ilmedia

TECHNISCHE UNIVERSITÄT ILMENAU

Harmon, Scott Y.; Hoffmann, Doug C.; Gonzalez, Avalino J.; Knauf, Rainer; Barr, Valerie B.:

Validation of human behavior representation

VALIDATION OF HUMAN BEHAVIOR REPRESENTATIONS

S.Y. Harmon
Zetetix, Oak Park, CA

C.W.D. Hoffman
U.S. Army White Sands Missile Range, Las Cruces, NM

A.J. Gonzalez
University of Central Florida, Orlando, FL

R. Knauf
Technische Universität Ilmenau, Ilmenau, Germany

V.B. Barr.
Hofstra University, Hempstead, NY

TABLE OF CONTENTS

ABSTRACT

All human behavior representations (HBRs) simulate some aspects of people, either individually or as groups. HBRs differ from simulations of other complex phenomena by their knowledge bases, effectively sophisticated computer programs written in a knowledge representation language. As a discipline in simulation technology, HBR validation is still relatively immature with no theory, few tools and techniques and considerable but poorly documented experience. Two sources of information establish a firm foundation for the advancement of HBR validation technology, a broad experiential base and a more mature related field, knowledge-based system (KBS) verification and validation. HBR validators have learned many lessons from existing and future systems that deal with requirements, the subject matter expert-software engineer process, association of the HBR with the synthetic natural environment, and documentation. These lessons supply a rich source of guidance for future HBR validation activities. KBS

verification and validation is considerably more mature with a very large literature base to support it. HBRs, as a form of KBS because of their knowledge bases, can benefit significantly from this technology base. With these resources and increasing realization of its importance across the broader simulation world, HBR validation is poised to mature very rapidly.

INTRODUCTION

All human behavior representations (HBRs) simulate some features of people. Many simulations represent elements of human behavior, ranging from the characteristics of large groups to the detailed behaviors of individuals and teams. Any situation in which the presence of people can affect its outcome requires representing some aspects of human behavior.

Despite HBRs' ubiquity, people have practiced validating them only sporadically and that practice remains one of the more poorly understood areas of validation technology. Compared with other simulation-related disciplines, the existing technical literature base is relatively small but not nonexistent. One can easily meaningfully group the bulk of this literature into requirements [1-8], conceptual modeling [9-11], tools [10-15], and experience [16-20]. The majority of this literature was published in a conference specializing in HBRs. Most of the literature associated with this conference series deals with developmental issues. Perhaps not surprisingly, considerably more experience exists in HBR validation than this meager literature base suggests but little of this experience has been documented in the technical literature.

A much more mature field, validation of knowledge-based systems (KBS), encompasses technology that applies to those HBRs that represent the cognitive aspects of people [21]. These simulations are becoming more common and their capabilities are rapidly becoming more sophisticated as computing technology advances and users seek more predictive representations of the processes manifested by human organizations (e.g., command and control systems). As a result, the maturity of KBS validation should become more important to HBR validation since they share many important properties.

The survey of HBR validation in this paper begins with an overview of the basic concepts inherent to the field. It describes the nature and peculiarities of HBRs that complicate their validation. It also briefly considers the impact of these peculiarities upon validation activities. The next section captures the invaluable experience gained from validating the HBRs used in large scale military simulations. While anecdotal, this information forms an important foundation for any future evolution of HBR validation technology that no one should ignore. The last section surveys the breadth of KBS verification and validation technology. KBS validation defines the future of HBR validation since it is vastly more mature and deals with one of the more unique but difficult areas of HBRs, the knowledge base.

BASIC CONCEPTS

This section briefly covers the nature of HBRs and their validation. It considers the problems that make HBR validation appear different from regular simulation validation.

It concludes by showing how many of the mechanisms of validation (e.g., fidelity, referents, conceptual models) apply to HBRs.

General Issues

All HBRs simulate some aspects of people and nearly all complex simulations represent the behavior of people at one level or another, implicitly if not explicitly. As simulation technology and application advance, more and more simulation results will depend upon the output of HBRs. This trend makes understanding HBR validation imperative.

HBRs are unique among other complex simulations. At first blush, they appear distinguished from the other parts of a simulation by their

- Very high inherent complexity,

- Numerous nonlinear relationships all interacting chaotically over many different orders of magnitude, and

- Complex coupling with other parts of a simulation system.

However, simulated environment and nuclear effects models both face similar daunting problems. The real distinction of HBRs comes from their knowledge bases. An HBR's knowledge base really constitutes a computer program, in many cases a very complicated one, that a behavior engine executes to create human-like behavior. An HBR's knowledge representation defines its programming language. Any simulation system that represents different humans contains many of these computer programs within the simulation system's computer program. In addition to the non-human representations, developers must thus debug two sets of computer programs: the behavior engine (usually written in a language like C) and the knowledge base for each individual represented (written in the knowledge representation language). These facts added to the inherent complexity of HBRs easily make them the most complex components of a simulation system, even when compared to simulated environments.

People hold fast to many myths about validating HBRs. Table 1 summarizes a few of the most prevalent myths and identifies their underlying fallacies.

Table 1. Common Myths about HBR Validation.

Myth	Explanation
Users are good sources of HBR requirements.	Users may understand the people in the situations they want simulated but they do not usually understand what about that human behavior their simulation can ignore. Thus, even if they appreciate the need for an HBR in their simulation (not a common occurrence), they tend to state that need in unrealistic terms.
A good referent for an HBR is a human doing the same job.	Identifying the corresponding human as a referent for its simulation may seem like a good idea but everything about that human is probably not well known. Human referents also tempt one to expect its abstraction to perform exactly like that human rather than its abstraction.

Table 1. Common Myths about HBR Validation (continued).

Myth	Explanation
A valid HBR is as realistic (i.e., error = 0 in all property dimensions for all dependencies) as possible.	Again, HBRs are necessarily abstractions of real humans. We do not understand human nature well enough to accurately model all of human behavior. Like other simulations, HBRs should only represent what the purpose requires. Even if technically possible, a perfectly accurate HBR would be prohibitively expensive to develop and use
A good HBR is stochastic just like humans.	Human behavior appears stochastic due to its high complexity and chaotic nature. Some aspects of human behavior lend themselves to stochastic representation but treating all of human behavior as random ignores the goal-directed nature inherent to all humans, a key property.
A good HBR is logical just like humans.	Humans seldom behave logically. They can, however, explain their behavior logically but that explanation rarely agrees with the real phenomena underlying their behavior.
"Fair Fight" is a clear and testable criterion for HBR validity.	The success of the fair fight criterion depends entirely upon the observer and, thus, cannot be objective. This will lead to irreconcilable differences between observers about whether the HBR actually met the criterion. Good validation criteria are both observable and observer-independent.
The experts will recognize invalid HBR behavior when they see it.	Experts may recognize some invalid behavior when they see it but the complex nature of human behavior will lead to many false positives. Experts have often declared quirky HBR behavior as distinctly "human" when it actually resulted from implementation errors.
Validating HBRs is too hard so why do it or even try to understand it.	This defeatist perspective only leads to accepting poorly performing HBRs. Like any validation task, a reasonably simple discipline can produce acceptable and cost effective results. Good understanding of HBR validity can even simplify the difficulty of abstracting the parts of human behavior necessary to achieve a purpose thereby reducing the developmental costs and risks.

All of these beliefs are wrong and only hamper the practical use and advance of HBR validation.

Validating HBRs does face several challenging problems:

- HBRs tend to interact with complex environments.

- HBRs operate within very large behavioral hyper-spaces.

- HBRs inherently involve nonlinear behavior.

- HBRs often use oblique model representations (e.g., neural networks).

- HBRs couple effects over many orders of magnitude (a property shared with complex environment simulations).

Advancing technology will only aggravate these problems by enabling realistic complex HBR execution. But, these challenges are not insurmountable. They simply supply the grist for future research efforts.

Description of HBR Nature

As mentioned above, all HBRs model the behavior of people at some level of abstraction. So, HBRs can model any combination of the many different facets of human behavior including

- Ability to reason (e.g., knowledge based systems)

- Ability to change the environment (e.g., operating equipment)

- Responds to comfort and discomfort (e.g., environmental safety)

- Susceptibility to injury and illness (e.g., injury models)

- Emotional responses (e.g., affective models)

- Ability to communicate with other humans

- Abilities to sense the environment (e.g., vision models)

- Physical capabilities and limitations (e.g., MANPRINT)

The terms computer generated forces (CGF), semi-automated forces (SAF and SAFOR), synthetic forces, automated forces (AFOR) and command forces (CFOR) all refer to different forms of HBRs.

Figure 1, the HBR canonical model, depicts the basic components of a simulation of the neurologically-related human behavior, considered by many as generating the most interesting parts of human behavior [21]. In this model, the knowledge base consists of the executable dependencies needed to create the internal state representation from sensory input and to respond to that state. The knowledge base also includes the decision functions that determine when and which of those dependencies should be executed to achieve goals at any particular time or combination of stimuli. The behavior engine chooses the dependencies from the knowledge base appropriate to the current state and executes those dependencies to modify the internal state representation or to generate the actions to achieve the HBR's goals. The state representation depicts the HBR's dynamic assessment of both the internal and external world state including all goals.
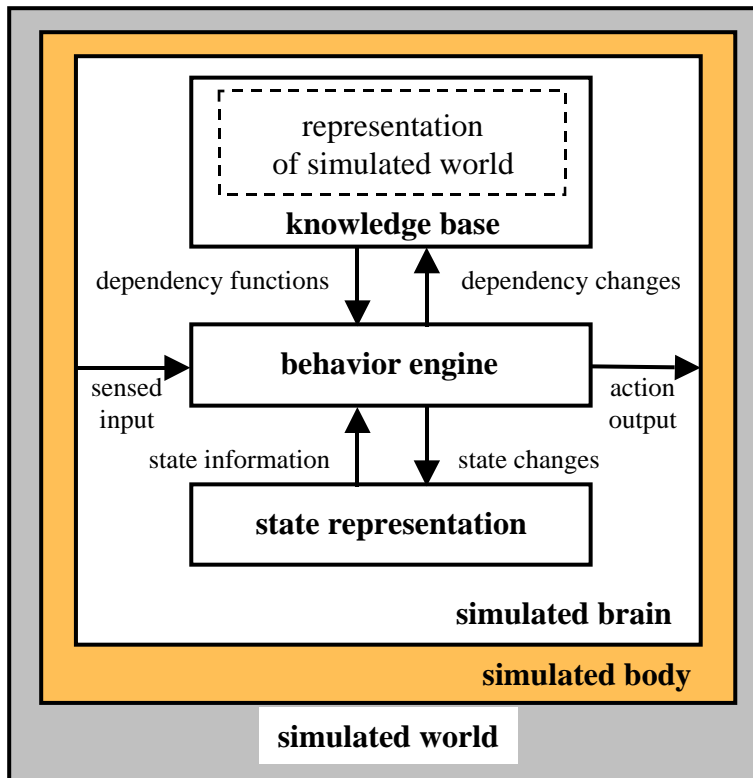
Figure 1. Canonical Model of Human Behavior.

