Ritter, F. E., Bittner, J. L., Kase, S. E., Evertsz, R., Pedrotti, M., & Busetta, P. (2012). CoJACK: A high-level cognitive architecture with demonstrations of moderators, variability, and implications for situation awareness. *Biologically Inspired Cognitive Architectures, 1*(1), 2-13.

# CoJACK: A High-Level Cognitive Architecture with Demonstrations of Moderators, Variability, and Implications for Situation Awareness

Frank E. Ritter[a*] (frank.ritter@psu.edu), Jennifer L. Bittner[a] (bittnerj@indiana.edu),
Sue E. Kase[b] (sue.e.kase.civ@mail.mil), Rick Evertsz[c] (rick.evertsz@aosgrp.com),
Matteo Pedrotti[c] (matteo.pedrotti@aosgrp.com), and
Paolo Busetta[c] (paolo.busetta@aosgrp.com)

[a]College of IST, Penn State, University Park, PA 16802, USA
[b]Army Research Laboratory, RDRL-CII-C, Aberdeen Proving Ground, MD 21005, USA
[c]AOS Group, Wellington House, East Road, Cambridge, CB1 1BH, UK

Draft of 29 April 2012. Word count: 8,449 inclusive of references, title, etc.
Corresponding author: Frank Ritter, +1 (814) 865-4453
316g IST Building, Penn State, University Park, PA 16802

## Abstract

We report a high-level architecture, CoJACK, that provides insights on behavior variability, situation awareness, and behavioral moderators. CoJACK combines Beliefs/Desires/Intentions (BDI) agents' high-level knowledge representation and usability with several aspects of low level cognitive architectures, including processing time predictions, errors, and traceability. CoJACK explores new areas for cognitive architectures, such as variability arising from moderators. It also allows aspects of situation awareness (SA) in a cognitive architecture to be explored. Its behavior and the effects of moderators on behavior are demonstrated in a simple adversarial environment. It provides lessons for other architectures including how to define, measure, and control variability due to individual and temporal aspects of cognition; the importance of SA and knowledge representations necessary to support complex SA; the potential for parameter sweeps and paths as measures of variability; and some of the complexities that will arise when aspects of moderators and SA are added to cognitive architectures.

## *Keywords*

Cognitive architecture; BDI; situation awareness; behavior moderators; usability;

# 1. Introduction

Cognitive models have been applied in a variety of places, including as agents in synthetic environments (SEs) (e.g., Best, Lebiere, & Scarpinatto, 2002; Jones et al., 1999; Pew & Mavor, 1998) and for predicting behavior in system design (e.g., Booher & Minninger, 2003; Pew & Mavor, 2007). Cognitive architectures have been used to create these models. However, there are many limitations on cognitive models in these applications, including a lack of high-level representations and the difficulties in modeling variability in performance.

This paper details a novel cognitive architecture, CoJACK, that can be used to address variability, situation awareness, and moderated behavior, while maintaining usability. We begin by defining and discussing these four major themes.

## *1.1 Variability*

We define variability as changes in human performance in three ways: across individuals, by situation, and across time. Variability has been used widely in the area of psychology, specifically in the area of individual differences (e.g., Jonassen & Grabowski, 1993; Simon & Simon, 1978). There appear to be many factors that can cause variability in human behavior including differences across individuals, differences across time on a task within an individual, and the effects of context on a task, such as time pressure, heat, sleep patterns, and food, drink, and psychoactive substances such as caffeine. These differences can be because performance is influenced by total workload or because other aspects of the environment (such as heat or social context) influence performance.

Although these sources of variability on performance have been studied in many ways in psychology, including sleep models (e.g., Belyavin & Spencer, 2004), the effects of these types of variability have been less often addressed in cognitive models (although, see Lovett, Daily, & Reder, 2000, Gunzelmann, Gross, Gluck, & Dinges, 2009, and Morgan, Morgan, & Ritter, 2010, for counter examples).

Variability is important in adversarial problem solving. Variability makes opponents more challenging because it makes interpretation more difficult. It also requires that multiple examples have to be taken to see the distribution of behavior. If an opponent is too predictable two problems can arise. One problem, limited learning, is that with time an agent will discover every move of the predictable opponent and learn only how to address those fixed responses. The other problem is fixed learning, that the agent learns to provide a fixed response because the opponent is not variable in their response to your behavior. These problems can lead to poorer learning, a false sense of confidence, and less development of alternatives in the learner's behavior against opponents. However, it can be difficult for analysts for the same reasons.

We have found one particularly interesting example of individual differences studied in SEs with a comparison to data. Poncelin de Raucourt (1997) studied individual differences in the World War II battle of Medenine using the Ironsides simulation (Harrison, Winters, & Anthistle, 1999). He also had actual performance based on diary reports of the battle. He found that Ironsides had a relatively flat distribution of tanks destroyed across the Allied crews, that is, each anti-tank crew on the Allied team had destroyed roughly an equal number of opponent tanks. The diary studies of the battle reported that the number of tanks destroyed per team was

significantly and reliably different from a flat distribution. His work suggests that existing simulations need to have variability (however it arises) across agents.

## 1.2 Situation awareness

Situation awareness (SA) is a macrocognitive construct representing how well an operator's representation is matched to the world, including the actors and activities, agents' goals, and future behavior (Endsley, 1995). SA and its maintenance depend on cognition, particularly the processes and mechanisms in an observer and orient phases of the Observe-Orient-Decide-Act (OODA) loop (Endsley & Jones, 1997). Factors that influence cognition will also influence SA thus influencing behavior via a secondary route—differences in SA can lead to further variability in human performance. For example, reductions in working memory capacity will not only influence performance, they will also influence the ability to perform activities related to SA.

Aspects of SA can be seen in agent models in SEs. The simple aspects—what are the objects in the world—is, of course, included. What the objects are doing is sometimes included in a model. What they are going to do is difficult, and few agents do this. It has been done in Soar (Laird, 2001), including TacAir-Soar (Nielsen, Crossman, & Jones, 2007), but doing so is an exception.

## 1.3 Moderators

A range of external and internal factors, which can be labeled as behavioral moderators, can also affect cognition, including the higher level aspects of SA and variability. These moderators include the effects of various types of stress (e.g., time pressure, task importance), the external world impinging on the user (e.g., heat, vibration), and substances such as caffeine. The moderators can be grouped into internal and external types, depending on whether their genesis is from the operator or from the world. Moderators have been studied physiologically and psychologically but are not often included in computational models due to limitations in architecture and as a working assumption about what to model first (e.g., problem solving) and what to defer (e.g., emotions and moderators). There have been cases where moderators have been included in architectures (Bach, 2008; Bates, 1994; e.g., Gratch & Marsella, 2004; Hudlicka, 2002; Ritter, Reifers, Klein, & Schoelles, 2007). These are relatively few, however, and are not yet widely used.

