an LVC architecture can be viewed as simulations that include a variety of entity types, thus broadening the scope of virtual simulations by including a wider range of live assets into the interactive environment and possibly augmented by a wider range of simulated systems.

While the classification framework is a good step towards clarifying the term ''LVC'' in terms of the entities

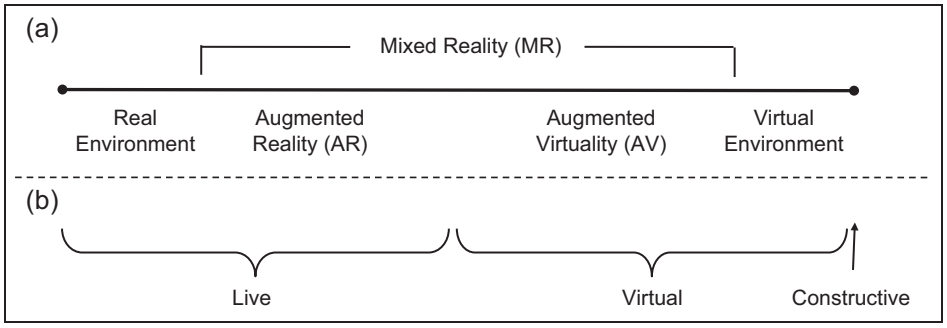

Figure 2. Reality–virtuality to DoD classifications.

they include, we believe it can be improved by leveraging Milgram and Kishino's reality-virtuality continuum defined for mixed reality displays.[4,5]

As shown in Figure 2(a), they defined a 'mixed reality' as anywhere between the extrema of the 'virtuality continuum', where the virtuality continuum extends from the completely real through to the completely virtual environment, with augmented reality (AR) and augmented virtuality (AV) ranging between. Augmented reality is a live, direct or indirect, view of a physical, real-world environment whose elements are augmented by computer-generated sensory input. Augmented virtuality refers to the merging of real world objects into virtual worlds.[4] Milgram's scale was developed considering display technology – we believe it is a useful conceptual framework to consider the different degrees of realism for the entities and the environment in which they interact within an LVC simulation.

Figure 2(b) presents a mapping from the mixed reality scale to the standard DoD classifications for live, virtual, and constructive. Using this scale, live entities span a range which includes real systems to real systems augmented with some aspect of a modeled world (i.e. augmented reality). Virtual entities (human or real systems) range from immersive environments that are completely computer generated to immersive environments that are augmented by real components (i.e. augmented virtuality).

From a military point of view, the endpoints define environments or conditions that are either completely real, meaning nothing is simulated which would include real-world operational conflicts, humanitarian efforts, peacekeeping, etc., to the other extreme, constructive, meaning nothing is real, everything is represented by a model (i.e. an abstraction of reality) whose system dynamics are simulated. It should be noted when mapping live simulations to the virtuality continuum, they *approach* the leftmost endpoint – they do not include it – if they did, a live simulation would not be a simulation at all. Combining frameworks, we offer the following definition for LVC:

> ''LVC simulations consist of a set of entities that interact with each other within a situated environment (i.e. world) each of which are represented by a mixture of computer-based models, real people, and real physical assets.''

It should be noted that some in the community describe or refer to LVC simulations based upon how they are implemented, with less emphasis on the interesting mixed reality aspects of the simulated environment itself. As an example, the interfacing of an existing constructive simulation with a virtual simulator is sometimes referred to as an LVC. We understand how this perspective has come about, as the desire and challenge in interfacing and reusing existing simulation assets for different purposes in itself appears to derive something uniquely new. We prefer to classify LVCs based on what is being simulated, and not solely based on how the simulation has been implemented or interfaced to each other; we believe this keeps the experimental aspects associated with mixed reality representations in more clear focus.

3. Experimental issues

There is no single 'best' way to undertake experimentation, rather the skill of the practitioner is to use a degree of artistic license in applying the science of experimental methods in order to maximize what can be achieved for a given problem under real-world constraints of resources, time, expectation and understanding [6].

In order to help practitioners, The Technical Cooperation Program (TTCP) developed a set of principles to consider when designing defense experiments. These principles are presented in the Guide for Understanding and Implementing Defense Experimentation (GUIDEx) and are categorized into three dominant themes,[6]'designing valid experiments', 'integrated analysis and experimentation campaigns', and 'considerations for successful experimentation'. Furthermore, the TTCP developed a

Figure 3. Experimental process.

general purpose planning flowchart or process centered around these principles and the application of the experimentation process – a simplified version is shown in Figure 3.

Because the major focus of GUIDEx is experiments based upon field events (live test, etc.) and human-in-the-loop (virtual) simulations, it provides a rich set of information relevant to these venues. Equally useful is the experimental planning strategy provided by Coleman and Montgomery.[7] While most experimental methods focus on industrial practice,[8] LVC simulation experimentation will require enhanced planning processes to:

- accommodate the range of stakeholders generally involved in an LVC event;
- manage and control the complexity creep prevalent in LVC architectural development preceding an event;
- define and accommodate the testing of multiple systems executing over common defense mission execution threads;[9] and
- unambiguously define and control the measurable objectives of the test along with the responses required to measure objective treatment.[10]

For the purposes of this article, we elaborate on issues particularly challenging in the domain of efficient experimental design methods (i.e. design of experiments) and architectural considerations and challenges associated with LVC simulation implementation.

3.1. Problem formulation

LVC simulations provide benefits across the system acquisition life cycle. During conceptual studies, system requirements can be refined. Basic system capabilities can be assessed during developmental testing. Feedback from live test can refine the system simulation models under development. Interactions with other systems and system effectiveness in a joint environment can be assessed during operational test. Such a range of capabilities means specific LVC events must be carefully defined and well-focused.