The partitioning in this model creates the flexibility needed to represent the behavior of different individuals performing in different roles without requiring the building of completely new execution infrastructures each time.

All behavior engines perform essentially the same functions in one form or another. These include

- Accepting input about the state of the surrounding simulated world and using that information to update the HBR's internal state representation

- Evaluating the decision functions in the HBR's knowledge base, in the context of the internal state representation changes, to identify the executable dependencies relevant to the current situation and goals

- Executing the appropriate dependencies to change the internal state representation and generate the actions needed to achieve the HBR's goals.

In other words, in the context of this model, the only things an HBR can do to manifest its behavior are change the contents of its internal state representation, knowledge base and output. Behavior engines can serve other roles in addition to these essential ones. This discussion will address these in later paragraphs.

One can partition behavior engines into the few components illustrated in Figure 2.
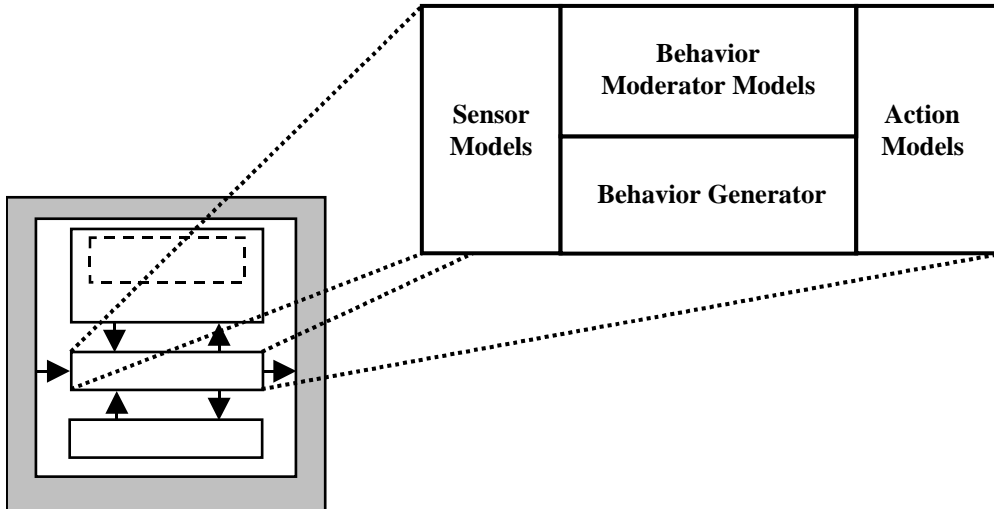
Figure 2. HBR Behavior Engine Partitioning.

In this partitioning, sensor models couple the HBR to the information available from the simulated world. They represent input modalities and bandwidths. Action models couple the HBR brain to the simulated world through the body's effectors (i.e., muscles). They represent the actions taken to change the simulated world state to achieve the HBR's goals. The behavior engine represents the cognitive processes through its interactions with the knowledge base and state representation. The behavior moderator models modulate the cognitive processes. Behavior moderator models represent the effects of non-cognitive processes upon the cognitive processes.

An HBR's knowledge base and its state representation share the same information partitioning illustrated in Figure 3.
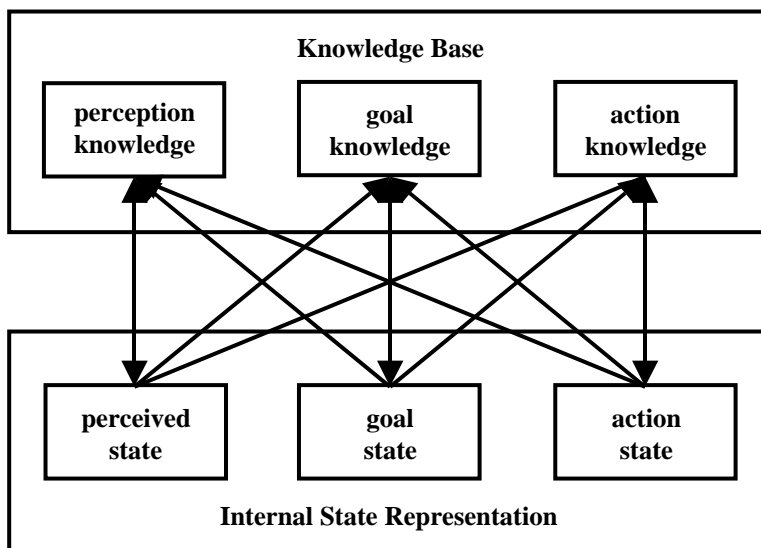

Figure 3. HBR Knowledge Base and State Representation Information Partitioning.

The arrows going from the internal state to the knowledge base represent independent property flows and those going from the knowledge base to the state representation are

the dependent property flows from the executable dependencies.

Finally, the behavior moderators modulate cognitive functions into realistic affective human behavior [22]. A behavior moderator is a condition that affects human behavior in ways other than those affected by the cognitive elements. Figure 4 shows a taxonomy of human behavior moderators.
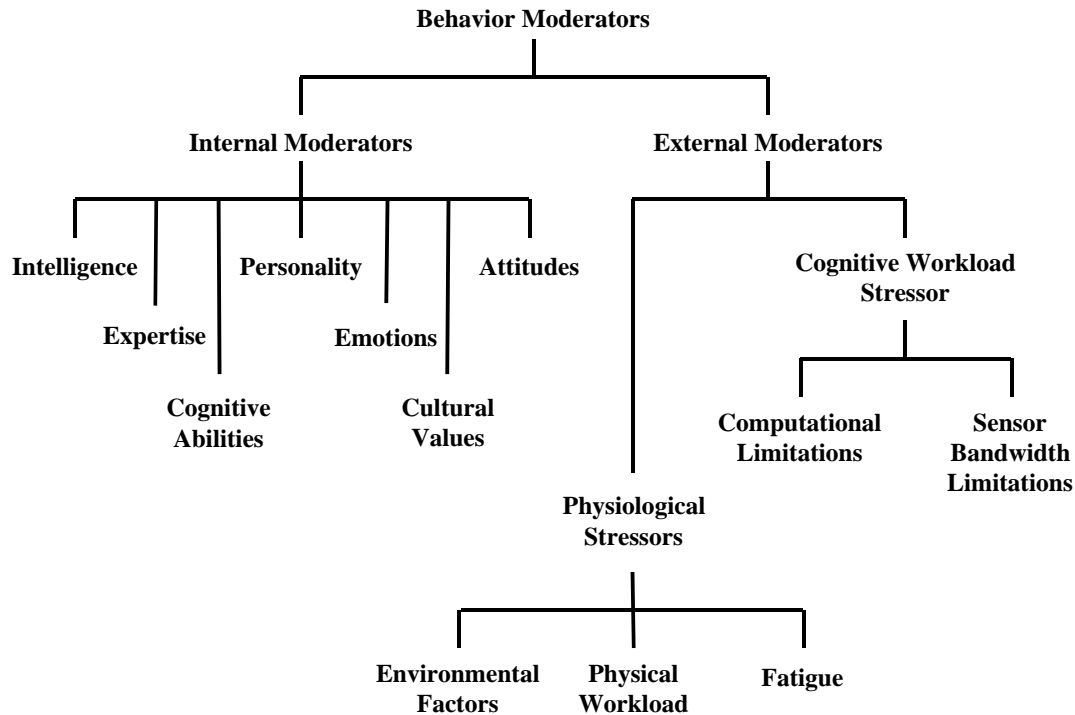
Figure 4. Taxonomy of Human Behavior Moderators.

Few consistent models exist that describe how the behavior moderator conditions actually change brain function and, in turn, human behavior. However, the community is actively pursuing this area of research and moderator representations do exist that suit the demands of some applications.

HBR Validation

In general, validation answers the question "How closely must the simulation resemble its simuland to achieve a purpose?" In effect, this question assesses the fitness of a simulation, whether of humans or some other phenomena, for a particular purpose. The reliability of this assessment and the answers it supplies depend upon the quality of three types of information:

- Validation criteria,
- Referent,
- Simulation capabilities.

Validation criteria come from the users' requirements. Several authors have examined the issues associated with HBR requirements and validation criteria [1-8, 20]. The

earliest of these explored the criteria for validating purely cognitive behavior [7], fewer than ten years ago. Since then authors have looked at the requirements for such specific applications as pilot training [1, 3], air defense [5] and sensor development [2]. Two authors [3, 4] have built their understanding of HBR requirements upon the foundations laid by Chandrasekaran and Josephson in their description of HBR fidelity through goal analysis [6]. Two authors have surveyed and categorized the requirements for HBRs in military simulations with various applications [4, 8]. These surveys have shown that requirements writers do not clearly understand the issues associated with specifying practical HBRs very well. Subject matter experts (SMEs) stand as one of the most important sources of requirements [8, 20] and several techniques from the broader area of knowledge acquisition apply well in eliciting this information from SMEs.

The referent comes from our best knowledge of the simulated phenomena. Reference [21] discusses several sources of referents, including SMEs, experimental data, human performance data and the technical literature describing human behavior. SMEs remain, by far, the most common source of referent information despite their inherent problems. While rich and broad, the applicable psychological and sociological literature contains many unresolved disparities thus forcing validators to choose one or another schools of thought, sometimes with little more justification than their intuition. Recently developed physiological techniques could help to close those gaps but this may take several more years. Until recently, human experimental data came from psychological and sociological studies. Some of the data from different studies correlates but much of it remains disparate. Observations of human performance in various tasks remain a relatively untapped resource, especially for military applications, although some have used these results [16, 17].

One can observe a simulation's capabilities in two places, its conceptual model and its results. HBR's uniquely provide one additional point of capability visibility, its knowledge base. Several authors have described their developing conceptual models for HBRs [9-11]. Their methods include acquiring the knowledge for their models and organizing that knowledge into some coherent form. Not unlike the issues of conceptual modeling in conventional simulations, conceptual modeling for HBRs has not evolved toward either a consistent form or content although ongoing efforts may achieve that goal. Further, the HBR community has not yet agreed upon a single definition of a conceptual model, a fact that continues to hamper effective communication within the community.

