Behavior modeling through CHAOS for simulation of dismounted soldier operations

Emiel Ubink[1], Frank Aldershoff, Wouter Lotens, Arend Woering
TNO Defense, Security & Safety, Kampweg 5, Soesterberg, The Netherlands, +31 346 356 322

ABSTRACT

One of the major challenges in human behavior modeling for military applications is dealing with all factors that can influence behavior and performance. In a military context, behavior and performance are influenced by the task at hand, the internal (cognitive and physiological) and external (climate, terrain, threat, equipment, etc.) state. Modeling the behavioral effects of all these factors in a centralized manner would lead to a complex rule-base that is difficult to maintain or expand. To better cope with this complexity we have developed the Capability-based Human-performance Architecture for Operational Simulation (CHAOS). CHAOS is a multi-agent system for human behavior modeling that is based on pandemonium theory. Every agent in CHAOS represents a specific part of behavior, such as 'reaction to threat' or 'performing a patrol task'. These agents are competing over a limited set of resources that represent human capabilities. By combining the element of competition with multiple limited resources, CHAOS allows us to model stress, strain and multi-tasking in an intuitive manner. The CHAOS architecture is currently used in firefighter and dismounted soldier simulations and has shown itself to be suitable for human behavior and performance modeling.

Keywords: human behavior representation, human performance modeling, cognitive architectures, dismounted soldier operations, multi-agent systems, military simulation

1. INTRODUCTION

Software simulation of soldier operations provides a means of assessing the potential effect of new operational tactics or new equipment on soldier performance. By means of simulation, controllable experiments in realistic operational settings can be conducted. Since human performance is a direct resultant of behavior, the behavior of the agents in such a simulation tool aimed at human performance research should be as realistic as possible.

One method for creating agents with realistic behavior is by completely scripting the behavior. Although this approach is adopted by many simulation tools, it has three major drawbacks. The first drawback is that for every scenario the scripts need to be adjusted, which takes a lot of time and effort. The second drawback is that complex scenarios (for instance scenarios that take several hours and incorporate several engagements) are almost infeasible. This is caused by the fact that all events in the scenario have to be handled by the script. Since events with multiple outcomes will lead to multiple branches of the scenario, the number of events that has to be handled can grow exponentially with scenario duration and complexity. The third drawback is related to human performance modeling. Since human performance is the *result* of behavior, behavior should be influenced *dynamically* by, for instance, stress or strain. If the behavior is completely scripted, then the resulting performance will be –at least to some extent– scripted as well.

In order to overcome these problems we have developed the Capability-based Human-performance Architecture for Operational Simulation (CHAOS). CHAOS is a behavior framework suitable for human behavior and performance modeling. It allows us to intuitively model multi-tasking and performance degradation due to stress or strain. CHAOS is a multi-agent system based on the pandemonium model of letter recognition as proposed by Selfridge [1]. Each agent in CHAOS represents a specific type of behavior. This allows for simple modification of behavior by adding agents to, or removing agents from the system. In this paper we will explain how this decentralized design simplifies the behavior modeling process.

CHAOS is currently used in two ongoing projects: Simulation for Assessment of Firefighter Effectiveness (SAFE) and Soldier Capability Optimization for Projected Efficacy (SCOPE). SAFE is a simulation of firefighter operations, SCOPE is a simulation of small unit infantry operations. Both simulation tools are developed by TNO Defense, Security &

[1] emiel.ubink@tno.nl

Modeling and Simulation for Military Operations III, edited by Dawn A. Trevisani
Proc. of SPIE Vol. 6965, 696504, (2008) · 0277-786X/08/$18 · doi: 10.1117/12.774650

Safety and are aimed at research in the field of operational performance, in close connection with field and lab experiments.

This paper is organized as follows: Section 2 will outline related work in the field of human behavior modeling and infantry simulation. In section 3 the CHAOS behavior framework will be presented. Section 4 gives a brief overview of SCOPE, an infantry simulation that incorporates CHAOS. In section 5 a study performed using SCOPE on the effect of fitness, acclimatization and ballistic protection on operational performance will be described. Finally in section 6 conclusions are drawn on the applicability of CHAOS to human behavior representation and human performance modeling.

2. RELATED WORK

2.1 Behavior modeling frameworks

Many frameworks for modeling behavior exist (see for instance the overviews in [2] and [3]). Most of them are considered *cognitive architectures*, i.e. models of cognition that are inspired by theories from cognitive neuroscience and cognitive psychology. Given the nature of these theories, cognitive architectures are often low-level, highly modular models of brain functionality. Each module in the architecture represents a specific cognitive function. Good examples of this approach are ACT-R [4] and SOAR [5]. Another well-known cognitive architecture is COGNET [6]. COGNET is also based on cognitive science research and has a modular design similar to SOAR and ACT-R. It is also inspired on the pandemonium model as proposed by Selfridge that provided the inspiration for CHAOS as well. The similarities and differences between COGNET and CHAOS will be discussed in more detail in section 3.5.