## 1.4 Usability

A problem related to making a more complete architecture is that of keeping the resulting system usable. Each of these constructs adds to the complexity of the model and its behavior. At the same time, a few authors have noted that models are already not as usable as they could be or need to be. Pew and Mavor (1998) in their review of modeling human behavior include a subsection that addresses usability of models. Pew and Mavor (2007), in their report on using user models as a shared representation to reduce risk in system design, explicitly call for making models easier to use. Shakir (2002) in an internal government report noted that it was better to build an interface to the JACK® architecture (Busetta, Rönnquist, Hodgson, & Lucas, 1999) and extend it rather than use an existing architecture because the existing alternative was too unusable. Usability as an important but fixable limitation of the application of cognitive architectures in SEs (Ritter, 2009; Ritter et al., 2003). Thus, because the usability of models is a current and pressing concern in model creation, future work needs to keep usability in mind.

One way to address usability in modeling is to use a high-level behavior representation language (Haynes, Cohen, & Ritter, 2009; Ritter et al., 2006). High-level representation languages provide several advantages, including clarity of representation and ease of creating models. When referring to high-level representations we mean the ability to describe behavior at a composite, structured level. Many cognitive architectures represent knowledge at the level of if-then rules. This has been a useful representation, but it has also limited the ability to create models.

### *1.5 Preview of the paper*

To explore these topics, how they interact, and how they can be put to work in synthetic environments, we report a novel high-level cognitive architecture called CoJACK (Cognitive JACK). CoJACK directly supports representations of variability, SA, and moderators. It combines the high-level knowledge representation used in Beliefs/Desires/Intentions (BDI) agents (Rao & Georgeff, 1995) and their associated high levels of usability with several aspects of low level cognitive architectures, including traceability, processing time predictions, working memory effects, and errors in behavior.

CoJACK includes explicit, traceable representations of the basic aspects of SA, including external actors and their actions. Its parameter sets, because it includes parameters that represent concepts such as noise in information processing mechanisms, can lead to variability between similar agents. The parameters can also be used to create moderators. For example, the parameters can also be modified to create an overlay to the architecture to describe a particular moderator, such as task appraisal stress (Ritter, Reifers, Klein, & Schoelles, 2007). These modifications to JACK implement a generic technique that provides a means of representing the influence of internal and external moderators on behaviour.

## 2. Materials and Methods

### *2.1 JACK*

To describe CoJACK, we first briefly describe JACK (Busetta, Rönnquist, Hodgson, & Lucas, 1999). JACK is based on the BDI approach, a paradigm that explicates rational decision-making about action. The BDI paradigm was developed in response to a perceived problem with existing AI approaches to planning. Agents are typically situated in a dynamic environment and must constantly review their goals and activities. They should also be aware of the resource-bounded nature of their reasoning. Earlier AI approaches to planning only addressed the offline planning problem—how to achieve the goal given infinite time and resources. They failed to address the temporal pressures that apply when trying to achieve goals within a fluctuating environment that presents a multitude of interacting, conflicting, and changing opportunities.

JACK represents and executes tactics in a manner that maps well to subject matter experts' (SMEs) introspections about their own reasoning. The knowledge representation includes a front-end that allows analysts to specify tactics graphically at a high-level (shown in Figure 1). The graphical representation is useful for visualizing the logical structure of tactics and discussing them with SMEs. This representation language provides higher level constructs and thus direct support for usability.

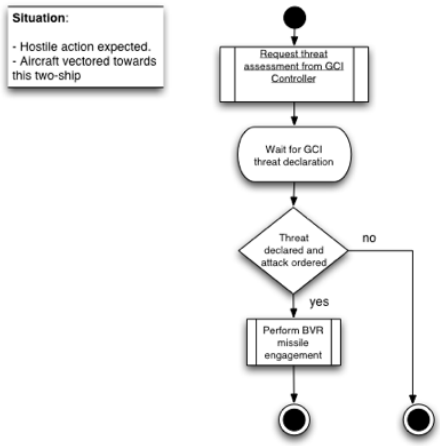



Figure 1. A simple plan represented in JACK.

## *2.2 The CoJACK architecture*

JACK was modified to make a cognitive architecture. It implements variability, SA, and moderators, while maintaining usability through a high-level representation of behavior.

### 2.2.1 CoJACK extends JACK with variability, SA, and Moderators

One way to view cognitive architectures is to classify them in terms of the constraints they impose upon cognitive models. Howes and Young (1997) suggest that cognitive architectures consist of soft and hard constraints. Soft constraints can be overridden whereas hard constraints enforce particular properties on the models, with no way to circumvent them within the architecture. From this perspective, CoJACK is intended to provide hard constraints (because they are provided automatically to a JACK model), but because of the Java implementation and the ability to extend JACK and CoJACK, the constraints are somewhat soft. This approach offers considerable flexibility because the degree of cognitive modelling sophistication and consequent behavioral variability can be altered to suit the goals of the simulation study.

We maintained JACK's usability while modifying the underlying architecture's performance. Thus, users write a JACK agent, and turn on the cognitive architecture features in CoJACK. CoJACK augments JACK with a set of cognitive architectural constraints and parameters, as well as a moderator layer. When enabled, the agent's behavior is modulated by the cognitive architecture components embedded in JACK. The major activities in the JACK architecture, such as adding a belief or instantiating a plan have a time cost associated with them. Based upon cognitive parameters' values, the architectural constraints add latency (either computed or in real time) to the current intention's reasoning steps and to memory access.

CoJACK also affects the choice of beliefs retrieved in response to a memory access attempt; this includes effects such as failure to retrieve a matching belief, retrieval of a belief that only partially matches, and retrieval of an alternative matching belief (i.e., not the one that JACK would have chosen first). A similar mechanism affects the selection of the next BDI intention to execute. Thus, the agent can choose an unanticipated intention or even fail to retrieve one of its current intentions, and it has less resources creating a more limited focus of attention.

The cognitive parameters can be moderated at runtime, leading to further variation in behavior. For example, a caffeine moderator can be added that decreases the time taken to perform reasoning steps, leading to shorter response times, which we demonstrate later.

### 2.2.2 Key CoJACK assumptions and features

CoJACK addresses the following five key aspects of cognition:

**Limited access to procedural and declarative memory**—Working Memory (WM) refers to the currently active subset of the human cognitive system. WM is the mechanism of human cognition that maintains information during processing. It is limited in capacity as evidenced by decreased performance in tasks that require many temporary items to be held in memory (e.g., Anderson, 2007). Although the term WM is usually restricted to declarative memory, it should also encompass the dynamic aspects of procedural memory. CoJACK incorporates WM access limits (termed errors of omission) through an activation-level mechanism similar to ACT-R's mechanisms (Anderson, 2007; Anderson et al., 2004).

In contrast to its support for declarative memory, ACT-R does not include a theory describing how recency and decay apply to dynamic procedural memory. This is a real issue for CoJACK because it represents procedures as plans rather than productions. A plan typically has a number of steps, and when a CoJACK model forms an intention from a plan, that intention is held in a dynamic memory buffer that enables the agent to step through the plan. CoJACK can maintain a number of such intentions (determined by sub-symbolic activation-level equations), and switches its focus to the most active intention. This allows it to work on a number of tasks concurrently, much as a short-order cook manages multiple dishes (Kirlik, 2006).

**Error-prone retrieval from procedural and declarative memory**—In addition to failing to access memory elements, errors can occur when the *wrong* memory item is retrieved. CoJACK adopts a similar approach to ACT-R's partial-matching mechanism (Anderson & Lebiere, 1998), producing errors of commission.