There are many papers on experimental design; see, for instance, those by Steinberg and Hunter,[11] Cortes et al.,[12] or Johnson et al.[13] Few focus specifically on the planning process for experimental design, exceptions being those by Coleman and Montgomery and Hahn.[7,14] None include the LVC simulation as a context for experimentation.

Defining the warfighter problem addressed by the LVC event helps to focus the discussion concerning the specific, measurable test objectives. Haase et al. indicated this definitional process can be quite difficult for LVC experimentation;[10] such definitional efforts are not often used in LVC practice. Properly defining test objectives helps focus a refinement of the factors examined in the test. The factor settings define proposed solution techniques considered in the test. These factor settings are developed along with the operational scenarios (in LVC terms, vignettes) relevant to the system operational deployment(s) considered.

Once the objectives, solutions and scenarios are defined, more traditional experimental design planning concerns arise. These traditional concerns relate to system response measurements and how these responses combine to create the metrics appropriate to the objectives of the test. Too often, test planners focus prematurely on metrics or key performance parameters. Collecting data out of routine is not as effective as the collection of data based on relevance to the study.

3.2. Experiment design

The full LVC simulation event may decompose into a series of sub events. For instance, early in the study, low fidelity studies may be used to inform subsequent higher fidelity studies. Sequencing test events yields a logical, efficient and effective test campaign.

Each test event should have some fully defined test hypotheses related to the specific event objectives. These hypotheses focus the LVC development and provide unambiguous test results. Too often, a LVC event is evaluated based on participant survey data; such data, while useful, often carries little in the way of statistically defendable results, particularly when the surveys are mis-constructed.

Experimental design considers two classes of factors pertinent to LVC simulation: those of interest that are controlled and varied, and those that are not of interest that are controlled but held constant. A third class, measurable but uncontrollable factors, for the most part do not apply to an event. The design may have held constant factors that are hard to control; the human operator involved in the event but not of interest to the study is an excellent example. The design planning should explicitly consider these factors and a method for varying or controlling the factor levels,

but for LVC simulation these planning considerations are more complex than found in the typical industrial or system design experiment.

LVC events require scenarios. These scenarios require varied levels of development. It is not unusual to develop specific operational concepts for each scenario. The design issue with scenarios becomes how to use them within the test execution; are the scenarios factors of interest in the design or are they part of the underlying environment over which the design is executed. This decision affects the statistical data collected from the test as it is not uncommon for these scenarios to vary multiple factors, some of which are not under particular control within the experimental design.

A final component of the experimental design is to collate all the work already accomplished to ascertain the technical impact to the LVC environment. An LVC event is still fundamentally about distributed software and its interactions. A technically infeasible LVC experimental design is unacceptable. Some of the technical issues that need to be considered with distributed simulation will be addressed later in this paper.

3.3. Experiment development

Once the technically feasible LVC experiment design is accepted, development of the experiment can commence; we realize that development and planning will always proceed concurrently.

Test hypotheses and test objectives are explicitly tied to LVC representation fidelity. A recent example clarifies. To answer questions of friendly aircraft penetration of protected airspace does not require the fidelity of air defense representation that a question of friendly aircraft survivability would require. Effective LVC testing will only need the fidelity of representation appropriate for the questions being asked.

It should be noted that for many years, as rapidly increasing computing power led to many new modeling possibilities, there was a generally held view that greater fidelity, or accuracy, was always better. Indeed, many took the term 'validity' to be almost synonymous with fidelity and detail. The modern view is that validity actually means 'fitness for purpose', with the purpose being to execute the desired experimental design to attain the data required to answer the analytical questions driving the LVC event. This means that we should consider the main measure of merit for M&S to be adequacy to support our experimentation, not the fidelity of representations or complexity of the implemented scenarios. The experimental design should effectively define what level of fidelity and complexity are adequate for the LVC event.[6]

Once fidelity of representation is defined, data and technical details supporting that representation are acquired. Data is arguably the most difficult aspect of a study. Decisions regarding data should consider representation fidelity as related to the testing hypotheses previously agreed upon.

As models (or representations) are developed, any verification and validation (V&V) of those models should be to the level necessary to define a 'fit for purpose' argument directed back to the test objectives. Model V&V is expensive and time consuming. Lack of model V&V can render study results useless. Experimental planning provides necessary insight to help define an appropriate level of model V&V.

Since a LVC event will generally involve a variety of systems as well as human operators, explicit consideration of pilot testing, subject training, and rehearsals are highly recommended (if not necessary). Pilot testing and rehearsal can generate useful statistical results and should provide a means to dry run the analyses. Humans can easily bias a test. Human operators learn, gain efficiency, even game the system, all of which bias the data collected to answer the agreed-to LVC event test objectives. The human-focused preparatory efforts help to reduce data bias and clarify test results.

3.4. Experiment execution

There is not a lot of published literature on best practices for test execution. Test ranges have best practices handed down over time. Industries codify their practices in internal manuals. LVC generally have exercise out-briefs or lessons learned. However, LVC tests and exercises are quite different in their planning, execution, and outputs.

An LVC test requires the production of defendable results. From a statistical perspective, the response data needed for defendable results should vary due to changes in the factors of interest, not due to other factors varied randomly during the test. The response data must be free of error due to unmeasured but influential effects.