In general, few developers currently build anything resembling a conceptual model for their HBRs. However, most developers exercise some form of results validation. Considerable experience exists in this respect; only some of which has been documented in the accessible technical literature [16-20]. Most of the current results validation relies upon SME opinions [14, 18, 20] although some efforts have compared HBR results against the results obtained from human-in-the-loop simulators [16, 17] and against the predictions of psychological models [19]. Getting all three types of information necessary for validation with the quality needed remains a challenging task, especially for simulations of such complex phenomena as HBRs.

The SISO Glossary of Fidelity-Related Terms defines fidelity as "the degree to which a model or simulation reproduces the state and behavior of a real world object or the

perception of a real world object, feature, condition, or chosen standard in a measurable or perceivable manner; a measure of the realism of a model or simulation; faithfulness" [23]. Recent concepts of simulation fidelity [24-27] create the language through which to describe both validation criteria and a simulation's capabilities in easily comparable terms. Understanding and describing HBR fidelity serves several purposes:

- It identifies specific terms with which to characterize HBR requirements and capabilities.

- It takes a step toward making quantitative evaluation of HBR validity possible.

- It makes HBR validation more objective and repeatable.

- It contributes to other aspects of HBR development and employment.

- It creates an important database of information about HBR capabilities that could be used to identify applications for which it might serve appropriately (i.e., reuse).

- It supplies the means to evaluate the consistency of the elements within the knowledge base and between the knowledge base and the HBR's behavior engine.

- It suggests ways to organize information about human behavior so as to better serve as referents for HBR validation.

Several other authors have considered the issue of HBR fidelity [6, 28-30]. Unfortunately, these views do not agree completely on the precise components of fidelity. Harmon and Youngblood [30] describe how to interpret fidelity for HBRs from the SISO Fidelity Conceptual Model [25]. Since this view is consistent with a definition that applies broadly to all types of simulation, regardless of purpose or form, the remaining discussion will use that definition. The discussion below summarizes this description.

Everyone knows that, unique among systems, simulations abstractly represent the behavior of something else for some purpose. Abstraction means that simulations must omit some of the details about the things they model from their representations. In other words, simulations only resemble their simulands. Therefore,

- Modeled objects $\subset$ real world objects

- Modeled dependencies $\subset$ real dependencies

- Modeled independent properties $\subset$ real independent properties

- Modeled dependent properties $\subset$ real dependent properties

- Modeled independent property domains $\subset$ real independent property domains

- Modeled dependent property ranges $\subset$ dependent property ranges

- Modeled dependencies $\approx$ real coupling phenomena

These abstractions of the real world represent the resolution or level of detail of the simulated world. This and the other fidelity components [24] describe the degree that a simulation abstracts its simuland. Figure 5 illustrates the result of the abstraction process when creating an HBR.
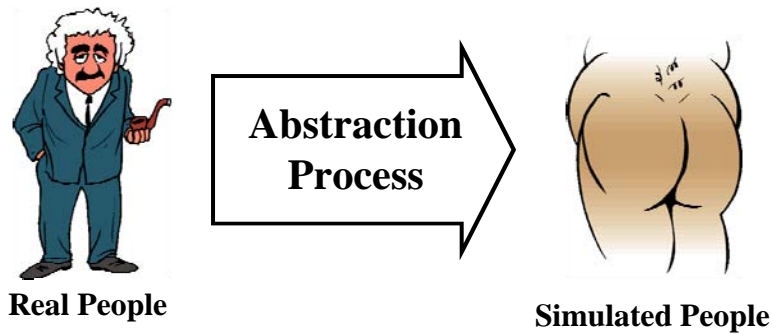
Real People   **Simulated People**

Figure 5. All HBRs Abstraction the Characteristics of the People They Model.

Dependency characteristics describe the amount of functional abstraction in a model. Figure 6 illustrates this part of the abstraction process.
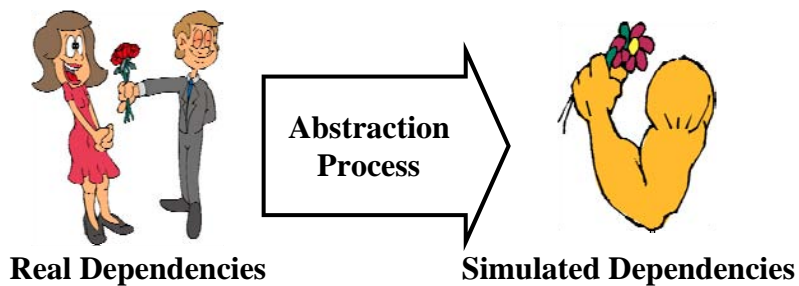

Real Dependencies   **Simulated Dependencies**

Figure 6. Dependency Abstractions of Real People's Behavior

The notions from the fidelity description specify the following parameters for describing a dependency.

- independent property sensitivity - minimum degree of independent property change necessary to produce a change in a dependency's dependent property state

- dependent property precision - minimum degree of dependent property state change that a dependency can manifest

- dependent property error - degree to which a dependency's behavior, as reflected in the state changes of its dependent properties, deviates from the behavior of the real phenomena it approximates

While additional characteristics may exist, those proposed above provide a good starting point for rigorously describing simulation fidelity, the vocabulary for describing simulation representational capabilities.

From this perspective of fidelity, few simple steps can help one gauge an HBR's fidelity:

1. Identify the representations of object properties and dependencies

2. Characterize the domains and ranges of those dependencies

3. Characterize dependency sensitivities to their independent properties and the precisions of their dependent properties

4. Characterize the errors of the dependency output relative to the behavior of some

11

referent

One can apply these steps to an HBR as either a black box or a white box. When dealing with an HBR as a black box, one can only assess the validity of its results. The nature of its design and knowledge representation obscure its correspondence with its referent thereby currently making validation at those intermediate levels uneconomical to impossible. Ongoing efforts to automatically explain the nature of inaccessible knowledge representations (e.g., neural networks) may enable treating these HBR designs as white boxes in the future.

White box fidelity applies to HBRs using more accessible knowledge representations (e.g., production rules). Interpreting HBR fidelity when treated as a white box begins by assessing the fidelity of its primary components:

- Knowledge base

- Internal state representation

- Behavior engine

Evaluating the fidelity of an HBR's knowledge base and internal state representation involves decomposing its knowledge representation into its representations of state (declarative knowledge) and behavior (procedural knowledge) then comparing those components against the referent [29]. The only trick in doing this involves remembering that the referent for the perceptions must come from the nature of the simulated world with which the HBR interacts and not strictly from our knowledge of relevant human nature. The knowledge-based systems verification, validation, evaluation and testing world provides an enormous resource for assessing knowledge base validity. A later section of this paper surveys this technology.

Determining the fidelity of HBR behavior engines involves somewhat more effort. One can organize HBR behavior engine dependencies into three broad function groups:

- Knowledge base element execution (e.g., theorem proving, condition matching, and conflict detection and resolution)

- Emotional effects manifestation (e.g., computing emotional state, determining emotion influences upon performance limits, and computing influence probability distributions)

- Performance limit representation (e.g., sensor and computational bandwidth constraints upon the observed behavior)

Behavior engine dependencies only include the functionality that manifests emotional effects and performance limitations when their HBR designs do not embed those representations into their knowledge bases.

After using fidelity concepts to describe HBR capabilities and validation criteria in equivalent terms, assessing HBR validity only involves comparing those descriptions to determine if the HBR meets the validation criteria. Letting the accreditation authority, if one exists, know where the deviations lie permit their determining the fitness of the HBR for the application or class of applications. Figure 7 shows how the flow of information from several sources fuels the basic HBR validation tasks.
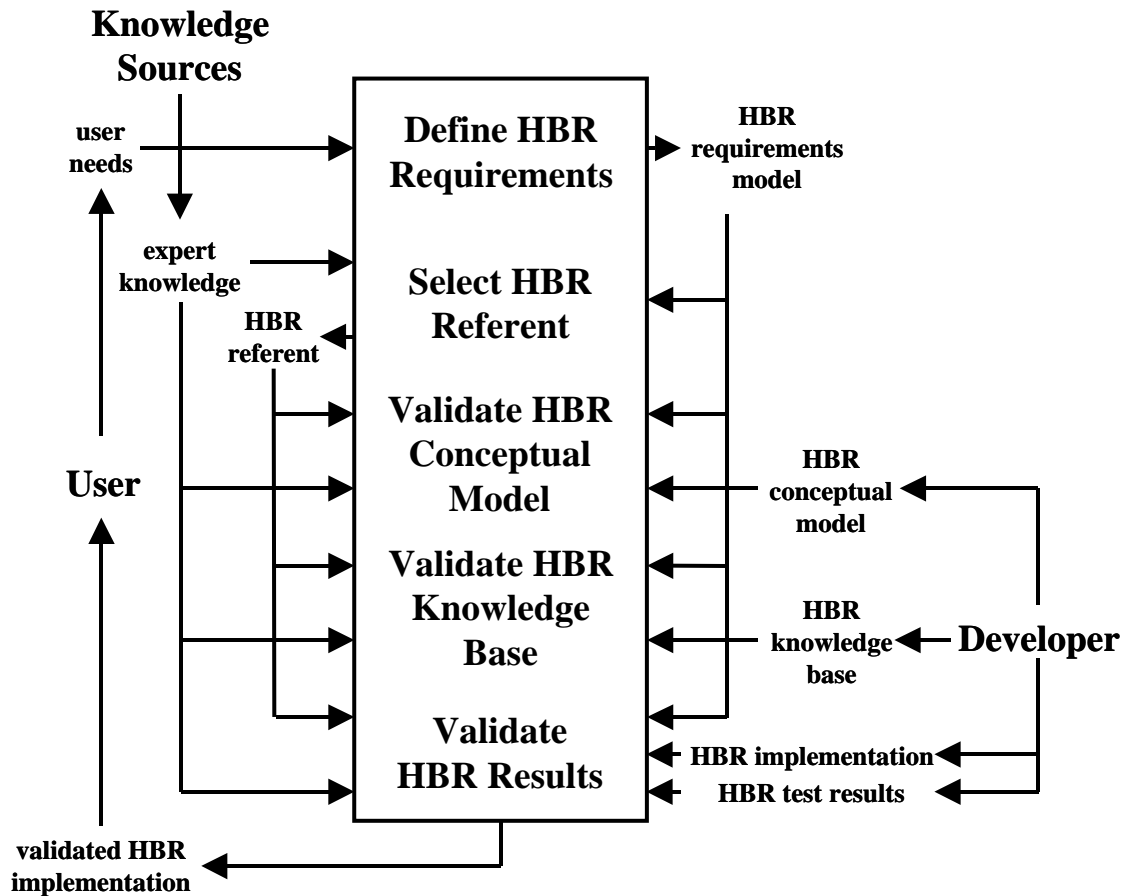
Figure 7. HBR Validation Task Information Flow

An HBR validation plan should address all of the components of HBR validation including

- Requirements development
- Referent development
- Conceptual model validation
- Knowledge base validation
- Results validation

All validation efforts, including those for validating HBRs, simply produce information, hopefully truthful information. Each component section should describe its

- Approach for collecting and assembling the information and the component activities
- Sources of input information and their management
- Approaches for combining and analyzing source information and the component activities

13

- Possible problems that could be encountered in the activities and the criteria for recognizing them

- Contingencies for dealing with possible problems and mitigating risks

- Activity products and their dependencies

- Approximate activity schedule

- Information produced by each component

The HBR validation plan should address each of these information needs for each of the components listed above.