**Cognition takes time**—One of the major limitations of most AI-based models is that they fail to represent the time taken to think and act. If, for example, the granularity of the simulation is 1 s, a model can, in theory, and often in practice, run through a series of decision-making steps in 1 s (simulated or real) that would take a human half an hour. CoJACK addresses this problem by computing the time of reasoning steps (based on its architecture and moderated parameter set).

**Limited focus of attention**—Resource limitation is a key property of human cognition. The allocation of this limited computational resource is generally referred to as attention. In CoJACK, the modelling of attention takes account of the following:

How an agent deliberately focuses its attention in a particular direction (e.g., focusing on an important goal while ignoring distracting environmental stimuli).

How factors like caffeine or fatigue moderate an agent's attention.

How the properties of memory affect attention. For example, WM elements have a limited life span. If a WM element is essential to the current task, the agent must counteract its natural decay through some form of rehearsal. This rehearsal process consumes attention resources.

**Cognition can be moderated**—Human behavior can be modified (moderated) by a range of factors, including temporal, environmental, physiological, and internal factors. Moderators can influence entity behavior directly—for example, caffeine increases the ability to focus, allowing performance on a vigilance task to remain virtually sustained at its original level instead of decreasing over the span of an hour (Boff & Lincoln, 1988, ch. 7, p. 804). Moderators can also influence lower-level mechanisms that give rise to the changed performance—for example, caffeine can provide a 5-10% faster reaction time on a simple reaction time task (G. P. Morgan, Ritter, Stine, & Klein, 2006), which will influence other aspects of behavior. By simulating the effects of moderators on underlying mechanisms, it is possible to predict behavior variation that will occur for a task that has not yet been studied closely. For example, if the effects of caffeine on the low-level aspects of cognition and the body are well understood, but the effects of caffeine on, say, radar operators had not been specifically studied, it should be possible to provide at least initial predictions of caffeine's effects on the behavior of radar operators based on knowing their cognitive mechanisms and the knowledge necessary to perform the task.

Figure 2 shows the relationship between JACK, CoJACK and the environment. JACK performs meta-level and tactical reasoning. Its performance is modulated by the cognitive architecture, which in turn is affected by the moderator layer. Percepts and other environmental data affect the appropriate moderators.

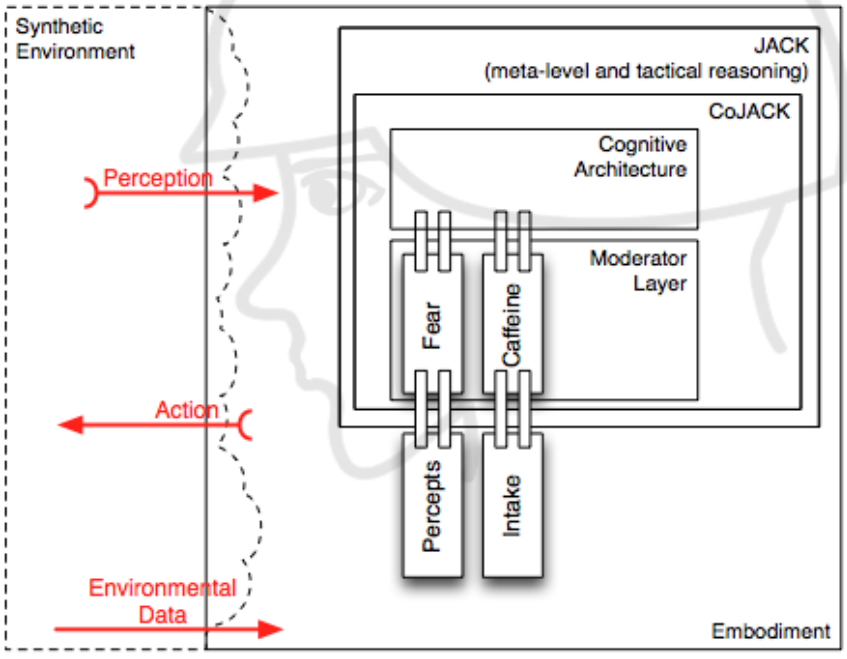



Figure 2. CoJACK's components and their relationship to JACK.

# 3. Theory and Calculations

## *3.1 Design and implementation of variability*

Models in CoJACK are largely built at a high level (specifying an agent's plans, events and beliefs handled). However, the final behavior of a CoJACK model depends on the sub-symbolic properties of the architecture, which are taken from ACT-R.

In CoJACK, beliefs and intentions are subject to sub-symbolic effects. To unify their treatment, they are collectively termed "chunks" in CoJACK. A belief is a declarative memory chunk; intentions are procedural memory chunks. From the point of view of working memory, a chunk is a single item[1].

Activation is a key aspect of the sub-symbolic properties of CoJACK. Chunks have an activation level that changes over time and when the chunk is used. This activation level is one of the main influences on the likelihood that a chunk will be retrieved, and how long retrieval will take. Given a set of competing chunks (i.e., ones that match retrieval specifications), CoJACK will select probabilistically the most activated chunk, subject to the chunk being above the retrieval threshold. Recall that CoJACK events govern the agent's computational flow (i.e., internal and external events produce intentions that then drive behavior). When an event is processed, it acts as an activation source for chunks. Event activation can be moderated to support goals with varying "salience" (e.g., "Stay alive" might have a higher activation than the goal "Watch TV"). This provides a straightforward method for the developer to focus the agent's attention on the events that are important to the agent.

CoJACK has over 30 cognitive parameters taken from ACT-R and adapted for a BDI architecture (Norling & Ritter, 2004a, 2004b). ACT-R's parameters have been experimentally validated across a wide range of tasks (see, for example, Anderson, 2007; Anderson et al., 2004; Anderson & Lebiere, 1998). We have found that these parameters are an excellent starting point for a novel architecture such as CoJACK. These parameters provide a way to modulate cognition by variations in physiology (as represented by the moderator layer; see next section).

CoJACK's parameters govern a wide range of sub-symbolic properties including: chunk activation, the time taken to retrieve or modify memory elements, and noise factors in chunk retrieval time and plan selection. Individual differences between agents can be represented by supplying differing initialization parameter values. Moderators can further affect these cognitive parameters at runtime—a "fatigue" moderator can increase the time taken to retrieve or modify memory elements.

JACK and agent architectures do not typically exhibit variability in their performance. It is a desirable feature, even, that they solve the problem in the best way every time, which is often the same way. Human cognition and performance, however, varies. It appears to vary across individuals, and it appears to be modified by outside factors such as what we eat and drink, and by internal processes such as how we appraise a situation. We will need to include these sources of variability in CoJACK.

[1] Even if an intention has, say, 50 steps, it will be treated as a single WM item. Thus, if the modeler creates a plan with 50 steps, it signifies that the plan is "compiled" and therefore its WM burden is low.

## 3.2 Design and implementation of SA

SA is also an important concept. There are two general ways in which the SA of CoJACK agents can be modified: via the perceptual module and via cognitive processes. The perceptual module primarily affects SA by filtering sense data to account for things such as focus of attention and information overload. Cognitive processes affect higher-level SA because this requires cognitive effort. If the agent is involved in a cognitively intense task, it may delay or discard completely the processes related to building higher-level SA. If too much effort is devoted to these processes, the agent also may miss important events at the perceptual level.