An example will clarify unmeasured but influential effects. An LVC architecture is flexible, easily modified. An experiment can span a fairly long period of time. A concerned participant decides to 'try something new' and changes the processes used by the human operator. While quite appropriate during an exercise, this change means prior LVC event data no longer relates to the present (and possibly future) event data and in fact a new effect, operator process, has been introduced into the experiment and will quite likely be confounded with other agreed to factors of interest. A worst case, but likely case, is that the test will become incapable of explicitly answering the objectives driving the test in a statistically defendable fashion.

LVC simulation experiment execution requires an adherence to test protocol likely unfamiliar to those experienced in LVC simulation execution. The history of LVC

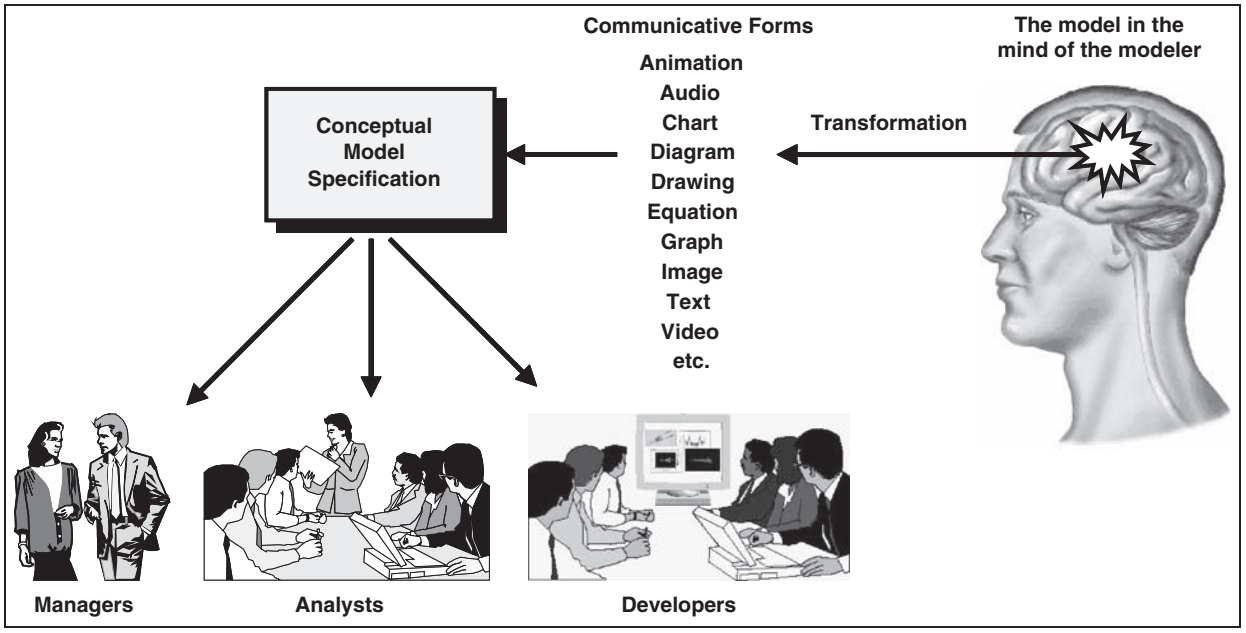

Figure 4. Conceptual model.[15]

has focused on exercise or demonstration events. Moving the future of LVC into test will require significant changes in event execution protocol.

3.5. *Analysis*

The environment brings about interesting challenges for analysis. Unlike live testing, the LVC environment can instrument just about everything. This means the test may collect response data of direct interest but also can collect response data indirectly related to the response. The challenge is to correlate the data to provide a complete answer to the objective driving the test. This is a new approach to data collection for the LVC practitioner.

Another challenge to analysis is capturing how well the system under study is received or employed. Past LVC events relied heavily on survey data based on participant experiences and perceptions. Statistically defendable test results applicable to system performance will need to support or refute such survey results to help the decision making process. For instance, if an event is examining chat capabilities within an operational aircraft, and pilot surveys positively favor this capability, indirect data such as chat use during periods of intense activity should support the positive use of the chat capabilities.

4. Conceptual modeling

After an experiment is designed, the requirements for what needs to be developed are communicated through a 'conceptual model'– in fact, a conceptual model serves as a bridge between the real world, requirements, and design – ideally developed as part of the experimental process. Balci and Ormsby define a simulation conceptual model as the model formulated in the mind of the simulation modeler and specified in a variety of communicative forms intended for different users such as managers, analysts, and simulation developers,[15] as shown in Figure 4. We consider the activity of developing a conceptual model more 'art' than exact 'science' and therefore it is difficult to define precise methods and procedures – it is inherently a creative endeavor guided by guidelines and best practice.

Conceptual modeling is the process of abstracting a model from a real or proposed system. It is almost certainly the most important aspect of a simulation project. The design of the model impacts all aspects of the study, in particular the data requirements, the speed with which the model can be developed, the validity of the model, the speed of experimentation and the confidence that is placed in the model results. A well-designed model significantly enhances the possibility that a simulation study will be a success.[16] We do contend that experimenter's/analyst in concert with subject matter experts (SMEs, i.e. people who possess special knowledge about the system or problem domain) perform this activity.