Despite its correspondence with general validation technology, HBR validation is quite an immature field. Table 2 summarizes the state of the art in terms of its theory, tools and techniques, experience and experimental data.

Table 2. Brief Assessment of the State of the Art of HBR Validation.

Factor	Maturity
Theory	- Almost none, if any, on validation of HBRs specifically - Considerable psychological, sociological & physiological theory on how actual humans behave under various circumstances - Considerable theory on testing/observing human and group behavior (some of which may be useful)
Tools & Techniques	- Some tools for generating execution traces for some specific HBRs (e.g., Soar) - Some tools to display observable HBR behavior (e.g., PVDs) - Very few tools to support HBR validation - Many tools for KBS validation
Experience	Some direct experience in validating particular HBRs for various purposes (e.g., Soar, ACT-R, HOS, MicroSaint, ModSAF)
Experimental Data	- Much on actual humans performing a variety of tasks - Much from psychological, sociological and physiological experiments on actual humans

Table 3 depicts the same situation in a different way. It summarizes the resources that are available for each of the activities and each of the components of the HBR.

Table 3. State of the HBR Validation Technology Currently Available

Validation Component	Conceptual Model	Knowledge Base	Observable Behavior
Requirements Characterization	few disparate methodologies	limited formal languages	SMEs
Referent Description	SMEs, physiological, psychological & sociological models	SMEs, documentation	SMEs, experimental data

Table 3. State of the HBR Validation Technology Currently Available (continued).

Validation Component	Conceptual Model	Knowledge Base	Observable Behavior
System Characterization	few disparate methodologies & forms	KB languages, NL explanations	observable behavior, explanation traces, performance meas.
Comparison Techniques	nothing specific	SMEs, KBS VV&E tools & techniques	SMEs, KBS testing tools & techniques

A few tools have been developed specifically to support HBR validation [12-15] and reported in the technical literature. All of these aim at aiding the results validation phase through automation. All of these tools apply to very specific systems and have not yet received wider application. By comparison, researchers in the cognitive science community have developed over 80 tools to support various phases of KBS validation [21]. This technology base provides a much more well established resource for HBR validation for the immediate future.

Several HBRs have been developed and validated for a variety of purposes [16-20, 22]. A recent National Research Council study provides an excellent survey of HBR technology [22]. These study results discuss the validation of many of the existing and developing HBRs. Table 4 recasts this information to summarize and compare the different approaches to validating HBRs discussed in Reference [22].

Table 4 shows that developers have applied primarily three types of referents to validating their HBRs: domain, psychological and physiological referents. They evaluated against referents from both human expert assessments and human performance data for a diversity of applications. Table 4 differentiates SME referents from experimental data referents in the last column as human interaction data referents and human behavior data referents, respectively. They evaluated against several different psychological referents, as well, including validated theory, experimental data and human interaction (a Turing test equivalent). Table 4 shows that integrated models of human behavior are most likely to be validated against psychological referents. Only neural network approaches to HBR have seen widespread validation against physiological referents. However, this validation has been limited to the relatively constrained performance of nonspecific neurons.

All of the techniques applied to validating existing HBRs have significant limitations. As mentioned, using domain referents requires unrealistic searches of very large and nonlinear behavior spaces. Using psychological and physiological referents requires extensive validated models of psychological and physiological phenomena. While many comprehensive psychological models exist, relatively few of them have been applied to HBR validation, especially for simulation applications. Like the physiological models, many psychological models deal with very restricted behavior spaces. These limitations prevent their useful application to HBRs representing behavior for realistic situations. As psychological and physiological models become richer and more consistent, their utility for HBR validation will increase. As with models and simulations of physical systems, validation must be done against referents at different abstraction levels. Only consistent

Table 4. Comparison of the Validation of Different HBRs.

System	Domain Referent	Domain Types	Psychological Referent	Physiological Referent	Referent Data Sources
ACT-R	X	submarine TAO & Aegis radar operators	X		human behavior data
COGNET	X	anti-submarine warfare			human behavior data
EPIC	X	computer interaction tasks	X		human behavior data
HOS			X		validated theory
Micro SAINT	X	helicopter crew, ground vehicle crews, C2 message, tank maintenance & harbor entry operations			human behavior data
MIDAS	X	757 flight crew			human behavior data
Neural Networks			X	X	validated theory, human behavior data
OMAR			X		validated theory, human interaction
SAMPLE			X		validated theory
Soar	X	air traffic control, test director, automobile driver, job shop scheduling	X		validated theory, human interaction, human behavior data
ModSAF	X	ground warfare			human interaction
CCTT SAF	X	ground warfare			human interaction
MCSF	X	small unit operations			human behavior data, human interaction
SUTT CCH	X	small unit operations	X		human behavior data, human interaction
IFOR (see Soar)	X	fixed & rotary wing air operations	X		validated theory, human interaction, human behavior data

results between these different levels can guarantee validity. At this point, and probably forever, no single level of referent testing should be sufficient for any application.

In general, immature technology and referent sources hamper HBR validation. Despite these shortcomings, HBR validation remains paramount in HBR development and use. Their great complexity and nonlinear nature creates the omnipresent danger that casual assessment of their validity will mistakenly classify erroneous behavior as correct. Only systematic validation can help to assure that the causal errors so prevalent in real human behavior arise from designer intent rather than from a flaw in the knowledge bases' or behavior engines' functionalities.

HBR VALIDATION LESSONS LEARNED

HBR validation experiences with military models and simulations (M&S) are replete with various approaches to instilling the discipline of doctrinally valid procedure among seemingly unruly computer generated forces. It is a tribute to the complexity of modeling even simple human behaviors, that so many different approaches have been attempted. In encounters with several M&Ss, from initial development through preparation of legacy systems for an application, there are some empirically learned lessons, for which if accounted in the initial planning, could improve both the efficiency and first-time success of valid HBR implementation of a new development and the efficiency and accuracy of an HBR validation of a legacy system.

There are four overarching areas that could affect a project's success: requirements, an SME-software engineer process, synthetic natural environment (SNE) and documentation. Within these areas orbit a number of hard-won lessons for consideration.

Requirements

In general, experience in several programs has indicated that requirements definition in Army M&S requirements documentation does not adequately reflect the intended uses. Information to support modeling decisions and therefore validation activities is not easily derived from most user requirements documentation. The users know they want a tool that simulates military activities not easily or cheaply encountered in reality, but how they intend to use the tool will influence its development. The users repeatedly have shown they have had difficulty in enunciating the specifics of their uses. Adequate time to execute a detailed use analysis, especially on programs such OneSAF with a vast array of intended applications across the three Army M&S domain users, and appropriate experience and skills at the levels that produce the requirements affect the user in developing a clear focus of exactly how the planned M&S is to be used.

The state of the user not knowing what is required is not necessarily a static limitation that can be addressed by iterative interaction between developer and user. It can tend to be a dynamically changing problem. Initial user direction of the requirements can be changed as a result of developer analysis. Usually these changes are catalyzed by newfound user clarity of what is wanted and not wanted, resulting in changes to the detriment of the very analysis that fostered the enlightenment. In effect, development resources are being used to define user requirements. Sometimes additional requirements may be added that are obvious extensions beyond the scope of the initial requirements documents, creating additional midcourse changes, as the user attempts to leverage in previously unconsidered applications. The WARSIM program is a recent example of such a fluid and evolving requirements environment in an in-progress development when it had to consider the incorporation of JSIMS.

There are some cases where articulating high-level and vague requirements is a systemic requirements writing approach to catalyze the creative faculties and expertise of competing bidders in an attempt to maximize the M&S functionality for the Army's development dollar. This takes advantage of the experience of those software houses with multiple development experiences to actually expand the possibilities of the development as a function of their experience from those multiple programs and avoids limiting the developers' innovativeness to the Army's detriment by providing a rigid

structure. In this case the Army purposely does not define a standard, requiring the developer to define one for its approval. This is the standard approach practiced by STRICOM and under which the WARSIM and OneSAF programs are being executed.

Within such environments, HBR specification and validation seeks solid ground. The impact of the above on HBR validation, whether done by omission or commission, is that required HBR effects and measures of performance that will satisfy intended M&S applications are specified late or not at all. The HBR effects and measures of performance are often indirectly derived as a function, iterative rework and face validation between software engineer and SME. At a minimum, this is expensive and inefficient, but it also may put application accreditation at risk because HBR effects and performance measures required for the intended applications are not specifically identified. The HBR implementation evolves into whatever the SME and software developer are able to produce within the project time and schedule. As seen below, the lack of specificity up front will ripple to other aspects of HBR validation.

HBR knowledge acquisition (KA) is usually viewed as the hand-off from the military domain to the software implementation domain and is the tool to define and validate HBR representations within the M&S and for the application. In spite of all of the military inter-domain success of producing and validating these documents, the documents often fail to provide adequate, intra-domain HBR modeling information. The over-arching element missing in the production of these documents is the focus on providing the information necessary to model the effects and performance measures required by the application, and likewise to set the standards focus for effective validation assessment. Without that type of focus, the documents are domain accurate, broad breadth essays describing military activities as opposed to technical effects and performance responses to questions within the context of a doctrinal military setting. In other words, without M&S requirements as the pre-eminent driver, KA documents are developed to without bounds and tend to attempt to describe reality. Any M&S implementation-specific useful information found within them is merely a serendipitous occurrence and not one of design. A third order effect of the lack of focus is they are written broadly and at too high a level, using military domain parlance from which software engineers must infer meaning. The focus on implementation and application versus reality may appear subtle, but it is actually huge and insidious because the impact is discovered late, creating program risk. The failure of these documents to provide the right information for the software domain becomes evident during implementation, causing desk-side, undocumented informational exchanges to fill the gaps of the KA documents. The decisions of these stopgap procedures tend not to go through a validation process.