As the name implies, cognitive architectures reduce behavior, or at least the decision process that precedes actions, to cognition. This simplification can be useful when modeling behavior of operators in front of a computer screen, which is indeed the most common application of cognitive architectures. However, as is argued by Brooks [7] and many others, behavior is often as much a matter of *embodiment* and *situatedness* as it is of cognition or rationality. This is definitely the case for the behaviors of dismounted soldiers, but even operating a computer may be influenced by the fact that the operator is an *embodied* and *situated* entity (imagine, for instance, a system that is operated with three joysticks and that is located in a noisy and crowded environment).

Another type of behavior modeling is based on the concept of *beliefs, desires* and *intentions* (BDI) [8]. This approach is rooted in philosophy rather than cognitive science and represents a notion of rationality. The *beliefs* of an agent represent the agent's informational state, i.e. what the agent knows about the world. The *desires* of an agent represent his motivations and objectives. Finally the *intentions* represent the plans the agent has to fulfill (some of its) desires. An example of a BDI system is JACK [9] that is used in the CAEN simulation tool that is described in section 2.2. Given its emphasis on rationality and information processing, BDI is not very suitable for modeling the behavior of embodied and situated entities.

2.2 Small unit infantry simulation

Many simulations of military operations have been developed. SCOPE is among the few that aim at simulating small unit operations for analysis purposes. Two other tools that do fall in this category are IWARS and CAEN. The Infantry Warrior Simulation (IWARS) [10] is a collaborative effort between the Soldier Systems Center (SSC) and US Army Materiel Systems Analysis Activity (AMSAA). The lowest simulation unit in IWARS is the single soldier, but IWARS also supports small unit operations. Most of the behavior in the current version of IWARS is scripted.

The Close Action ENvironment (CAEN) model is developed by the UK Defence Science and Technology Laboratory (DSTL). The focus of CAEN is to enable the evaluation of small-scale infantry combat scenarios. CAEN is, just as SCOPE, used to assess the effectiveness of equipment and tactics in the close combat infantry battle. The model can represent personnel and vehicles in close combat up to Company level in rural and urban areas [11].CAEN started out with completely scripted behavior. To overcome the problems related to scripted behavior, CAEN has incorporated the Simulation Agent Infrastructure (SAI) [12] which is based on JACK [9].

3. THE CHAOS BEHAVIOR FRAMEWORK

The CHAOS behavior framework is inspired on the pandemonium model of letter recognition as proposed by Selfridge. Selfridge's pandemonium consists of *demons* that are shrieking for attention in an arena called the *pandemonium*. The

demons in CHAOS represent specific types of behavior, or *behavior chunks*. These behavior chunks are competing for the control of the behavior of the entity they are part of. Besides demons, CHAOS consists of *resources* and of course the *pandemonium* itself. The resources represent the capabilities of the modeled entity while the pandemonium is the "arena" that contains the demons, and that controls and coordinates the "behavior process". These three types of components (demons, resources, pandemonium), will be described in more detail in the following sections. Since CHAOS introduces a somewhat unorthodox paradigm for decomposing an intelligent system into subsystems, we will first provide some background on the prevailing decomposition methods.

3.1 Functional decomposition

The modular design of most behavior modeling architectures is based on a decomposition along the *functional* lines of the modeled system. Some examples of *functional decompositions* are given in Figure 1. In the simple, prototypical example on the left, a specialized component deals with perceiving the world, another component deals with interpreting the things perceived, and finally there is a component that deals with (re-) acting, thereby possibly changing the environment.