Robinson emphasizes the idea that a conceptual model is a non-software-specific description of the computer simulation model that will be,[16] is, or has been developed, describing the objectives, inputs, outputs, content, assumptions, and simplifications. The IEEE recommended

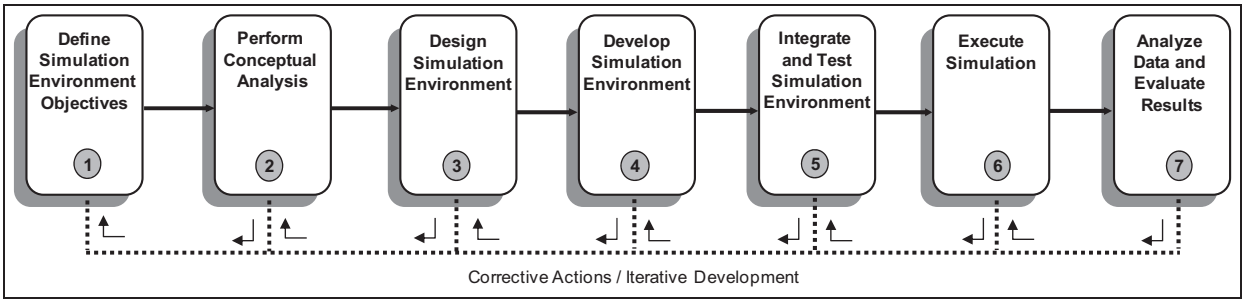

Figure 5. DSEEP Process.[17]

practice for the distributed simulation engineering and execution process (DSEEP) is a standard that describes a conceptual model in slightly more concrete terms,[17] namely, it is what the simulation environment will represent, the assumptions limiting those representations, and other capabilities needed to satisfy the user's requirements. As shown in Figure 5, step 2, a conceptual analysis is performed to define a conceptual model.

On the surface, the DSEEP process appears to be different than the TTCP experimental process shown in Figure 3– in fact, it is not. A mapping of the steps in the DSEEP process to TTCP process is provided below:

- Problem formulation
  - Step 1: Define simulation environment objectives. The user, the sponsor, and the development/integration team define and agree on a set of objectives and document what must be accomplished to achieve those objectives.
- Experiment design
  - Step 2: Perform conceptual analysis. The development/integration team performs scenario development and conceptual modeling, and develops the simulation environment requirements based upon the characteristics of the problem space.
- Experiment development
  - Step 3: Design simulation environment. Existing member applications that are suitable for reuse are identified, design activities for member application modifications and/or new member applications are performed, required functionalities are allocated to the member application representatives, and a plan is developed for the development and implementation of the simulation environment.
  - Step 4: Develop simulation environment. The simulation data exchange model (SDEM) is developed, simulation environment agreements are established, and new member applications and/or modifications to existing member applications are implemented.
  - Step 5: Integrate and test simulation environment. Integration activities are performed, and testing is conducted to verify that interoperability requirements are being met.
- Experiment execution
  - Step 6: Execute simulation. The simulation is executed and the output data from the execution is preprocessed.
- Analysis
  - Step 7: Analyze data and evaluate results. The output data from the execution is analyzed and evaluated, and results are reported back to the user/sponsor.

This relationship is highlighted to emphasize a few key points. As previously mentioned, to date much of the technical literature and effort has focused on architecture and implementation. This is clearly present in the DSEEP standard with three process steps addressing this aspect alone. In fact, a major source of variation in how the seven-step process is implemented relates to the degree of reuse of existing products.[17] This focus on reuse is important – LVC architectures are complicated software systems – in fact, they are some of the most complicated systems to build as will be discussed in Section 6. But even though LVC simulations are challenging to build, that should not distract the experimenter from focusing on the first two steps, namely, problem formulation and experiment design.

Another reason to emphasize this relationship is to recognize a natural reoccurring conflict – the conceptual model developed during the experimental design phase should *drive* the development phase, not the other way around. Experience has shown that a strong desire or demand to reuse existing simulations and software can result in a simulated system representation that no longer maps to the experimenter's conceptual model – this can introduce adverse effects into the system (such as fidelity

mismatches, confounding, and the changing of simulation objectives to match the conceptual model) and should be avoided. The desire to reuse existing simulations is often motivated by cost, risk, and some would argue the validity of specific models - although usage of the term *validity* without regard to purpose should throw a red flag, as validity is related to fitness for purpose – an aspect that is often overlooked.

The minimal information needed regarding a simulation at the modeling level can be summarized as the entity-event-states paradigm.[18]

- Entities are the simulated things that represent weapon systems or military units that cannot be further separated. They are characterized by attributes.
- Events connect entities, which can be individual or groups of entities. When events happen, entities are affected.
- States of entities are equivalent to specified attribute value constellations of these entities. For example, a system may be defined to be in a state of ''standing'' if the attribute velocity has the value zero.

By defining what entities exist, what events exist, what events and entities are related, and how the state of an entity changes when it is affected by an event, the conceptual model conveys to the simulation engineer an understanding of the causes and effects present in the simulated environment. With this understanding, the development of the LVC architecture proceeds by distributing the management of entities to a set of interconnected resources (i.e. simulations or real assets) that exchange entity state information and events to form a larger interactive synthetic environment.

5. Interoperablity

The IEEE glossary defines *interoperability* as ''the ability of two or more systems or components to exchange information and to use the information that has been exchanged''.[19] The term is closely related to the term *compatibility* which is defined as ''the ability of two or more systems or components to perform their required functions while sharing the same hardware or software environment''. In the context of a distributed simulation, interoperability concerns itself with the interfaces between different models, simulation applications and/or resources so that the simulated system (or system of systems) operates correctly and aligns with the experimenter's conceptual model.

Harkrider and Lunceford defined composability as the ability to create, configure, initialize, test, and validate an exercise by logically assembling a unique simulation execution from a pool of reusable system components in order to meet a specific set of objectives.[20] They introduced the aspect of logically assembling and, as such, emphasized the necessity for a common basis for the conceptual models that describe the underlying logic. The composition itself is driven by requirements defined by the experimenter's conceptual model.