These points will play out in the system based on how the SA is implemented and how well it can be interpreted.

The SA feature in CoJACK (Evertsz, Ritter, Busetta, & Pedrotti, 2008; Yngling, 2008) allows the system modeler to inspect the level of SA, according to Endsley's (1995) model of SA. JACK detects and tags a belief as an Endsley level 1 (perception) or 2 (comprehension) when making use of some simple extensions. Comprehension entails inferring the implications of perception for the task at hand. Beliefs are augmented with the origins of beliefs, including the level, the name of the plan that generated the belief, and the context (the values of variables in the plan at the time of addition of the beliefs). Level 3 beliefs (projection) have to be explicitly tagged by the system modeler, and have to be computed by the model using reasoning. Projection is the most complex aspect of situation awareness. It uses the results of levels 1 and 2 to predict what will happen in the near term to the objects comprehended in level 2. Future work should help explore how to represent SA using this approach, and what level 3 SA will mean in a computational cognitive architecture.

## 3.3 Design and implementation of moderators

Each moderator is represented as a mathematical function denoting its input/output mapping. Moderators can also have internal reservoirs and decay functions that determine how the reservoir level varies over time (e.g., to model the rate caffeine is absorbed and excreted, but this is not used in these examples). Composite moderators can be created that represent the interaction between one or more moderators (e.g., the interaction between caffeine and alcohol, and how that affects reaction time). The inputs to a moderator can be (i) events from the simulation environment, (ii) internal events (e.g., the timing and amount of caffeine dosage where the simulation environment does not provide such data), and (iii) outputs from other moderators.

This representation supports creating multiple types of moderators. In Section 4 we demonstrate the use of three moderators to explore variability.

## 3.4 Summary

CoJACK is a novel architecture based on a BDI agent architecture that has been modified to include timing predictions, stochastic elements, and a theory of cognitive and behavioural moderators. The timing predictions are created by adding time costs to the agent's internal and external actions. The time of each action are influenced by a set of parameters taken from ACT-R and modified slightly to correspond to BDI elements (Norling & Ritter, 2004a, 2004b). So, instead of selecting rules to apply, the mechanisms select plans to apply. Noise parameters thus influence the selection of plans instead of rules.

CoJACK is supplied with a library of configurable cognitive parameters (Norling & Ritter, 2004b; Yngling, 2008). CoJACK includes GUIs for setting parameters and moderators, selecting scenarios, specifying data logging and tracing, and for data monitoring and visualization. It will take further work to understand how well this mapping of the parameters works, but a preliminary test is provided in the next section.

CoJACK includes a range of supportable internal (from the agent) and external (from an environment) moderators made from these parameters. These include task appraisal (challenged and threatened) and the effects of caffeine. We have explored other moderators, including fear and fatigue.

SA in CoJACK is moderated by factors influencing attention and working memory. The decay rate of memory and processing speed both have been shown to influence performance (Ritter & Bittner, 2008; Ritter, Kase, Bhandarkar, Lewis, & Cohen, 2007). The moderators tested so far show that parameters related to decay of working memory have effects on performance, and we can expect that the moderators will have an effect on cognition and the aspects of cognition related to each level of SA, which are examined in turn.

Table 1 provides a summary comparison of CoJACK, JACK, and two prominent cognitive architectures, Soar (Laird, 2012) and ACT-R (Anderson, 2007). CoJACK uses components from the other architectures, so as the table suggests, CoJACK is not completely different from them. CoJACK is attempting to copy how ACT-R generates timing predictions and errors, so this is not intended to represent a large difference.

There are a few places, however, where CoJACK is different. The table highlights two differences. The first is that CoJACK uses a higher-level knowledge representation. This is a significant difference. The second is that CoJACK includes moderators explicitly in its core functionality. This can be implemented in ACT-R, but the moderators are not included in ACT-R's graphical interface or textual trace.

Table 1. Comparison of CoJACK with ACT-R and Soar.

| | **CoJACK** | **ACT-R** | **Soar** | **JACK** |
|---|---|---|---|---|
| **Knowledge Representation** | Beliefs, Plans (BDI) | Productions | Problem Space Computational Model (PSCM) implied in rules | Beliefs, Plans (BDI) |
| **Programming Elements** | Plans, Beliefs, Events | Rules, Declarative chunks | Rules | Plans, Beliefs, Events |
| **Timing Predictions** | Yes | Yes | Can be created | No |
| **Retrieval errors** | Commission, Omission | Commission, Omission | Possible | None |
| **Moderators** | Yes | Can be done | Not routinely | None |

# 4. Results and Discussion

To demonstrate the CoJACK architecture and the effects of variability, battles were created in dTank (v. 4.4 of March 2008), a lightweight SE designed to support projects like this. These

battles involved three types of moderated CoJACK agents as well as versions with a range of moderated variables.

## 4.1 Tank commanders

The tank platforms are represented separately from their commanders. As a simplification, the commanders manipulate the tanks directly (rather than through crew members). These commanders, then, represented the cognition and internal performance of the tanks. The use of multiple models allows for analysis of variability based on the model parameters and effectiveness.

### 4.1.1 Java, JACK, and default CoJACK commanders

The Java commander implements the basic dTank tactics, which were taken from the simple default knowledge used to create previous tanks (Ritter et al., 2007). The basic subtasks are shown in Figure 3. This knowledge makes the commander capable of moving forward, turning, rotating its turret, aiming at a target, and firing. This strategy relies on a directed form of wandering that prevents it from merely rotating in place. After it spots an enemy, it attacks.

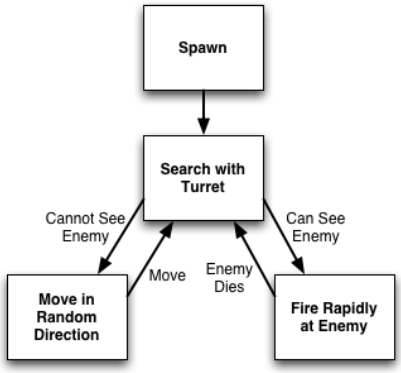



Figure 3. Default knowledge in the tank commanders.

This is a simplistic knowledge set. It includes stochastic elements, which have the potential to obscure results. However, this knowledge set has several advantages. It is based on observations of how people play dTank; it can be implemented in most architectures fairly quickly; it is easily understood and has been used in previous work (Ritter et al., 2007).

The JACK tank commander provides a comparison with a non-cognitive agent. This commander is included in the CoJACK v2 distribution. The other commanders use the same plans and belief sets. The default CoJACK tank commander was created using the CoJACK architecture (Norling & Ritter, 2004a). Parameter settings for the CoJACK commander were created using the default values of the CoJACK system. Table 2 shows the parameter values for the default CoJACK commander and the moderated commanders. These parameters in CoJACK are based on the corresponding ACT-R parameters and applied to declarative memory elements, plans, and instantiated plans.