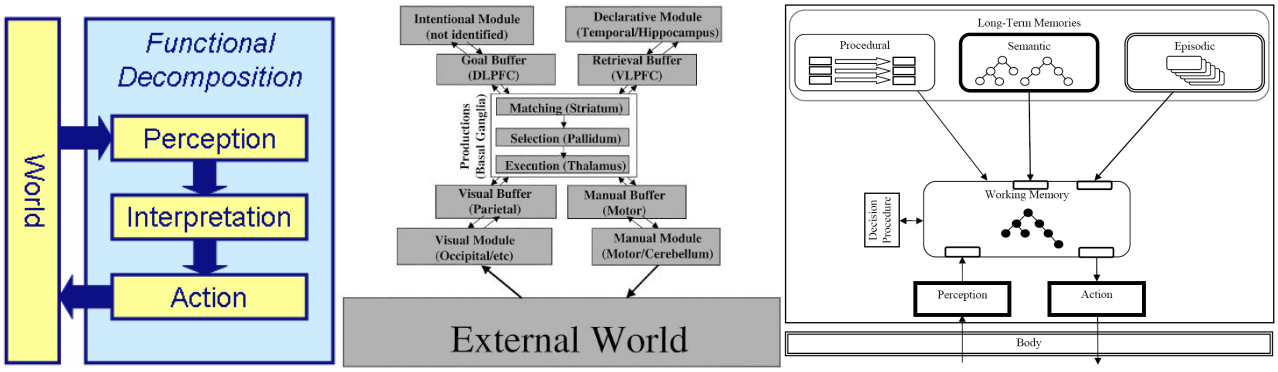

Figure 1: Three examples of functional decompositions. On the left: a simple, prototypical decomposition; in the middle: the more elaborate modular design of ACT-R [4]; on the right: the SOAR memory structure [5]. Although more complex than the prototypical example, the ACT-R and SOAR decompositions are still *functional* decompositions.

Functional decompositions have many useful applications, but are not very suitable as a basis for an embodied human behavior model in a high fidelity constructive simulation. This has to do with the wide range of behaviors that need to be represented, and the interactions and exceptions that can occur between these behaviors and the stimuli that trigger them.

To give an example: suppose we want to model a group of dismounted soldiers that has to perform a patrol task. The soldiers should, besides patrolling, be capable of reacting when fired at by taking cover and returning fire. Furthermore, when exhausted, they should take a break to recover. Suppose we model these behaviors in a system that is based on a functional decomposition, similar to the leftmost figure. In that case, the perception layer should be able to perceive many types of objects, ranging from trees and houses to other entities. To incorporate the effects of fatigue in our behavior model, it should also need to "perceive" the physiological condition of the soldiers. All this data has to be processed and interpreted by the interpretation component. This component should be able to recognize enemies, houses and trees and should also know a thing or two about threat assessment and physiology. Based on this interpretation of the data, the action layer needs to decide which actions are appropriate. Maybe the correct course of action is to simply continue the patrol task. However, if the soldiers are very tired it may be better to take a break from patrolling in order to recover, *unless* the soldiers are tired while being fired at, in which case they should take cover and return fire. In other words: the action layer has to be capable of dealing with many interactions and exceptions.

What this simple example shows is that, even though the components can be considered specialized from a *functional* perspective, from a *behavioral* perspective the components are not specialized at all. Every component contains some "knowledge" pertaining to every type of behavior. When many types of behavior need to be modeled this lack of specialization will result in a complex and difficult to maintain behavior model, because many exceptions and interactions between stimuli and behaviors will have to be handled. The obvious solution then is to base the design of the behavior model on a *behavioral* rather then *functional* decomposition. This brings us to the demons in CHAOS that, as stated above, represent "behavior chunks".

3.2 Demons

The central concept behind CHAOS is that of "chunks of behavior". Inspired by Selfridge's pandemonium, these chunks of behavior are represented as demons in a pandemonium. If we return to our example of the dismounted soldiers, the soldiers' three types of behavior (patrolling, reacting to threat, reacting to exhaustion) would be represented in CHAOS by three separate demons that are specialized from a *behavioral* rather than a *functional* perspective (see Figure 2). At first glance Figure 2 seems to be much more complex than Figure 1. However, the complexity in Figure 1 is just not *visible* since the real complexity is *within* the components.

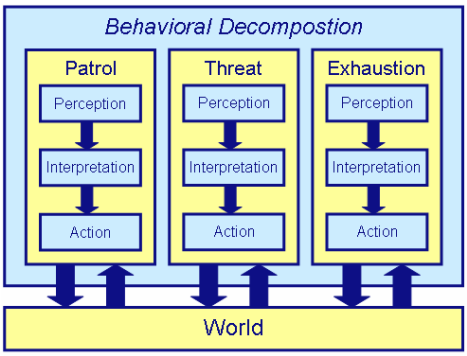

Figure 2: An example of a behavioral decomposition that could be used in CHAOS. Note that the "World" can also include the entity itself. For "Exhaustion" for instance, the internal physiological condition is perceived.

Figure 3: A specialized demon only needs to monitor a small part of the entire state space. He is only able to assess the situation in this small part of the state space, and he can adjust his shrieking level accordingly. He is also able of taking actions if necessary, but will only take actions that are related to his specialization.

When we look closer at the demons in Figure 2, it shows that the demons themselves are based on a functional decomposition. In other words: each demon is responsible for the whole "behavior chain", from perception through interpretation to action, but only for a very specific type of behavior. The demon only needs to monitor a small part of the state space. He has to be able to interpret only the situation in that specific part of the state space, and he has to be able to act accordingly (see Figure 3). From the perspective of the modeler, this specialization means that he can focus on a specific type of behavior while designing and implementing a demon, without having to deal with exceptions or interactions. If a specific type of behavior needs to be added (for instance "reacting to a chemical attack"), he only has to create a new demon that is capable of producing the lacking behavior, without the need to alter any of the existing code. Note that this does not prevent the reuse of code, as will be explained below.