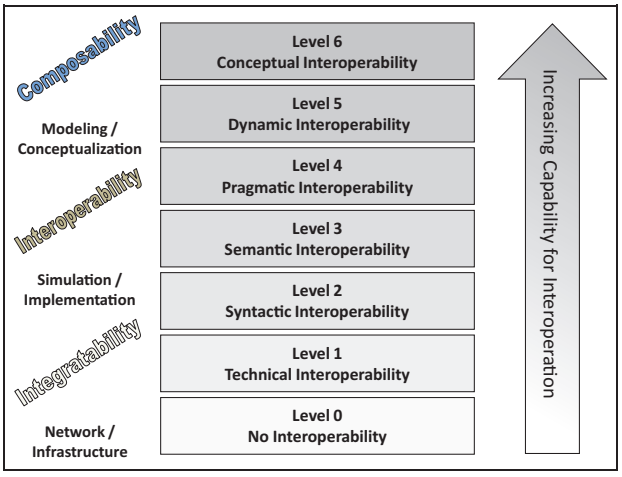

Figure 6. Levels of conceptual interoperablity model (LCIM).

Given these definitions, interoperability is seen as the broader, technical principle of interacting systems based on information exchange, while composability deals with the selection and composition of preexisting domain solutions to fulfill user requirements.[21]

Issues can arise as a result of potential misalignment between the models used in an implementation and the conceptual model as defined by the experimenter, and also possibly between applications. Tolk introduced a framework to consider the problem of interconnecting different models and/or simulation applications together.[21] His *levels of conceptual interoperablity model (LCIM)* framework is shown in Figure 6.

The framework spans the entire spectrum starting from no interoperability to a level that perfectly aligns with the experimenter's conceptual model. A brief description of each interoperability level is given as follows:

- Level 0 (No) – no connectivity, the systems or resources are stand-alone and do not interact.
- Level 1 (Technical) – a means exists to connect and exchange data between systems, for example, a network connection.
- Level 2 (Syntactic) – protocols to exchange the right forms of data in the right order have been established, but the meaning of data elements is not established.

- Level 3 (Semantic) – the meaning of the data elements being exchanged is shared.
- Level 4 (Pragmatic) – systems are aware of the methods and procedures that each system is employing for information exchange.
- Level 5 (Dynamic) – systems are able to reorient information production and consumption based on understood changes to meaning, due to changing context as time increases.
- Level 6 (Conceptual) – systems at this level are completely aware of each others information, processes, contexts, and modeling assumptions. These assumptions and constraints of the meaningful abstraction of reality are aligned.

In addition, as shown by Page et al.,[22] Figure 6 also distinguishes interoperation into three key concepts:

- Integratability contends with the physical/technical realms of connections between systems, which include hardware and firmware, protocols, networks, and so on.
- Interoperability contends with the software and implementation details of interoperations; this includes exchange of data elements via interfaces, the use of middleware, mapping to common information exchange models, and so on.
- Composability contends with the alignment of issues on the modeling level. The underlying models are purposeful abstractions of reality used for the conceptualization being implemented by the resulting systems.

We contend that most of the challenges associated with interoperability stem from the desire to reuse existing simulation solutions most likely designed for different purposes. This desire for reuse is understandable as the US Department of Defense (DoD) has a massive and expensive problem – the 'expensive' aspect of the problem arises due to the complexity and different types of simulations that are developed during the entire life cycle of a system. By and large, models and simulations of a new system are developed 'from scratch' by each agency and contractor at each stage of system development, implementation, and use. Historically, there has been little reuse of simulations across systems or phases of development, so taxpayers have often paid for the creation of multiple simulations during each major system development program.

Tolk contends that for many M&S developers,[21] questions regarding the future interoperability and composability of their solution are not the main concern during design and development. They design their M&S system or application to solve special problems and provide a solution. This paradigm needs to change to help mitigate the problems associated with usage in other simulation contexts, such as an LVC simulation.

6. Implementation issues

In an LVC event, the *system under study* is often a collection of systems or 'system of systems' which includes humans and/or real operational system hardware. Because these real-world elements are present, timing constraints are imposed on the simulation environment which, from a software and hardware engineering standpoint classifies the simulation as a real-time system. From an experimenter's perspective, LVC simulations are large-scale distributed simulations that include real people and operational assets. From a simulation engineer's perspective, LVC simulations are real-time, distributed, concurrent systems that interact with people and operational hardware or assets. The aspect of including people and/or real hardware *in-the-loop* and its inherent distributed implementation distinguishes these kinds of simulations from typical standalone constructive simulations - time cannot be stopped, paused, or reversed, it moves forward in sync with the wallclock. We first consider some differing approaches to LVC architecture design in relation the LCIM framework, then present some complicating issues.

6.1. Common framework

As the LCIM framework in Figure 6 indicates, the highest level of interoperability (i.e. conceptual interoperability) is achieved when the integrated models are in alignment with each other and the representation of the system as envisioned by the experimenter. From an implementation standpoint, Pratt et al. addressed two main issues associated with composability,[23] *functional* and *semantic* granularity. The semantic granularity of a system is where things like thoughts, concepts, and level of interactions begin to separate the system components. These divisions go a step further and start to define groups of modules that 'make sense' together. We contend that this aspect of interfacing or aligning models to work in meaningful ways is associated, if not the same, as the highest forms of interoperability. The LCIM clearly indicates this relationship, as shown in Figure 6; in fact, composability is considered a higher level or form of interoperability. This level of interoperability can be reached using a common framework with common model abstractions to build reusable LVC components. In this context, the LVC components consist mostly of both new simulations assembled ''from a pool of reusable system components in order to meet a specific set of objectives'' and/or existing simulation built from the same abstractions.