Because the required application effects are unknown, the producing and validating SME's conduct their activities without regard to them. The documents easily support validation by designated authorities because they were written with regard and doctrinally assessed relative to reality and independent of implementation. If HBR effects and performance measures were more a part of the validation question, the validating SMEs would have been making their assessments within the context of implementation. Assuming they were implementable, they would have been the representations that had been validated.

The above might be hotly contested by many in the M&S development community who have worked on programs such as CCTT or WARSIM because of the extensive KA production that was done for these programs. In fact, dedicated teams of professional KA researchers produced extensively detailed, well-illustrated documents describing military tasks. They brought together in a single document the essential procedures and considerations of a military task from multiple authoritative military sources. They coordinated with software engineers, gaining insights for formatting their military documents in the most beneficial way, iterating the format several times to derive the best format, producing documents such as the Combat Instruction Set (CIS). They had the content of these documents validated by other SMEs, representatives from the appropriate Army branch schools. However, all of these measures are meaningless if the modeled implementation does not reflect their content. The validation is meaningless or the implementation is invalid because the modelers were not able to achieve the content and intent of the HBR KA artifact. As a result, when military decision-makers learn of behavioral implementation limitations relative to their understanding of the HBR KA documents, they conclude the process of validation or implementation has failed them.

The lack of application effects and performance requirements makes any implementation subject to criticism from any SME. One SME can declare an HBR valid and the next declare it invalid. Since no HBR is a perfect representation of reality, every HBR is vulnerable to an invalid assessment. This was a problem for ModSAF in preparation for STOW-97. The effects and performance requirements were unknown for STOW, and the effects and performance requirements for the HBR implementations were unknown for those versions of ModSAF. It was easy for validators to declare ModSAF HBRs invalid because they had no choice but to compare them to reality.

Summary of Requirements Lessons Learned.

- Poor HBR requirements specifications create a domino effect of barriers to achieving HBR validity.

- Validation decisions cannot be defended without the effects and performance requirements that drove the implementation.

- Requirements specifications usually do not provide effects and performance measures.

- Some acquisition approaches purposely avoid specific requirements.

- Lacking any other specification, HBR KA attempts to describe its domain from a reality perspective, using its parlance.

- HBR KA is too general in description, creating documents requiring developers to infer meaning and make choices of importance.

- Development can never achieve reality.

- Users and decision-makers incorrectly assume validity is related to reality instead of application requirements.

SME-Software Engineer Process

The ideal process for efficient and economic software development is validation of HBR conceptual models before implementation. These are then verified as implemented properly. The reality is that validation is usually not achieved until after implementation. Most providers of military KA are unaware of how to develop information that supports software implementation. The shortfall is not merely an issue of inadequate description but a difference in language. The software engineer 'speaks' in quantitative terms, mathematics tied together with logic, easily followed and explained but limited in expression and nuance. The military SME uses qualitative terms, 'English' descriptions that are rich in expression and nuance further complicated by both common terms loaded with insider meaning and military unique terminology. Most software engineers have no or limited military experience, and most military have no or limited software background. Without an explicit process designed for the purpose, it is almost impossible for one to write for the other and transfer the critical information necessary to model. The shortfall is usually resolved when implementation begins, during which software engineers, in trying to develop software requirements specifications, scramble to execute a variety of informal and undocumented information gathering efforts with SMEs or, when time constrained, make their own best interpretation of the KA artifacts. Then validation occurs concurrent with implementation or after it is completed.

For this well-known M&S development problem, several different approaches have been attempted by different programs to preclude it. The CCTT program introduced the CIS, WARSIM evolved into the ODO, and OneSAF developed the BKAD. These and similar HBR document templates attempted to identify critical information categories that military SMEs would attempt to populate. These documents fell short in two areas. The SMEs continued to use their language and the documents did not drive the KA SME to the level of specificity that is required for implementation.

More an art than science, the task is difficult. The OneSAF program chose to eschew its own format attempt, the BKAD, and adopted a software engineering approach to developing KA products. It initiated a program to train its KA SMEs to decompose military tasks into basic generic processes. Once described, the KA SMEs must conceptualize and define quantitative expressions of these processes. The OneSAF program then put the responsibility on its corps of validation SMEs to validate the processes and their quantitative representations relative to OneSAF requirements. It was not enough to say something was invalid from a doctrinal perspective. Defects against these processes had to be justified relative to OneSAF requirements documentation. The OneSAF user representatives would then adjudicate defects. This process is similar to that developed for the JWARS program.

The OneSAF KA approach takes the onus off of the software engineers to interpret KA documentation by providing them with implementable specifications. It also facilitates validation. What is validated is what is going to be modeled. If the KA represents the needs of the OneSAF program, when validated, the validation will be with respect to the program needs. However, the OneSAF program has not specified the HBR required effects and performance measures, so it remains to be seen if the KA effort is producing representations adequate for the known application goals of the program.

Summary of KA-Software Engineer Process Lessons Learned

- Regardless of the quantity of information produced by military KA SMEs, the descriptions tend not to support the requirements of software development. In fact, the larger the quantity of information, the more difficult it is for a software engineer to infer the essential elements that need to be implemented.

- Validation of such KA documents has little relationship to M&S HBR validation, for the software engineers will analyze them and develop a document that suits implementation. If this approach was the only means to transform military domain information into software domain information, the software engineer's document is what should be validated. It represents what will be modeled.

- The more that KA documentation development processes are improved to support implementation, the greater the value of the HBR validation activities of those documents.

Association of HBR with the SNE

Even if all of the problems described above are resolved, a perfectly defined HBR task, validated against application requirements, and implemented properly may not perform validly if the synthetic natural environment does not support the modeling implementation.

ModSAF has an excellent occupation of battle position implementation that finds primary and alternate positions with cover and concealment, and hide positions. When a user selected battle position fails to provide adequate terrain to meet positioning requirements, the implementation distributes the members of the unit equally over the designated area equidistant from one another. What was discovered is that the implementation opted for the default 97% of the time. After examination it was determined that the terrain database did not support the implementation. The implementation was rarely exercised.

The OneSAF program plans to have multiple terrain resolutions. It rapidly became apparent that one implementation of a behavior that successfully operated on one resolution of terrain may not be adequate for another terrain resolution. OneSAF faces the likelihood of having to have more than one version of the same behavior. For example, KA was producing detailed descriptions of formation position keeping. The process was depending on intervisibility checks between entities; failure to maintain intervisibility would cause the entities to reduce their spacing until intervisibility was achieved. It was pointed out that if this methodology was played on large terrain cells with constant vegetation higher than entity sensors and that thereby denied intervisibilty, it was conceivable that all of member entities of the unit would continue closing in on another until they were literally located on top of each other.

Summary of association of HBR representations with the SNE

- Evaluate the SNE against the HBR requirements.

- Many of the anomalies with regard to the interaction of HBR and SNE cannot be anticipated.

- Adequate results testing against SNE should be planned.

Documentation

Well-documented HBR KA documentation is a critical component to future M&S application assessments. Without it tremendous time can be taken in backwards engineering or great risk taken in not adequately evaluating HBR functionality. For example, in the early versions of ModSAF, there was almost no documentation. In assessing HBR for STOW-97, there was no way to evaluate behavior performance except by face validation. Multiple test designs had to be exercised to establish the limitations and capabilities implied by the instantiated HBRs. Anomalies were found that could not be explained. The only basis for comparison was relative to reality because the effects and performance measures for which they were designed were unavailable.

It is a fact that integration of different representations will affect modeling decisions, for example, the terrain limitations driving possible changes in the KA of formation position keeping discussed above. When the KA documentation is discovered not to serve implementation, the software engineers will find some means to find the information they need to complete the implementation. This is usually done informally and quickly because development schedules usually do not provide for KA rework resources. Because the program must take the necessary measures to be a success, the historical trace of the HBR development is lost. Documentation is not updated to reflect what was actually done. The documentation itself is no longer valid. This impacts the usefulness of that documentation for future verification and validation (V&V) and accreditation activities. It can cause the user of that documentation to make misguided decisions.

The OneSAF program is posed to capture changes to its HBR KA by instituting an overt process for capturing the changes in the KA documentation. It is anticipated that the KA documents will serve the implementation effort, but there will be implicit, explicit, integration and performance modeling issues that may drive changes to the way the KA is implemented. The process is set up to capture these changes and maintain documentation faithfulness to the implementation.

<div align="center">

Summary of documentation lessons learned

</div>

- Perfectly formulated HBR KA documents will not always be followed due to implementation considerations.

- There usually is not, and there must be, a means to capture in-process decisions that depart from the previously validated HBR KA documentation.

- Documentation is critical and must be valid for future validation and accreditation assessments.

KBS VERIFICATION & VALIDATION TECHNOLOGY

It is true that different communities within the computer software field define validation and verification in somewhat different ways. The traditional knowledge-based systems community is no different. Even within this community, there has been significant confusion about the meaning of these terms [31]. We will adopt here their definitions for validation and verification of traditional knowledge-based (expert) systems.

KBS Verification

Gonzalez and Barr [31] define verification of traditional knowledge-based systems as the process of ensuring that the intelligent system 1) conforms to specifications, and 2) has a knowledge base that is consistent and complete within itself. One major advantage of verifying the internal consistency and completeness of the knowledge base is that it can be done without exercising the system, a process that lends itself to automation. Numerous approaches have been developed for automated verification of intelligent systems, particularly in the area of rule-based systems. In general, these approaches focus on determining consistency and completeness and identifying anomalies within the intelligent system. This section discusses the background of and the early efforts in verification of rule-based knowledge-based systems. We focus on rule-based systems because of their popularity and success. Furthermore, we will focus on the efforts made at automating the verification and (to a slightly lesser extent) validation processes.

Suwa et al [32] are credited with one of the earliest works in automated verification. Their contribution in automated verification systems was the static rule checking utility developed as part of the ONCOCIN system. ONCOCIN's verification capabilities are divided into checking for inconsistencies (defined as conflict, redundancy, and subsumption) and incompleteness (defined as missing rules). An automated rule checker is used which displays potential errors. An expert is then asked to determine which of these potential errors are in fact actual errors. A weakness of this approach is its limitation to identifying problems at the rule level (i.e., it cannot identify problems that are the result of longer reasoning chains, such as redundancies resulting from several inference steps). Furthermore, the system's computational complexity is rather poor, making this approach only useful for a small set of parameters [33].