Table 2. CoJACK commander settings.
(AR: Adjustment Ratio between default values and moderated values)

| Memory Parameter | Default CoJACK | Caffeine AR | Caffeine Value | Challenged AR | Challenged Value | Threatened AR | Threatened Value |
|---|---|---|---|---|---|---|---|
| Activation noise | 0.3 | -0.05 | 0.285 | -0.1 | 0.27 | 1.0 | 0.6 |
| Base level constant | 1.0 | nc | 1.0 | 0.2 | 1.2 | -0.15 | 0.85 |
| Decay rate | 0.5 | -0.05 | 0.475 | nc | 0.5 | 0.2 | 0.60 |
| Default action time (s) | 0.05 | -0.05 | 0.0475 | -0.1 | 0.045 | 0.3 | 0.065 |

### 4.1.2 Moderated CoJACK commanders

The basic CoJACK commander was modified using static moderator overlays representing the effects of Caffeine, a Challenged task appraisal, and a Threatened task appraisal. The same knowledge set was used for each commander type, however, parameter values were manipulated dependent upon the agent. The Caffeine agent's changes to performance were taken from a review of caffeine's effects on cognition prepared for a similar project (Morgan et al., 2006) for a fixed, moderate amount of caffeine. The effects of task appraisal for Challenged and Threatened agents were similar to and taken from previous work for representing changes from stress to ACT-R and applied, broadly defined, to CoJACK (Ritter, Reifers, Klein, & Schoelles, 2007) based on task appraisal theory (e.g., Cannon, 1932; Lazarus & Folkman, 1984; Selye, 1976). There may be better sets of parameter changes.

## *4.2 Battles*

Battles took place on an empty 5 km X 5 km dTank terrain, shown in Figure 4, and consisted of a four on four battle of Sherman tanks. Each battle lasted until a side won by destroying all of the other tanks (times range was typically 1,000-8,000 simulated seconds). Ties resulted when the last remaining tanks would destroy each other simultaneously. Forty runs were performed for each agent scenario.

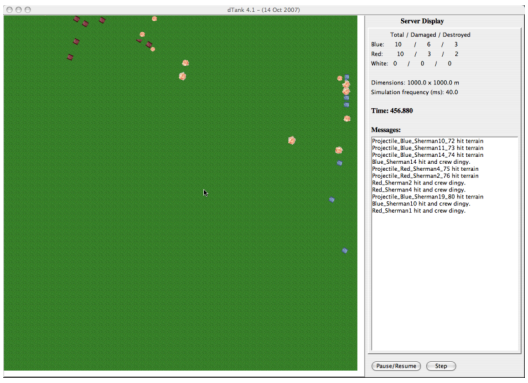



Figure 4. dTank terrain used for testing.

Two control conditions were used as comparison baselines for CoJACK agents. These were Java tanks versus JACK agents and JACK versus JACK agents. This created non-varying and non-cognitive matched comparisons. To test for variability effects with the addition of a cognitive component, JACK versus default CoJACK battles were conducted. Additionally, the moderated CoJACK agents were tested at various ratios of their moderator against JACK agents.

## *4.3 Model comparisons*

There are many types of variables that could be chosen for analysis of the battles listed. We chose the number of tanks destroyed because it appeared sensitive to changes and is easy to interpret. The paths of agents are also analyzed as they are another way that differences and variability can be measured.

Figure 5 shows the number of JACK tanks destroyed by each modeled agent. The Java results are reliably different from all the agents, ($p<.05$, two-tailed). CoJACK agents destroyed less tanks than JACK[2], which is quite acceptable because these agents are supposed to be slower and make more errors. Default CoJACK agents should and do destroy less tanks than CoJACK Caffeine agents ($p<.05$), and destroy less than the CoJACK Challenged agents. CoJACK Threatened agents do not destroy fewer tanks, and it should be lower according to appraisal theory. This effect could be due to the task, where being aggressive is not always better (shooting slightly later is slightly more accurate because the target is closer); it could be stochastic variation; and it could be that we have not implemented the effects of threatening appraisal correctly.

Overall, these differences show that CoJACK performs differently than an AI agent, and that the moderated versions are different from the AI agent and each other. While the results are not final, they are promising.

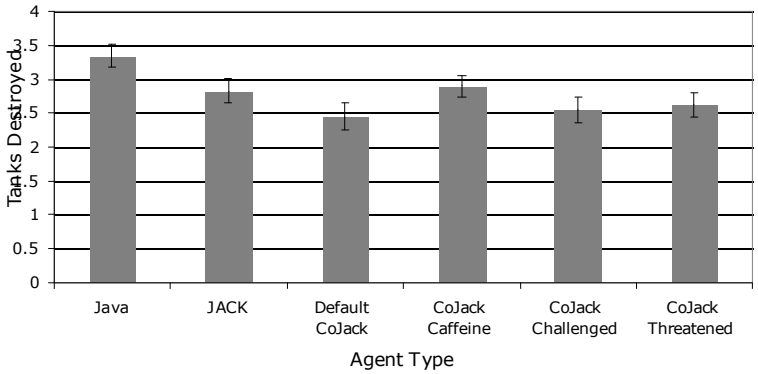



Figure 5. Basic comparison between agents (N=40). (Error bars show SEM)

Figure 6 shows the effects of CoJACK moderator overlays across a range of ratio adjustments. The specific parameter values were calculated by applying a weighting to the original ratio such that a ratio of 1.0 reflects the changes shown previously in Table 2, a ratio of 2.0 doubles the adjustment ratio (thus doubling the effects on the agent), and so forth. Each level contains 40 runs versus 4 JACK tanks.

[2] This effect is only marginally reliable, but with additional runs it is likely to become reliable. Differences where the error bars do not overlap are reliable differences on 1-tailed t-tests at .05.

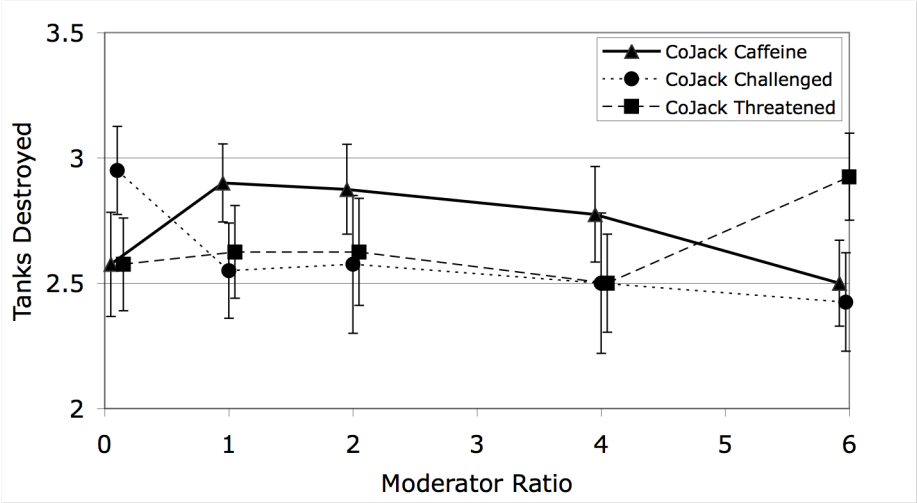



Figure 6. Effects of CoJACK Moderator Overlays. SEM shown as error bars. Bars that do no overlap are marginally significant to significant ($p<.05$, two tailed).