An example of this approach as been employed by the Air Force Simulation and Analysis Facility (SIMAF) at Wright-Patterson Air Force Base. As a common baseline or set of modeled abstraction, they leverage the Open Extensible Architecture for the Analysis and Generation of Linked Simulations (OpenEaagles) framework.[24,25] Given this framework, unique simulation solutions to meet a specific set of objectives are built. (In the interest of full disclosure, one of the authors of this paper is the project leader for the open-source OpenEaagles framework – the intent is not to advocate for a particular solution, but to illustrate that using this approach, improvements in interoperability with high degrees of reuse exist. Using them helps simultaneously resolve model alignment and composability issues and provides a simulation engineer the capability to design uniquely new solutions that better fits requirements.)

6.2. Middleware

As Section 5 indicated, many models and simulations of a new system are developed from scratch, which implies a large number of models and simulations have been, and continue to be, built with different modeled abstractions. For this typical situation, the highest levels of interoperability and composability are less likely to be realized. From a simulation engineering standpoint, the ability to craft a unique simulation solution is constrained by the lack of commonality. In this situation, the so-called 'middleware' products or standards are used to interconnect the differing resources (simulations, hardware, etc) into a unified system.

Middleware works at the application layer of the network stack and typically comes in the form of libraries that can be included into a coding project. The role of middleware includes:[26]

- Abstracting away operating system differences.
- Providing high-level services that manage connections between users.
- Providing functionality for the formatting and sending of messages for specific application domains, or for specific types of services.
- Some types of middleware, providing a more complete framework that includes all the necessary services to build a class of applications.

As shown in Figure 7, middleware can be viewed as layers of services, each abstracting away some technical details for how to connect and interact with other networked resources. For example, at the lowest level of abstraction is the concept of ''sockets'' for establishing network connections.

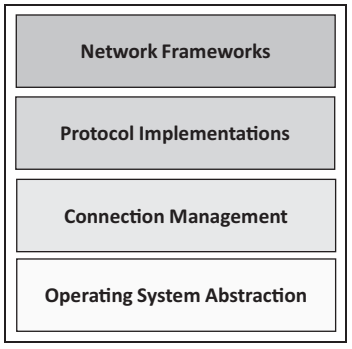

Figure 7. Layers in middleware software.[26]

In the domain of LVC simulations, three middleware solutions are the most common; the distributed interactive simulation (DIS) protocol,[27] the high-level architecture (HLA),[28] and the test and training enabling architecture (TENA).[29] Whereas DIS is a message-based oriented solution, HLA and TENA are object-sharing solutions. Each has their strengths and weaknesses – which is the best choice depends upon conceptual model requirements and fundamental architectural trade-offs that must be considered as a result of the real-time distributed nature of the simulated system itself.

Many, if not most, large LVC simulations consist of multiple architectures (so-called 'mixed architecture') in that they reuse component simulations developed to be compliant with one of the three major simulation architectures (DIS, HLA, and TENA). Bridges connect simulations developed for different versions of the same architecture (e.g. HLA 1.3 and HLA IEEE 1516) and gateways connect different architectures. Many implementations of these have proliferated over the last 15 years. Reducing this proliferation has been a major focus of LVCAR implementation efforts.[30]

6.3. State consistency issues

As a combination of software and hardware, LVC simulations are designed to *instantiate* the conceptual model envisioned for test or analysis purposes. In order to implement the conceptual model, simulated entities are often mapped to different resources. Figure 8(a) presents a notional conceptual model that includes two interacting entities that exchange events as discussed in Section 4. As shown in Figure 8(b), a possible architecture for the LVC simulation might be two different simulations or simulators interconnected by a network. Given this arrangement, entity data created and locally managed will need to be *shared* using a network so that interactive environments can be created for each of the human participants.

Generally speaking, LVC simulations execute by passing state data between distributed applications and other

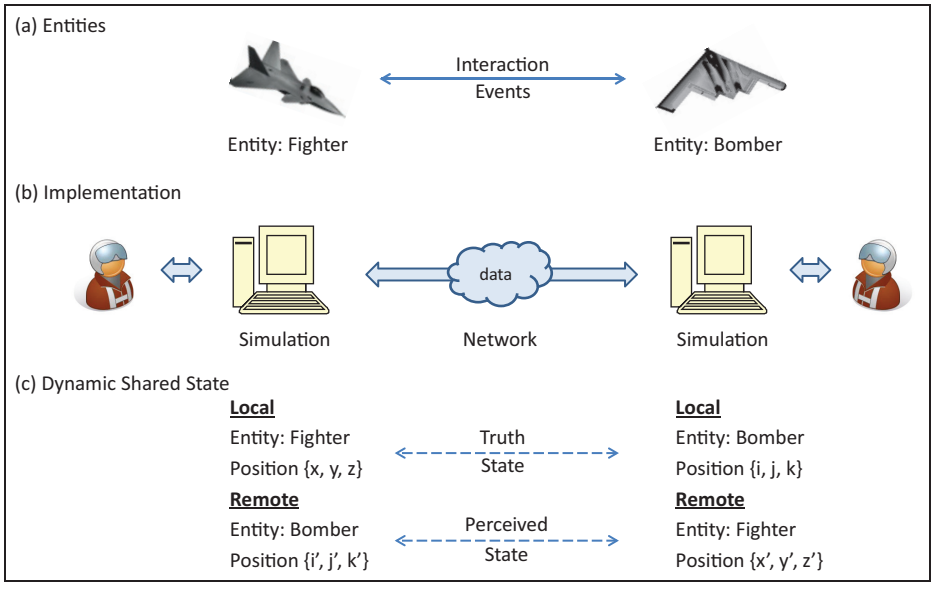

Figure 8. Entities, implementation, and dynamic shared state.

resources. As a result, a fundamental issue arises; applications require state data that is not locally managed to produce correct outputs, but they cannot wait until all data is received and still be highly responsive to new user inputs.