Another well-known example of automated verification via static analysis of rule-based systems came only a few years later. Nguyen et al [34, 35] developed CHECK to verify the consistency and completeness of knowledge-based systems built using the Lockheed Expert Systems development environment (LES). While the algorithms used in CHECK could theoretically be applied to rule-bases developed in other environments, the CHECK program itself could not be. Operation of CHECK is based on the construction of a dependency chart, which shows the dependencies among rules and between rules and classes. CHECK identifies the following rule-base errors (or, more accurately, potential errors): conflict, redundancy, subsumption, unnecessary IF conditions, circular rule chains (consistency checks) and un-referenced attribute values, illegal attribute values, unreachable conclusions, dead-end goals, and dead-end IF conditions (completeness checks). The complexity of CHECK appears to be better than that of the ONCOCIN verification system, but the tool is not suitable for general use since it depends on the rules being set up in a particular fashion.

In 1987 Cragun and Steudel [36] developed a system called the Expert Systems Checker (ESC). ESC makes use of decision-tables to check the completeness and consistency of a knowledge base. ESC looked for discrepancies, ambiguities (including conflict), redundancies, and missing rules. Conflicts and redundancies were determined in quadratic time, followed by a completeness check. If there were no conflicts or redundancies, then the completeness check would be carried out in linear time. However, if conflicts or redundancies were found then the completeness check required exponential

time.

KB-Reducer [37, 38] was designed to reduce knowledge bases to more concise contents without sacrificing correctness or functionality. However, it could also be used as a verification tool to check rule bases for inconsistency and redundancy, including contradictions and redundancies that are the result of inference chains, not just pairs of rules. An extension to KB-Reducer called KBR3 [39] was designed to assist in the maintenance of large knowledge bases. KBR3 is based on an 'application-neutral' language into which the knowledge base must be translated before it can be processed.

COVER (COmpleteness VERifier) [33; 40; 41] was the next tool in the horizon for verification of rule-based systems. COVER carries out seven verification checks: redundancy, conflict, subsumption, unsatisfiable conditions (rules which cannot be fired, missing values), dead-end rules, circularity and missing rules. The rules must either be written in or converted to a language based on first-order logic, and COVER must be given the set of final hypotheses (classes), as well as information about any semantic constraints.

Zhang and Nguyen [42] used a Pr/T net representation to do static analysis of a rule base. However, this work does not take certainty factors into account, and it also does not handle negated information in the rule-base.

Valiente [43] presents a method for redundancy and subsumption detection based on a hypergraph representation of the rule-base. His premise is that hypergraphs provide a more compact representation for rule-bases than do graphs, and they also facilitate use of graph transformations based on graph grammars and algebraic graph transformation. His method is restricted to antecedents with conjunctions, and any disjunctions in a rule must be rewritten using multiple rules.

The above systems all have one thing in common – they look for inconsistencies and incompleteness in a rule base without exercising the rule base. While other such systems exist, the above are the main ones described in the technical literature since the concept was originally conceived in 1982. Now let us take a look at efforts to automate the other aspect of verification - ensuring satisfaction of specifications.

VSE and VSE-II [44] are general-purpose verification systems that show promise for a range of software systems, including intelligent systems. VSE can be viewed as a CASE tool that supports software development from the earliest stages through code generation. System developers use formal representations of system properties and system requirements to aid the development process, and VSE is invoked to prove that specified requirements hold on the knowledge represented in the system. The core of VSE is an interactive inductive theorem prover, which uses full first-order predicate logic based on abstract data types and syntactic components that represent temporal constraints. VSE has been used for development of systems that handle robot control within a nuclear power plant, control software for a North Sea storm surge barrier, and smartcard operation software. Current work on VSE-II will tailor it towards the special needs of particular application domains, such as e-commerce systems.

Other rule checkers exist that generally perform the same function of formally proving that specified requirements hold.

KBS Validation

Once again citing Gonzalez and Barr [31], they define validation as the process of ensuring that the output of the intelligent system is equivalent to that of human experts when given the same input. This typically requires that the knowledge-based system be exercised with test data. The challenges here are how to select the test data (test cases) such that they in fact "cover" the entire system without being exhaustive and how to determine the correctness of the solution presented by the knowledge-based system for the test inputs. The latter frequently requires human expertise (an "Oracle") to serve as the judge of the correctness of the system's response. For these reasons, KBS validation has not lent itself to automation as well as has verification.

In order to determine whether or not the output of an intelligent system is equivalent to that of a human expert, we must interrogate the intelligent system with test data. Therefore, all validation systems are characterized by some mechanism that facilitates review of system results on test data. Perhaps the earliest dynamic analysis method for validation of an intelligent system was TEIRESIAS [45], developed for use with the MYCIN system. TEIRESIAS allows the rule-base developer to find in the rule-base the errors that led to incorrect conclusions. TEIRESIAS is both a debugging tool and a knowledge acquisition tool, allowing the alteration, deletion or addition of rules in order to fix an error. TEIRESIAS shows the user the reasoning used by the system to reach a conclusion, and the user can then approve of the rules used or make corrections if an error is determined to exist. TEIRESIAS, however, makes no effort to develop test cases automatically, or facilitate for Oracle review. It assumes that the developer will serve as the Oracle - a sensible approach for debugging, but a questionable practice for validation.

The development of TEIRESIAS was followed by EMYCIN [46], which fixes spelling errors, checks that rules are semantically and syntactically correct, and points out interactions among rules that could lead to errors. EMYCIN uses a trace of the system's reasoning process, an interactive mechanism for reviewing and correcting the system's conclusions (like TEIRESIAS), and a facility that compares the system's results with stored correct results for the test cases.

There are also a number of dynamic analysis tools that carry out (or assist) knowledge base refinement. Among these are SEEK [47] and SEEK2 [38, 48]. We categorize these as validation systems, since rule refinements are carried out in order to enable a modified knowledge-based system to produce improved performance, i.e., closer to that of the Oracle. SEEK is an interactive rule refinement system that uses stored cases to provide guidance for rule modifications. It makes suggestions of possible rule generalizations and specializations. As each test case is run, SEEK comparesan Oracle's conclusion for that case with the system's conclusion, creating a performance summary. Once the performance summary is complete the developer can refine the rules, weakening or strengthening rules with the guidance of heuristics built into SEEK. SEEK2 improves on SEEK by automating many of the knowledge base refinement tasks. For example, SEEK2 automatically decides for which conclusions rules should be evaluated. It also decides which rule refinements to try, and which of those to keep in the final knowledge base, as indicated by improvements when the refined knowledge base is used to evaluate the test cases.

Path Hunter and Path Tracer [49, 50] are based on the idea that functional validation may show that the system performs well on the test cases, but there may still be problems in portions of the rule base that were never exercised during testing. The goal of Path Hunter/Path Tracer is the selection of a set of test cases that exercise the structural components of the rule-base as exhaustively as possible. This involves firing all rules, and also firing every 'causal sequence' of rules. The model used to identify all possible dynamic causal rule-firing sequences is the *rule execution path.*

A weakness of many validation approaches is that, because they focus on a set of test cases with known results, they only assess the system's behavior for those known cases. However, this can lead to misleading prediction of the system's behavior in actual use, and may not identify errors in the system because of lack of coverage by the test set. The program Testing with Rule-Base Coverage (TRUBAC) [51-53] is a tool for both verification and validation, based on the idea that functional testing alone is inadequate for demonstrating equivalent output, and that structural testing of an intelligent system (in this case, a rule-based system) must also be used. An extension of TRUBAC [54] presents an approach for predicting a system's behavior in actual use. It accomplishes this by combining the coverage information provided by TRUBAC with information about the performance of the system on test data, meta-knowledge about the kinds of cases the system should handle and how realistic they are, and information about how representative of the intended population the test set is. TRUBAC facilitates both verification and validation of the KBS, and it improves the predictions we can make about future performance of the system. It is able to accomplish this because TRUBAC incorporates structural analysis into the validation process, whereas the majority of formal expert systems evaluation methods focus on verification only, or on a strictly functional approach to validation.

The concept of combining verification, validation and refinement in the same tool became popular in the early 1990's. The Validation, Verification and Refinement (VVR) system [55] performs all three functions. VVR generates a set of test cases and assists the developer in refining rules in light of the test case results. It makes suggestions on how to refine rules found to be guilty.

Lastly, Knauf et al. [56] propose a complete methodology for the validation of rule-based intelligent systems. This methodology is composed of the following steps:

1. Test case generation – Like TRUBAC, their methodology attempts to generate a minimal set of test cases that provides sufficient coverage to meet the pre-defined validation criteria. It uses a combination of structural testing with combinations of inputs to generate the original test case set (called the Quasi-exhaustive set of test cases, or QuEST). Then it reasons about the input values as well as several validation criteria to cull the test case set to the smallest size possible for the requirements. This last set is called the Reasonable set of test cases (ReST)

2. Test case experimentation – The intelligent system being validated is interrogated via the set of test cases, and its response is recorded. The same test cases are given to a panel of experts (size and makeup not addressed) who likewise provide a set of responses to the test cases.

3. Test case evaluation – This phase involves a Turing-Test-like process through which the test case responses from the experts as well as the system are evaluated by the same panel of experts interrogated previously. The methodology provides several formulae that can be used to identify the erroneous rule. The formulae take into account a judgment of the competency of each expert based on his/her responses.

4. Validity Assessment – This step of the process computes an assessment of the final overall validity of the system as presented. This validity is de-composed into output validity, rule validity, and test case validity. The rule validity is helpful in the next step.

5. System refinement – This step aims to improve the system, and makes use of the rule and test case validity to suggest how to improve the rule(s) found erroneous.

With the potential of this resource in mind, the current literature about the verification and validation of KBSs was surveyed to assess the state of the art and to determine its applicability to HBR validation. Several survey articles [57-64] and books [65-67] contain overviews of this important technology area.

CONCLUSIONS

The immaturity of HBR validation only widens the possibilities for its advancement. Many people have shown committed interests in HBR development and use. These interests, particularly from the user community, will drive the research, development and application necessary to advance HBR validation technology. Even funding agencies have begun to show interest in supporting HBR validation activities. All of the conditions appear right for an explosive growth in this important aspect of simulation validation.

REFERENCES

[1] A.J. Lang, M.R. Hulthen & G.A. Wadel, "Fighter Training Requirements Change With The Weather," Paper # 9th-CGF-042, Proc. 9th Conf. on Computer Generated Forces & Behavior Representation, Orlando, FL, 16-18 May 2000, np.

[2] M. Lorenzo, W. Riggs, P.D. Rizik, A.D. Christiansen, D. Florek & M.D. Petty, "Next Generation CGF Requirements to Support Sensor Development," Paper # 9th-CGF-056, Proc. 9th Conf. on Computer Generated Forces & Behavior Representation, Orlando, FL, 16-18 May 2000, np.