Although the overall performance of each moderator in terms of tanks destroyed is in the same general range (shown here from 2.0 to 3.5 tanks destroyed per battle), three distinct patterns arise across parameter modifications. As the moderator ratio approaches zero, the tanks destroyed value should reach the default CoJACK result (which is about 2.5, shown in Figure 5). Here it appears that CoJACK Caffeine and Threatened do just that. However, CoJACK Challenged as it approaches the default CoJACK settings produces a result that appears to be greater than that of the default value. We have checked our moderator settings (which correctly approach the default settings) and are investigating this anomaly. It does show that a useful test for moderators is that as they decrease they should approach the architectural default.

As was hypothesized for caffeine, the default effect (for an optimal amount of caffeine) led to better performance. An inverted U-shaped curve appears to be emerging across the various caffeine values. This may be due to the effect of the parameters. It could also be due to the task having a non-linear response to performance speed. Additional testing is required to fully understand this relationship, but this moderator at least changes performance.

CoJACK Challenged moderation produces a different pattern. It appears that as the effect of the Challenged task appraisal increases between 0.1 and 1.0 a drop off in performance occurs followed by consistent performance across the higher levels. This is somewhat surprising, as the parameter changes should represent better processing, yet, the effects are a decrement in performance. This could reflect that caution in this environment lead to better performance.

CoJACK Threatened moderation should lead to poorer performance. Performance initially decreases as the agent is more threatened, and then performance improves. This increase in performance for higher levels of the threatened moderator may be caused by changes in performance similar to where higher levels of the default action time (which also occur here) lead to improved performance because shots are more accurate.

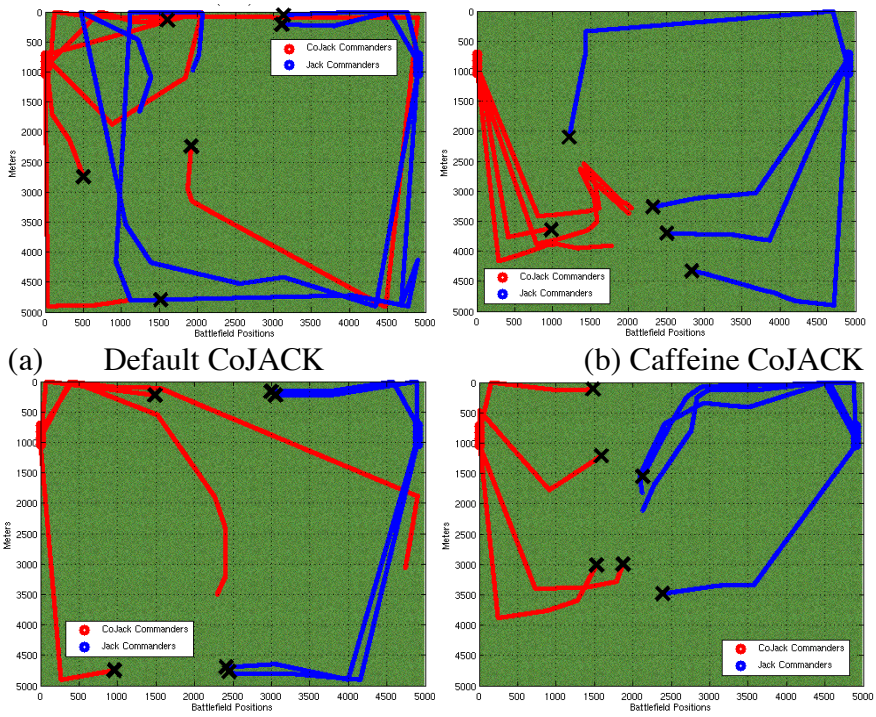
These moderators touch just a few of the parameters in CoJACK and even fewer of their combinations. Work with ACT-R has suggested that parameters are related, and we know that they interact (Ritter et al., 2007). Moderators in CoJACK will influence performance, and they influence a range of mechanisms. For a single moderator, it might not influence all the

mechanisms and thus might have a limited or more limited effect on SA. In particular, moderators that affect motor performance are more likely to have little direct effect on SA.

We did not combine these moderators as we do not have intuitions let alone data for testing the combination. We would, however, predict that caffeine would lead to a less threatening appraisal because many studies report that caffeine leads to greatly improved measures of self-reported alertness (Morgan et al., 2006).

Figure 7 shows another way to understand variability, single run traces of each tank by each CoJACK agent. Path patterns vary across agent types. Anecdotally, CoJACK (the default and the moderated versions) generally has a wider and less consistent set of movement patterns than the corresponding JACK agents. The default CoJACK agent appears to have the greatest variability within and between tank paths while CoJACK Caffeine has the smallest spread and variability within tanks. Caffeine tanks appear to choose the same general direction and pattern as each other. CoJACK Threatened is generally like the CoJACK Caffeine agents, however, there is greater variability within the tank paths. CoJACK Challenged shows greater spread and variability between tanks.

This type of analysis is yet another way to demonstrate how differences and variability can be measured within and across moderator overlays. It is only preliminary, but it suggests that there are interesting differences that arise as part of variability.




(a) Default CoJACK (b) Caffeine CoJACK

(c) Challenged CoJACK (d) Threatened CoJACK

Figure 7. Example traces of tank paths for CoJACK agents (in red) versus JACK (in blue).

# 5. Conclusions

We have introduced a novel cognitive architecture based on BDI agents. CoJACK adds a set of constraints to the JACK agent architecture to provide predictions of time, errors, and the effects

of moderators while retaining the high level representation of a BDI architecture. The result is a cognitive architecture with usability provided by an existing high-level language, as well as GUIs and support materials for setup, data monitoring, recording and analysis of the cognitive architecture mechanisms that have been added. We included an explicit, traceable representation of agent SA, initially based on Endsley's (1995) representation of levels of situation awareness.

The resulting approach of modified agent mechanisms provides a generic technique for representing individual differences and the influence of internal and external moderators of cognition, and the effect of variation in that state on behavior. The resulting behavior can be extended to be empirically-based (e.g., on caffeine and task appraisal studies) or used to explore difficult to validate effects, such as fear and group cohesion, applied to create agents in SEs, or a combination of these.

We have used CoJACK in other tasks. We have used CoJACK to explore the effect of rules of engagement (ROE) on performance (Evertsz, Ritter, Russell, & Shepherdson, 2007). There, we found that applying ROEs slow down behavior and would be influenced by factors that influence cognition. We have also used CoJACK in a demonstration at the Combined Arms Tactical Trainer at Warminster, UK showing how different task appraisals would lead to different behavior in a tank squadron ambush scenario. And, we have used CoJACK to model how fear and morale vary and lead to action and inaction in an improvised explosive device scenario (Evertsz, Pedrotti, Busetta, Acar, & Ritter, 2009).

## *5.1 Implications for modeling variability*

The development of CoJACK so far has shown that including variability in an architecture is not just having it behave randomly. We started to add more realistic variability by modifying the information processing mechanisms in the agent and adding noise to how declarative knowledge was retrieved and which procedural knowledge was applied. Further work is needed to include a wider range of knowledge in the agent so that the effects of these choices are larger. We have seen how reasonable variability appears to include variability in choice between reasonable strategies, variation in pace of decisions and accuracy of choice, and how these lead to variations in physical paths.