This sharing of data in combination with the requirement for each simulation to be responsive to real people or hardware creates a fundamental conflict. The conflict is a result of the application not being able to wait for the most current data value(s) and still meet real-time interactive response time constraints. Response time for interactive systems is the time it takes for inputs (e.g. stick and throttle) to affect outputs (e.g. computer display). If the distributed processes are connected via a network infrastructure with a relatively high latency, data being used by one simulation might be out of date or 'old' when compared to its current correct value as managed by the other. This inconsistency in state data is a distinguishing characteristic of both virtual and LVC simulations which must be recognized and managed. This problem is not new, it has existed since the advent of distributed interactive simulation and is sometimes referred to as the consistency-throughput tradeoff.[31] Today, the problem can often be found in the online gaming community.

Figure 8(c) shows the result of this conflict for our simple example, the state data associated with the location of *remote* entity(s) most likely will not match their true position as managed by their local hosting application. This results in a consistency issue associated with simulation state - meaning, that state of some of the shared data is in error.

The DIS standard addresses this issue by defining thresholds to determine when updates of specific data should be sent based on exceeding a pre-defined error values.[27,32] The standard defines default *threshold* values, but they should be determined based upon simulation requirements. Hodson investigated this aspect of LVC simulations by considering not only network latency issues,[3] but also additional delays associated with application structure and their meaning in terms of *validity intervals*– a term borrowed from literature in the domain of real-time software system development.

7. Conclusion

7.1. Challenges

The future use of LVC technology for T&E purposes, or for decision making purposes in general, are not without challenges. Below are some of these challenges.

- Keeping architectures scoped to answer specific test objectives. LVC still suffers from a 'cool factor'. More complexity can be added to increase 'realism' but this will often come at the expense of the analytical insight gained. LVC for test needs to avoid the complexity creep.
- Operators must think like analysts and analysts must think like operators. Testing must represent realistic scenarios (and operator view) while retaining just enough complexity to answer the test objectives (an analyst view). Cooperative negotiation must arrive at the right balance of realism and computational complexity.
- Statistical experimental design methods, well-established in industrial settings and gaining

prominence in DoD test and evaluation, require a tailoring for LVC simulation events; the tests will have lots of things occurring, will involve the human element, and are based on the use of scenarios or vignettes, all of which do not directly fall within the current statistical experimental design framework.

- The executing LVC simulation is a test event not a training or demonstration exercise. An LVC test will involve the human operator. If not the focus of test, human operators are controlled factors and thus their free-will may need to be constrained to obtain the objective data required to estimate system effectiveness and system factor contributions. This will require a change in current practice.
- Use or purpose for LVC simulation. An LVC environment can be used for training, exercises, demonstrations, or as per the focus in this paper, testing. The architecture, planning, representation, fidelity and eventual test execution will differ based on the desired use.
- Even though LVC simulations include aspects of reality, they are not completely real. Thus, an LVC simulation is an abstraction of reality which can be used to inform a test program and subsequent decision making, but cannot entirely replace live test events.
- Using LVC technology is not without cost. Since LVC events include many resources, can require longer run times, and involve humans, the cost of each run can be considerable. In additiion, there may be many factors of interest, resulting in a large number of trials.

7.2. The LVC promise for test

As Gray notes,[33] LVC can be used to generate test results. The LVC ''provides the test community with an infrastructure capability that supports testing across the acquisition process''.[9] The LVC environment can also support the density and diversity of assets found in the actual joint operational environment. Thus, the LVC technology does present a very real capability for the joint test world from conceptual through operational testing. Growth of statistical rigor applied to LVC test planning along with the technological advances to mature LVC architectures will help realize the promise for test.

Acknowledgment

Special thanks go to Dr. Catherine Warner and Mr. George Rumford for their continued support of the Science of Test Research Consortium.

Funding

This research was supported by the Office of the Secretary of Defense, Directorate of Operational Test and Evaluation, the Test Resource Management Center and the Simulation and Analysis Facility.