[3] E. Rolek, T. Hughes & E. Martin, "A Strategy for Defining Human Representation Requirements," Paper # 9th-CGF-068, Proc. 9th Conf. on Computer Generated Forces & Behavior Representation, Orlando, FL, 16-18 May 2000, np.

[4] D.J. Allsopp, "Title: Capturing the UK MOD CGF Requirement," Paper # 02-CGF-014, Proc. 11th Conf. on Computer Generated Forces & Behavior Representation, Orlando, FL, 7-9 May 2002, np.

[5]   T. Hughes, "Human Behavioral Representation Requirements for Integrated Air Defense System," Paper # 02-CGF-077, <u>Proc. 11th Conf. on Computer Generated Forces & Behavior Representation</u>, Orlando, FL, 7-9 May 2002, np.

[6]   B. Chandrasekaran & J.R. Josephson, "Cognitive Modeling For Simulation Goals: A Research Strategy for Computer-Generated Forces," Paper # 8th-CGF-100, <u>Proc. 8th Conf. on Computer Generated Forces and Behavior Representation</u>, Orlando, FL, 11-13 May 1999, np.

[7]   Veit, C.T. & Callero, M., <u>Criteria for Validating Human Judgments and Developing Behavioral Representation Models</u>, The Rand Corporation, Santa Monica, CA, 1993.

[8]   S.Y. Harmon, "A Taxonomy of Human Behavior Representation Requirements," Paper # 11TH-CGF-024, Proc. 11th Computer Generated Forces and Behavior Representation Conference, Orlando, FL, May 2002, np.

[9]   H.S. Delugach & D.J. Skipper, "Knowledge Techniques for Advanced Conceptual Modeling," Paper # 9th-CGF-004, <u>Proc. 9th Conf. on Computer Generated Forces & Behavior Representation</u>, Orlando, FL, 16-18 May 2000, np.

[10]   R.D. Dubois, Jr. & R.J. Might, "Conceptual Modeling of Human Behavior (CMHB)," Paper # 9th-CGF-087, <u>Proc. 9th Conf. on Computer Generated Forces & Behavior Representation</u>, Orlando, FL, 16-18 May 2000, np.

[11]   G.P. Lannon, H.A. Klein & D.H. Timian, "Integrating Cultural Factors into Threat Conceptual Models," Paper # 10th-CGF-013, <u>Proc. 10th Conf. on Computer Generated Forces & Behavior Representation</u>, Norfolk, VA, 15-17 May 2001, np.

[12]   G. Rigg, R. Morley & R. Hepplewhite, "Themis: The Objective Assessment of CGF Performance," Paper # 9th-CGF-020, <u>Proc. 9th Conf. on Computer Generated Forces & Behavior Representation</u>, Orlando, FL, 16-18 May 2000, np.

[13]   A.J. Courtemanche & S.A. Gugel, "Automated Testing for CGF Systems," Paper # 10th-CGF-021, <u>Proc. 10th Conf. on Computer Generated Forces & Behavior Representation</u>, Norfolk, VA, 15-17 May 2001, np.

[14]   S. Wallace & John Laird, "Toward Automatic Agent Validation," Paper # 02-CGF-053, <u>Proc. 11th Conf. on Computer Generated Forces & Behavior Representation</u>, Orlando, FL, 7-9 May 2002, np.

[15]   M. Oakes & M. Kitchen, "Computer Generated Force (CGF) Interaction Validation and Assessment," Paper # 02-CGF-027, <u>Proc. 11th Conf. on Computer Generated Forces & Behavior Representation</u>, Orlando, FL, 7-9 May 2002, np.

[16]   Y.J. Tenney & S.L. Spector, "Comparisons of HBR Models with Human-in-the-Loop Performance in a Simplified Air Traffic Control Simulation with and without HLA Protocols: Task Simulation, Human Data and Results," Paper # 10th-CGF-074, <u>Proc. 10th Conf. on Computer Generated Forces & Behavior Representation</u>, Norfolk, VA, 15-17 May 2001, np.

[17]   D.G. Hoagland, E.A. Martin, M. Anesgart, B.E. Brett, N. LaVine & S. Archer, "Combat Automation Requirements Testbed (CART) Case Study One: Results

from Modeling Human Behavior Within High Fidelity Constructive Simulations," Paper # 10th-CGF-077, Proc. 10th Conf. on Computer Generated Forces & Behavior Representation, Norfolk, VA, 15-17 May 2001, np.

[18]   M. Avraamides & F. Ritter, "Using Multidisciplinary Expert Evaluations to Test and Improve Cognitive Model Interfaces," Paper # 02-CGF-002, Proc. 11th Conf. on Computer Generated Forces & Behavior Representation, Orlando, FL, 7-9 May 2002, np.

[19]   F. Ritter, M. Avraamides & I. Councill, "Validating Changes to a Cognitive Architecture to More Accurately Model the Effects of Two Example Behavior Moderators," Paper # 02-CGF-100, Proc. 11th Conf. on Computer Generated Forces & Behavior Representation, Orlando, FL, 7-9 May 2002, np.

[20]   J. Miller, "Domain-Expert and End-User Participation in the Design and Testing of Synthetic Teammates," Paper # 02-CGF-110, Proc. 11th Conf. on Computer Generated Forces & Behavior Representation, Orlando, FL, 7-9 May 2002, np.

[21]   S.Y. Harmon, "Validation of Human Behavior Representations," Paper # 99S-SIW-048, Proc. Spring 1999 Simulation Interoperability Workshop, Orlando, FL, 14-19 March 1999, np.

[22]   R.W. Pew & A.S. Mavor, eds., Modeling Human and Organizational Behavior, National Academy Press, Washington, DC, 1998.

[23]   Simulation Interoperability Standards Organization, Glossary of Fidelity-Related Terms, Simulation Interoperability Standards Organization, Orlando, FL, 1999.

[24]   D.C. Gross et al., "Report from the Fidelity Implementation Study Group," Paper # 99S-SIW-167, Proc. Spring 1999 Simulation Interoperability Workshop, Orlando, FL, 14-19 March 1999, np.

[25]   S.Y. Harmon & D.C. Gross, Appendix B of Fidelity ISG Report, Simulation Interoperability Standards Organization, Orlando, FL, 1999.

[26]   D.C. Gross et al., "Why Fidelity," Paper # 168, Proc. Spring 1999 Simulation Interoperability Workshop, Orlando, FL, 14-19 March 1999, np.

[27]   D.C. Gross, Z.C. Roza & S.Y. Harmon, "Report Out of the Fidelity Experimentation Group," Paper # 00S-SIW-151, Proc. Spring 2000 Simulation Interoperability Workshop, Orlando, FL, 26-31 March 2000, np.

[28]   A.E. Henninger, A.J. Gonzalez, M. Georgipoulos & R.F. DeMara, "The Limitations of Static Performance Metrics for Dynamic Tasks Learned Through Observation," Paper # 10th-CGF-047, Proc. 10th Conf. on Computer Generated Forces & Behavior Representation, Norfolk, VA, 15-17 May 2001, np.

[29]   D.R. Law, "Characterizing Fairness of SAF Interoperations," Paper # 8th-CGF-074, Proc. 8th Conf. on Computer Generated Forces & Behavior Representation, Orlando, FL, 11-13 May 1999, np.

[30]   S.Y. Harmon & S.M. Youngblood, "Fidelity and Human Behavior Representations," Paper # 01S-SIW-113, Proc. Spring 2001 Simulation Interoperability Workshop, Orlando, FL, np.

[31] A.J. Gonzalez & V. Barr, "Validation and Verification of Intelligent Systems – What are They and how They are Different?," <u>Journal of Experimental & Theoretical Aritifial Intelligence</u>, 12(4), 2000, pp407-420.

[32] W. Suwa, A.C. Scott & E.H. Shortliffe, "An Approach to Verifying Completeness and Consistency in a Rule-Based Expert System," <u>AI Magazine</u>, 3(4), Winter 1982, pp16-21.

[33] A.D. Preece, "Verification of Rule-Based Expert Systems in Wide Domains," <u>Proc. 6<u>th</u> Conf. on Research and Development in Expert Systems</u>, N. Shadbolt, ed., London, England, Cambridge University Press, September 1989, pp66-77.

[34] T.A. Nguyen, W.A. Perkins, T.J. Laffey & D. Pecora, "Checking an Expert System Knowledge Base for Consistency and Completeness," <u>Proc. 9<u>th</u> Int. Joint Conf. on Artificial Intelligence (IJCAI-85)</u>, Los Angeles, CA, August 1985, pp375-378. [2, 5, 16, 20, 26, 29]

[35] T.A. Nguyen, W.A. Perkins, T.J. Laffey & D. Pecora, "Knowledge Base Verification," <u>AI Magazine</u>, 8(2), Summer 1987, pp69-75.

[36] B.J. Cragun & H.J. Steudel, "A Decision-Table-Based Processor for Checking Completeness and Consistency in Rule-Based Expert Systems," <u>International Journal of Man-Machine Studies</u>, 26, 1987, pp633-648.

[37] A. Ginsberg, "A New Approach to Checking Knowledge Bases for Inconsistency and Redundancy," <u>Proc. 3<u>rd</u> Annual Expert Systems in Government Conf.</u>, Washington, DC, 1987, pp102-111.

[38] A. Ginsberg, "Knowlegde-Base Reduction: A New Approach to Checking Knowledge Bases for Inconsistency and Redundancy," <u>Proc. 7<u>th</u> National Conf. on Artificial Intelligence (AAAI-88)</u>, August 1988, pp585-589. [5, 6, 19, 20, 26, 29]

[39] M. Dahl & K. Williamson, "Experiences of Using Verification Tools for Maintenance of Rule-Based Systems," <u>Working Notes, AAAI-93 Workshop on Validation and Verification of Knowledge-Based Systems</u>, Washington, D.C.., 1993, pp114-119.

[40] A.D. Preece & R. Shinghal, "Analysis of Verification Methods for Expert Systems," <u>Working Notes, AAAI-92 Workshop on Validation and Verification of Knowledge-Based Systems</u>, San Jose, CA., 1992, np.

[41] P.D. Grogono, A.D. Preece, R. Shinghal, R., et al., "A Review of Expert Systems Evaluation Techniques," <u>Workshop Notes, AAAI-93 Workshop on Validation and Verification of Knowledge-Based Systems</u>, AAAI Report WS-93-05, A.D. Preece, ed., Washington, DC, July/August 1993, np.