Figure 8 shows a relationship we can hypothesize between variation and performance in a competitive environment. With low variability, behavior is predictable. As long as the opponent does not learn enough, then low variability may lead to good performance. When the opponent learns, such as users of SEs in training situations, then the opponent knows what will happen and counteracts that behavior. With increased variability, an agent becomes more difficult to predict, and we can hypothesize that there is an amount of variability that is optimal. Above that amount, variability reduces your ability to carry out your own tasks and also makes it difficult to coordinate behavior across teams.

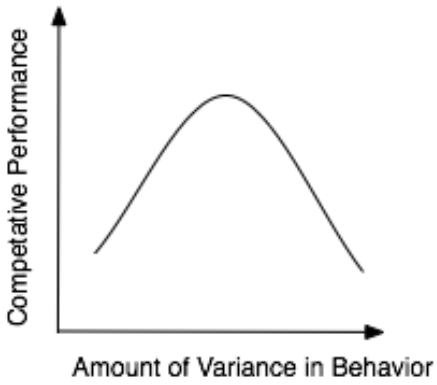



Figure 8. An inverted U-shaped curve of variability in behavior and performance.

This curve of variability versus performance seems to have been little studied in current competitive environments. This might be for two reasons. One reason is that very few agents in SEs learn, so variability is neither rewarded nor required. The other reason is that teamwork in agent teams has probably mostly existed through set performances and cannot easily vary. Thus, it is difficult to include variability in this type of teamwork, so variability has been discouraged. This hypothesized relationship lays the theoretical groundwork for the study of variability, which we only started to explore in this paper.

Opposing agents will need learning to be able to recognize and take advantage of the lack of variability. Otherwise, they can always be surprised by what they see. So, an implication of this is not that only that training against a fixed agent leads to gaming by the learner, where they know what the agent will do, the lack of learning in the opponent will also lead to the learner producing a fixed response, and performing predictably. While this result is useful in early training, it is less useful at higher levels of expertise.

## *5.2 Implications for moderated architectures*

CoJACK provides several lessons for developing cognitive architectures. First, it suggests that a way forward for creating easy to use cognitive architectures is to modify existing agent architectures to create cognitive architectures. Our example, CoJACK, appears to be a more useful architecture, but one that needs to be tested further. These architectures may be particularly useful in simulations where only static or predictable agents were used before.

The example runs provide lessons for testing cognitive architectures. The first lesson is that architectures with stochastic elements will need to run their models multiple times (Ritter, Schoelles, Quigley, & Klein, 2011), which we have done. The second lesson is that the parameter settings will need to be explored before the architecture is deployed. Work exploring parameter spaces (Gluck, 2008; Kase, 2008; Kase, Ritter, & Schoelles, 2008; Ritter, 1991) suggests that the range of parameters that modelers use, and the way that they find these parameters, might not be finding the right parameters. Thus, the developers for CoJACK and similar architectures will want to explore their architecture's parameter spaces.

## *Acknowledgements*

This work was supported by the UK MoD's Directorate of Analysis, Experimentation and Simulation corporate research programme (Project No: RT/COM/3/006). We thank Susan

Chipman, Simon Goss, and Harold Hawkins for useful discussions about modeling, and Ian Greig, Roy McNee, Bharat Patel, and Colin Sheppard for useful discussions about CoJACK. Bil Lewis, along with Damodar Bhandikar and Jeremiah Hiam, helped develop dTank. ONR N00014-06-1-0164 helped support dTank. Comments from Olivier Georgeon, Greg Plumb, participants at the ARL Workshop on Developing and Understanding Computational Models of Macrocognition, and anonymous reviewers of previous versions of this paper as related conference papers, as well as six very responsive, helpful, and kind BICA reviewers helped improve this presentation.

# References

Anderson, J. R. (2007). *How can the human mind exist in the physical universe?* New York, NY: Oxford University Press.

Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, C., & Qin, Y. (2004). An integrated theory of the mind. *Psychological Review, 111*(4), 1036-1060.

Anderson, J. R., & Lebiere, C. (1998). *The atomic components of thought*. Mahwah, NJ: Erlbaum.

Bach, J. (2008). *Principles of synthetic intelligence: Building blocks for an architecture of motivated cognition*. New York, NY: Oxford University Press.

Bates, J. (1994). The role of emotion in believable agents. *Communications of the ACM, 37*(7), 122-125.

Belyavin, A. J., & Spencer, M. B. (2004). Modeling performance and alertness: The QinetiQ approach. *Aviation, Space, and Environmental Medicine, 75*(3, Section II), A93-A103.

Best, B. J., Lebiere, C., & Scarpinatto, K. C. (2002). Modeling synthetic opponents in MOUT training simulations using the ACT-R cognitive architecture. In *Proceedings of the 11th Computer Generated Forces Conference*, 505-516, 502-CGF-056. U. of Central Florida: Orlando, FL.

Boff, K. R., & Lincoln, J. E. (Eds.). (1988). *Engineering data compendium: Human perception and performance, Volumes I and II*. Wright-Patterson Air Force Base, OH: Harry G. Armstrong Aerospace Medical Research Laboratory.

Booher, H. R., & Minninger, J. (2003). Human systems integration in Army systems acquisition. In H. R. Booher (Ed.), *Handbook of human systems integration* (pp. 663-698). Hoboken, NJ: John Wiley.

Busetta, P., Rönnquist, R., Hodgson, A., & Lucas, A. (1999). JACK intelligent agents - Components for intelligent agents in JAVA. *AgentLink News Letter, 2*(Jan.), www.agent-software.com/white-paper.pdf.

Cannon, W. B. (1932). *The wisdom of the body*. New York, NY: Norton.

Endsley, M. R. (1995). Toward a theory of situation awareness in dynamic systems. *Human Factors, 37*(1), 32-64.

Evertsz, R., Pedrotti, M., Busetta, P., Acar, H., & Ritter, F. E. (2009). Populating VBS2 with realistic virtual actors. In *Proceedings of the 18th Conference on Behavior Representation in Modeling and Simulation*, 09-BRIMS-04.

Evertsz, R., Ritter, F. E., Busetta, P., & Pedrotti, M. (2008). Realistic behaviour variation in a BDI-based cognitive architecture. In *Proceedings of SimTecT 2008*, 245-250. SIAA Ltd.: Melbourne, Australia.