References

1. Parker EP, Miner NE, van Leeuwen BP, et al. Testing unmanned autonomous system communications in a live/virtual/constructive environment. *ITEA J* 2009; 30: 513–522.
2. US Department of Defense. *Department of Defense modeling and simulation (M&S) glossary*. DoD 5000.59-M, 1998.
3. Hodson DD and Baldwin RO. Characterizing, measuring, and validating the temporal consistency of live-virtual-constructive environments. *Simulation* 2009; 85: 671–682.
4. Milgram P and Kishino F. Taxonomy of mixed reality visual displays. *IEICE Tran Inf Syst* 1994; E77-D: 1321–1329.
5. Milgram P, Takemura H, Utsumi A, et al. Augmented reality: a class of displays on the reality–virtuality continuum. *Proc SPIE Int Soc Opt Eng* 1995; 2351: 282–292.
6. Bowley D, Comeau P, Edwards R, et al. *Guide for understanding and implementing defense experimentation (GUIDEx) – version 1.1*. The Technical Cooperation Program (TTCP), 2006.
7. Coleman RE and Montgomery DC. A systematic approach for planning a designed industrial experiment. *Technometrics* 1993; 35: 1–27.
8. Goh TN. A pragmatic approach to experimental design in industry. *J Appl Stat* 2001; 3: 391–398.
9. Foulkes JB. Realizing jointness through early testing of systems in a distributed live-virtual-constructive environment. *ITEA J* 2009; 30: 191–192.
10. Haase CL, Hill RR and Hodson D. Using statistical experimental design to realize LVC potential in T&E. *ITEA J* 2011; 32: 288–297.
11. Steinberg DM and Hunter WG. Experimental design: review and comment. *Technometrics* 1984; 26: 71–97.
12. Cortes LA, Duff E and Bergstrom D. A primer on design of experiments for US Navy T&E managers. *Technical report, Naval Sea Systems Command*, 2011.
13. Johnson RT, Hutto GT, Simpson JR, et al. Designed experiments for the defense community. *Qual Eng* 2012; 24: 60–79.
14. Hahn GJ. Some things engineers should know about experimental design. *J Qual Technol* 1977; 9: 13–20.
15. Balci O and Ormsby WF. Conceptual modelling for designing large-scale simulations. *J Simul* 2007 (1): 175–186.
16. Robinson S. Conceptual modelling for simulation. Part I: definition and requirements. *J Oper Res Soc* 2008; 59: 278–290.
17. IEEE 1730–2010. Recommended practice for distributed simulation engineering and execution process (DSEEP).
18. Tolk A. *Engineering principles of combat modeling and distributed simulation*. Addison Wesley, 2012.
19. Geraci A. *IEEE standard computer dictionary: compilation of IEEE standard computer glossaries*. Piscataway, NJ: IEEE Press, 1990.

20. Harkrider S and Lunceford WH. Modeling and simulation composability. In: *Interservice/industry training, simulation and education conference (I/ITSEC)*, 1999.
21. Tolk A. Interoperability and composability. In: Sokolowski JA and Banks CM (eds) *Modeling and simulation fundamentals: theoretical underpinnings and practical domains*. Wiley, 2010, pp.403–433.
22. Page EH, Briggs R and Tufarolo JA. Toward a family of maturity models for the simulation interconnection problem. In: *Proceedings of the simulation interoperability workshop*, Spring 2004.
23. Pratt DR, Ragusa C and von der Lippe S. Composability as an architecture driver. In: *Interservice/industry training, simulation and education conference (I/ITSEC)*, 1999.
24. Hodson DD, Gehl DP and Baldwin RO. Building distributed simulations utilizing the EAAGLES framework. In *Interservice/industry training, simulation and education conference (I/ITSEC)*, 2006.
25. Rao DM, Hodson DD, Stieger JM, et al. *Design & implementation of virtual and constructive simulations using OpenEaagles*. Linus Publications, 2009.
26. Steed A and Fradinho Oliveira M. *Networkd graphics – building networked games and virtual environments*. Elsevier, 2010.
27. IEEE 1278.1–2012. Draft standard for distributed interactive simulation (DIS) – application protocols.
28. IEEE 1516–2000. Standard for modeling and simulation high level architecture (HLA) – framework and rules.
29. DoD Foundation Initiative 2010. *The test and training enabling architecture – architecture reference document*. US Department of Defense, 2002.
30. Henninger AE, Cutts D, Loper M, et al. Live virtual constructive architecture roadmap (LVCAR) final report. *Technical report, Institute for Defense Analyses*, September 2008.
31. Singhal S and Zyda M. *Networked virtual environments: design and implementation*. Addison Wesley, 1999.
32. IEEE 1278.2–1995. Standard for distributed interactive simulation (DIS) – communication services and profiles.
33. Gray F. Experimental designs for development tests in distributed joint environments. In: *Proceedings of the 2007 Air Force T&E Days*, 2007.

Author biographies

Douglas D. Hodson is an Assistant Professor of Software Engineering with the Air Force Institute of Technology. He received a B.S. in Physics from Wright State University in 1985, and both an M.S. in Electro-Optics in 1987 and an M.B.A. in 1999 from the University of Dayton. He completed his Ph.D. at the Air Force Institute of Technology in 2009. He has over 25 years of experience in the domain of modeling and simulation and has a research interest in characterizing the consistency of shared simulation state data in terms of its temporal properties to estimate Live-Virtual-Constructive and Distributed Virtual Simulations performance, cloud computing and modeling quantum key distribution systems. He is the technical lead developer for the open-source OpenEaagles simulation framework that has been used to develop a wide variety of standalone and distributed simulation applications and solutions. He is also a DAGSI scholar and a member of Tau Beta Pi. His email address is doug@openeaagles.org

Raymond R. Hill is a Professor of Operations Research with the Air Force Institute of Technology. He has a Ph.D. in Industrial and Systems Engineering from The Ohio State University. His research interests include applications of simulation, applied statistical modeling, mathematical modeling, decision analytical methods, the design and analysis of heuristic optimization methods, and agent-based modeling. He is a member of the INFORMS, the Military Applications Society of INFORMS, the Military Operations Research Society (MORS), the International Test and Evaluation Association, the Institute of Industrial Engineering (IIE) and is a former Director of the Operations Research Division of the IIE. He is an Associate Editor for Naval Research Logistics, Military Operations Research, the Journal of Defense Modeling and Simulation, and the Journal of Simulation. He was a proceedings co-chair for both the 2008 and 2009 Winter Simulation Conferences and will serve as the General Chair for the 2013 conference. His email address is rayrhill@gmail.com.