[42] D. Zhang & D. Nguyen, "A Technique for Knowledge Base Verification, Proc. IEEE International Workshop on Tools for Artificial Intelligence, 1989, pp399-406.

[43] G. Valiente, "Verification of Knowledge Base Redundancy and Subsumption Using Graph Transformation," <u>International Journal of Expert Systems</u>, 6(3),

1993, pp41-55.

[44]   D. Hutter, G. Langenstein, G. Rock, J. Siekmann, W. Stephan & R. Vogt, "Formal Software Development in the Verification Support Environment (VSE)" <u>Journal of Experimental & Theoretical Aritifial Intelligence</u>, 12(4), 2000, pp383-406.

[45]   R. Davis, "Interactive Transfer of Expertise," <u>Rule-Based Expert Systems</u>, B.G. Buchanan & E.H. Shortliffe, eds, Addison Wesley, Reading MA, 1985, pp171-205.

[46]   W. van Melle, E.H. Shortliffe & B.G. Buchanan, "EMYCIN: A Knowledge Engineer's Tool for Constructing Rule-Based Expert Systems," <u>Rule-Based Expert Systems</u>, B.G. Buchanan & E.H. Shortliffe, eds, Addison Wesley, Reading MA, 1985, pp302-313.

[47]   P. Politakis & S.M. Weiss, "Using Empirical Analysis to Refine Expert System Knowledge Bases," <u>Artificial Intelligence</u>, 22(1), January 1984, pp23-48.

[48]   A. Ginsberg, S.Weiss & P. Politakis, "SEEK2: A Generalized Approach to Automatic Knowledge Base Refinement," <u>Proc. 9th Int. Joint Conf. on Artificial Intelligence (IJCAI-85)</u>, Los Angeles, CA, August 1985, pp367-374.

[49]   C. Grossner, A. Preece, P.G. Chander, T. Radhakrishnan & C.Y. Suen, "Exploring the Structure of Rule-Based Systems," <u>Proc. 11th National Conf. on Artificial Intelligence (AAAI-93)</u>, Washington, DC, July/August 1993, pp704-709.

[50]   A.D. Preece, C. Grossner, P.G. Chander & T. Radhakrishnan, "Structural Validation of Expert Systems: Experience Using a Formal Model," <u>Workshop Notes, AAAI-93 Workshop on Validation and Verification of Knowledge-Based Systems</u>, AAAI Report WS-93-05, A.D. Preece, ed., Washington, DC, July/August 1993, pp19-26.

[51]   V. Barr, "TRUBAC: A Tool for Testing Expert Systems with Rule-Base Coverage Measures," <u>Proc, 13th Annual Pacific Northwest Software Quality Conference</u>, Portland, OR., 1995, np.

[52]   V. Barr, <u>Applications of Rule-Base Coverage Measures to Expert System Evaluation</u>, Rutgers University Technical Report DCS-TR-340, 1996.

[53]   V. Barr, "Applications of Rule-Base Coverage Measures to Expert System Evaluation, <u>Journal of Knowledge Based Systems</u>, 12, 1999, pp27-35.

[54]   V. Barr, "Applying Reliability Engineering to Expert Systems," <u>Proc. FLAIRS-99</u>, Orlando, FL., 1999, pp494 - 498.

[55]   N.P. Zlatareva, "A Framework for Knowledge-Based System Verification, Validation and Refinement: The VVR System," <u>Proc. of the 5th Florida Artificial Intelligence Symp.</u>, Ft. Lauderdale, FL, April 1992, pp10-14.

[56]   R. Knauf, <u>Validating Rule-Based Systems. A Complete Methodology</u>, ISBN 3-8265-8293-4, Shaker, Aachen, Germany, 2000.

[57] Lopez, B., Meseguer, P. & Plaza, E., "Knowledge-Based Systems Validation: A State of the Art," <u>AI Communications</u>, 3(2), 1990, pp58-72.

[58] Grogono, P., Batarekh, A., Preece, A., Shinghal, R. & Suen, C., "Expert System Evaluation Techniques: A Selected Bibliography," <u>Expert Systems</u>, 9(4), 1992, pp227-239.

[59] Nazareth, D.L. & Kennedy, M.H., "Knowledge-Based System Verification, Validation and Testing: The Evolution of a Discipline," <u>Int. Jour. of Expert Systems</u>, 6(2), 1993, pp143-162.

[60] Preece, A.D. & Suen, C.Y., eds., "Special Issue on Verification and Validation," <u>Int. Jour. of Expert Systems - Research and Applications</u>, 6(2/3), 1993.

[61] Mengshoel, O.J. & Delab, S., "Knowledge Validation: Principles and Practice," <u>IEEE Expert</u>, 8(3), June 1993, pp62-68.

[62] O'Keefe, R.M. & O'Leary, D.E., "Expert System Verification and Validation: A Survey and Tutorial," <u>Artificial Intelligence Review</u>, 7, 1993, pp3-42.

[63] Zlatareva, N. & Preece, A., "State of the Art in Automated Validation of Knowledge-Based Systems," <u>Expert Systems with Applications</u>, 7(2), 1994, pp151-167.

[64] Preece, A.D., "Validation of Knowledge-Based Systems: Current Trends and Issues," <u>Knowledge Engineering Review</u>, 10(1), 1995, pp69-71.

[65] Ayel, M.C. & Laurent, J.-P., eds., <u>Validation, Verification and Test of Knowledge-Based Systems</u>, John Wiley & Sons, New York, NY, September 1991.

[66] Bahill, A.T., ed., <u>Verifying and Validating Personal Computer-Based Expert Systems</u>, Prentice Hall, Englewood Cliffs, NJ, 1991.

[67] Gupta, U.G., ed., <u>Validating and Verifying Knowledge-Based Systems</u>, IEEE Computer Society Press, Los Alamitos, CA, 1992.

AUTHOR BIOGRAPHIES

Scott Y. Harmon is president and founder of Zetetix, a small business that performs research into modeling and validating models of complex information systems. Mr. Harmon has been involved with modeling and simulation for over twenty five years, working in industry, both large and small, and in the Government. Throughout this time, he has primarily concentrated on understanding the nature of human behavior. Recently, he derived the SISO fidelity conceptual model, the first formal treatment of model and simulation fidelity, managed the construction of the SISO Glossary of Fidelity Related Terms, developed concrete approaches for validating human behavior representations, and authored the sections in the DMSO VV&A Recommended Practices Guide on validation and HBR validation. He is currently developing a detailed model of human behavior based upon neurophysiology, deriving rigorous techniques for objectively validating models and simulations, and elaborating his theory of information physics. Mr. Harmon is a member of AAAS, AAAI, AFCEA, APS, ASME, IEEE, IEEE

Computer Society, SISO, and Sigma Xi. He is also an active member of the distributed simulation community and has authored or contributed to over 75 technical publications on modeling and simulation, robotics, information system architectures and system engineering.

C.W. David Hoffman is a retired Army officer and operations research analyst. He has 19 years experience in the development and validation of military human behavior representations.   Mr. Hoffman is currently a member of the Simulation Support Directorate for the Army's Training and Doctrine Command (TRADOC) Analysis Center at White Sands Missile Range (TRAC-WSMR), New Mexico. He has more than 19 years of active duty and civilian experience developing, evaluating, and validating human behavior representations in military analytical and training M&S. In addition, he is consulted with STRICOM and the National Ground Intelligence Center (NGIC) in the development of composable behavior technologies. Mr. Hoffman is currently responsible for the development of the OneSAF behavior knowledge engineering effort in support of the OneSAF composable behavior functionality and responsible for management of the OneSAF program V&V activities.

Mr. Hoffman holds a BA in English Literature from the Virginia Military Institute and an MS in Operations Research from the Florida Institute of Technology.   His Army experience includes Field Artillery, nuclear weapons, and operation research; and he is a graduate of the Command and General Staff College and other Army and NATO courses. He is a member of MORS and is the current chairman for the Army Model Improvement Program Standards Committee for Computer Generated Forces (CGF).

Avelino J. Gonzalez received his Bachelor's and Master's degrees in Electrical Engineering from the University of Miami, in 1973 and 1974 respectively. He obtained his Ph.D. from the University of Pittsburgh in 1979 also in Electrical Engineering. He spent nearly 12 years with Westinghouse Electric Corp. in Pittsburgh, PA and Orlando, FL.   His work during his last seven   with Westinghouse was working in computer applications to electrical power engineering. One of his most significant efforts was the development of the Westinghouse generator diagnostic expert system GenAID for which he received the Westinghouse Award of Excellence in Engineering. Before his departure from Westinghouse in 1986, he directed the GenAID development engineering group, as well as the Westinghouse Diagnostics Center in Orlando, Fl. He has authored a number of papers as well as one textbook in the above topics. In 1986, Dr. Gonzalez joined the Computer Engineering Department (which later became the Electrical and Computer Engineering Dept., and is now the part of the School of Electrical Engineering and Computer Science) at the University of Central Florida (UCF) in Orlando. There he has focused his research and teaching in Artificial Intelligence and knowledge-based systems. He has served as PI in several large Federal research contracts and grants.

He is presently involved in two specific research areas related to artificial intelligence: 1) Human behavior models for computer generated forces.   This includes efficiently representing their behaviors, acquiring the behaviors in an efficient and effective manner through observation, and validating the behaviors; 2) Diagnosis and control of electrical the power distribution systems. This research investigates the feasibility of using model-based reasoning to perform real-time diagnosis and control. Dr. Gonzalez is a registered Professional Engineer in the State of Florida, and was the founding president of the

Florida Artificial Intelligence Research Society. He presently serves as the Treasurer of that organization.

Rainer Knauf was born in 1963 in Erfurt (Germany). He received his Diploma degree in Electrical Engineering in 1987 at the Ilmenau Institute of Technology. From 1987-90 he worked as an Assistant for Research and Teaching in the field of Artificial Intelligence at the Ilmenau Institute of Technology and received a Doctoral degree (Ph.D.) in Computer Science in 1990.

Since 1990 he has worked in several positions at the Ilmenau Technical University. He attained an Habilitation degree in 2000 and holds an Associate Professorship (Privatdozent) at the Chair of Artificial Intelligence at the Faculty of Computer Science and Automation. His research is focused on the evaluation of intelligent systems.

Valerie Barr is an associate professor in the Department of Computer Science at Hofstra University. Her research focuses on verification and validation of intelligent systems, primarily rule-based expert systems and natural language processing systems. Prof. Barr received a PhD in computer science from Rutgers University. She is a member of the IEEE Computer Society, ACM, AAAI, ACL, and Computer Professionals for Social Responsibility. She is also on the Advisory Board of the Institute for Women and Technology.