Evertsz, R., Ritter, F. E., Russell, S., & Shepherdson, D. (2007). Modeling rules of engagement in computer generated forces. In *Proceedings of the 16th Conference on Behavior*

*Representation in Modeling and Simulation*, 123-134. 107-BRIMS-021. U. of Central Florida: Norfolk, VA.
Gluck, K. (2008). Large-Scale Computing Resources and ACT-R Modeling Symposium: Held at the 2008 ACT-R Workshop.
Gratch, J., & Marsella, S. (2004). A domain-independent framework for modeling emotion. *Journal of Cognitive Systems Research, 5*(4), 269-306.
Gunzelmann, G., Gross, J. B., Gluck, K. A., & Dinges, D. F. (in press). Sleep deprivation and sustained attention performance: Integrating mathematical and cognitive modeling. *Cognitive Science*.
Harrison, A., Winters, J., & Anthistle, D. (1999). Ironside A command and battle space simulation. In *Proceedings of the 1999 Summer Computer Simulation Conference*, 550-554. http://www.scs.org/scsarchive/getDoc.cfm?id=418. The Society for Modeling & Simulation International.
Haynes, S. R., Cohen, M. A., & Ritter, F. E. (2009). Designs for explaining intelligent agents. *International Journal of Human-Computer Studies, 67*(1), 99-110.
Howes, A., & Young, R. M. (1997). The role of cognitive architecture in modeling the user: Soar's learning mechanism. *Human-Computer Interaction, 12*, 311-343.
Hudlicka, E. (2002). This time with feeling: Integrated models of trait and state effects on cognition and behavior. *Applied Artificial Intelligence, 16*, 611-641.
Jonassen, D. H., & Grabowski, B. L. (1993). *Handbook of individual differences, learning, and instruction*. Hillsdale, NJ: Erlbaum.
Jones, R. M., Laird, J. E., Nielsen, P. E., Coulter, K. J., Kenny, P., & Koss, F. V. (1999). Automated intelligent pilots for combat flight simulation. *AI Magazine, 20*(1), 27-41.
Kase, S. E. (2008). *Parallel genetic algorithm optimization of a cognitive model: Investigating group and individual performance on a math stressor task.* Unpublished PhD thesis, College of IST, Penn State University, University Park, PA.
Kase, S. E., Ritter, F. E., & Schoelles, M. (2008). From modeler-free individual data fitting to 3-D parametric prediction landscapes: A research expedition. In *Proceedings of the 30th Annual Conference of the Cognitive Science Society*, 1398-1403. Cognitive Science Society: Austin, TX.
Laird, J. E. (2001). It knows what you're going to do: Adding anticipation to a Quakebot. In *Proceedings of the Fifth International Conference on Autonomous Agents*, 385-392. ACM Press: New York, NY.
Laird, J. E. (2012). *The Soar cognitive architecture*. Cambridge, MA: MIT Press.
Lazarus, R. S., & Folkman, S. (1984). *Stress, appraisal and coping*. New York: Springer Publishing.
Lovett, M. C., Daily, L. Z., & Reder, L. M. (2000). A source activation theory of working memory: Cross-task prediction of performance in ACT-R. *Journal of Cognitive Systems Research, 1*, 99-118.
Morgan, G. P., Ritter, F. E., Stine, M. M., & Klein, L. C. (2006). The cognitive effects of caffeine: Implications for models of users: unpublished mss.
Morgan, J. H., Morgan, G., & Ritter, F. E. (2010). A preliminary model of participation for small groups. *Computational and Mathematical Organization Science, 16*, 246-270.
Nielsen, P. E., Crossman, J., & Jones, R. M. (2007). Human factors in opponent intent. In A. Kott & W. M. McEneaney (Eds.), *Adversarial reasoning: Computational approaches to reading the opponent's mind*. Boca Raton, FL: Chapman & Hall.

Norling, E., & Ritter, F. E. (2004a). A parameter set to support psychologically plausible variability in agent-based human modelling. In *The Third International Joint Conference on Autonomous Agents and Multi Agent Systems (AAMAS04)*, 758-765. ACM: New York, NY.

Norling, E., & Ritter, F. E. (2004b). *COJACK Software Specification*: Agent Oriented Software Limited.

Pew, R. W., & Mavor, A. S. (Eds.). (1998). *Modeling human and organizational behavior: Application to military simulations*. Washington, DC: National Academy Press. books.nap.edu/catalog/6173.html.

Pew, R. W., & Mavor, A. S. (Eds.). (2007). *Human-system integration in the system development process: A new look*. Washington, DC: National Academy Press. http://books.nap.edu/catalog.php?record_id=11893, checked March 2012.

Poncelin de Raucourt, V. P. M. (1997). *The reconstruction of part of the Battle of Medenine.* Unpublished MSc thesis, RMCS, Cranfield University.

Rao, A. S., & Georgeff, M. P. (1995). BDI-agents: From theory to practice. In *Proceedings of the First International Conference on Multi-Agents-Systems*, 312-319. AAAI Press: Menlo, CA.

Ritter, F. E. (1991). Towards fair comparisons of connectionist algorithms through automatically generated parameter sets. In *Proceedings of the 13th Annual Conference of the Cognitive Science Society*, 877-881. Erlbaum: Hillsdale, NJ.

Ritter, F. E. (2009). Two cognitive modeling frontiers: Emotions and usability. *Journal of Japanese AI Research, 24*(2), 241-249.

Ritter, F. E., & Bittner, J. L. (2008). *Moderators in CoJACK* (Deliverable 2.1.2, PSU Task 2B Report).

Ritter, F. E., Haynes, S. R., Cohen, M. A., Howes, A., John, B., Best, B., et al. (2006). High-level behavior representation languages revisited. In *Proceedings of ICCM - 2006- Seventh International Conference on Cognitive Modeling*, 404-407. Edizioni Goliardiche: Trieste, Italy.

Ritter, F. E., Kase, S. E., Bhandarkar, D., Lewis, B., & Cohen, M. A. (2007). dTank updated: Exploring moderator-influenced behavior in a light-weight synthetic environment. In *Proceedings of the 16th Conference on Behavior Representation in Modeling and Simulation*, 51-60. 07-BRIMS-014. U. of Central Florida: Norfolk, VA.

Ritter, F. E., Reifers, A. L., Klein, L. C., & Schoelles, M. J. (2007). Lessons from defining theories of stress for architectures. In W. Gray (Ed.), *Integrated models of cognitive systems* (pp. 254-262). New York, NY: Oxford University Press.

Ritter, F. E., Schoelles, M. J., Quigley, K. S., & Klein, L. C. (2011). Determining the number of model runs: Treating cognitive models as theories by not sampling their behavior. In L. Rothrock & S. Narayanan (Eds.), *Human-in-the-loop simulations: Methods and practice* (pp. 97-116). London: Springer-Verlag.

Ritter, F. E., Shadbolt, N. R., Elliman, D., Young, R. M., Gobet, F., & Baxter, G. D. (2003). *Techniques for modeling human performance in synthetic environments: A supplementary review*. Wright-Patterson Air Force Base, OH: Human Systems Information Analysis Center (HSIAC).

Selye, H. (1976). *The stress of life*. New York, NY: McGraw-Hill.

Shakir, A. (2002). Assessment of models of human decision-making for air combat analysis. [Unpublished technical report. Abstract, put on the web with permission, at http://acs.ist.psu.edu/papers/shakir02-abstract.pdf].

Simon, D. P., & Simon, H. A. (1978). Individual differences in solving physics problems. In R. S. Siegler (Ed.), *Children's Thinking: What Develops?* (pp. 325-348). Hillsdale, NJ: Erlbaum.
Yngling, A. A. (2008). *CoJACK2 Moderators - End User Documentation* (Technical Report No. RT/COM/3/006): Agent Oriented Software.