The problem with having a system of these specialized demons is of course deciding which demons should at any moment control behavior, and which should not. In Selfridge's pandemonium the demons are "shrieking" for attention. This concept is adopted by CHAOS; the demons can themselves determine their shrieking level (see Figure 3). In the soldier example, the demon that represents the behavior related to enemy threat may for instance be able to assess the threat level exposed by enemies, and adjust his shrieking level accordingly. In other words, its shrieking level would depend directly on the experienced threat level.

The easiest solution for the competition between the demons would then be to allow only the loudest shrieking (i.e. most important) demon to control behavior. However, such a winner-takes-all solution would prevent us to model multi-tasking and performance degradation[2]. Therefore in CHAOS it is allowed for multiple demons to simultaneously influence behavior. The pandemonium controls which *set* of demons is allowed to determine the behavior. However, note that to do so the pandemonium does not need to *understand* the behaviors or the current situation. How the pandemonium can nevertheless determine which demons may affect behavior will be explained in section 3.4.

[2] Performance degradation occurs when some form of stress affects the performance on a task, or when two or more conflicting tasks are executed simultaneously. Since both stress and tasks are represented as demons in CHAOS, performance degradation can only be modeled if at least two demons can be active simultaneously.

Stress and Strain

Stress and strain can have a big impact on behavior. In CHAOS stress and strain are represented as demons. Let's say we want to model thermal stress, i.e. stress related to the thermal condition of a person. In that case we can implement a "thermal stress demon" that monitors, for instance, the person's body temperature, and that will shriek louder as the temperature of that person rises. To be more precise, the demon would "know" in what *range* the variable affects performance, i.e. is considered stressful, as depicted in Figure 4. The stress level would then be reflected by the demon's shrieking level.

This demon could then simulate the effects of thermal stress on performance by impairing the capabilities of the system, depending on its shrieking level. How this is achieved will be explained in detail in the following sections. Note however that besides impairing capabilities a stress demon can, just like a "normal" demon, implement a *behavioral* component. In the case of a stress demon this behavior represents the entity's strategy to *recover* or escape from the stressful situation the entity is in. In case of thermal stress this may involve taking some clothes off, eating ice cream or taking a break from physical activity.

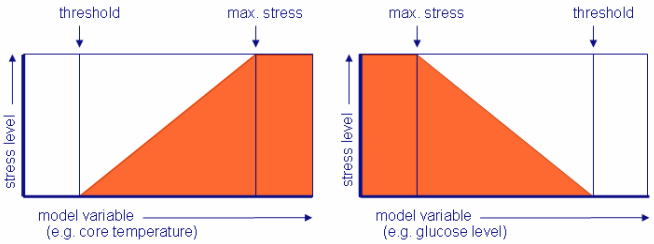

Figure 4: Representing stress in CHAOS. A stress-demon is connected to a model variable, such as body temperature. The stress-demon knows the value at which the variable will start affecting performance (stress-threshold) and also knows at which value the stress is considered maximal. The stress level is interpolated in between. Note that the stress level can also be reversed to model, for instance, glucose depletion.

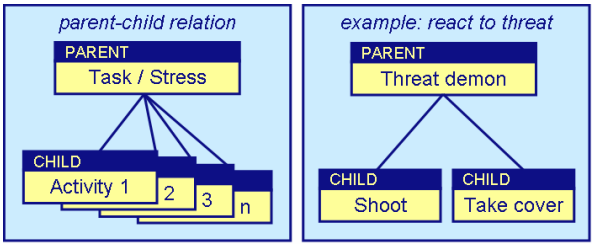

Figure 5: Hierarchical behavior modeling in CHAOS. A high level demon representing some task or stress can have several child demons as "sub-contractors". On the right is a possible implementation of the "Threat demon" that is responsible for reacting to threat. It has two children; one that is capable of shooting at the enemy, the other that is capable of taking cover.

Hierarchical behavior

In the discussion above we proposed that "reacting to threat" is potentially one of the demons in a soldier's pandemonium. However, if we look closer at this "behavior chunk" it seems to be quite complex, involving for instance taking cover and returning fire. If we would model behavior of this complexity as a single chunk the advantages of our behavioral decomposition may be lost. Therefore, in CHAOS behavior can be structured hierarchically. This means that each demon can have child-demons that can be considered "sub-contractors", responsible for performing a specific aspect of the more complex behavior of its parent. In our example, the parent demon would be the demon responsible for reacting to threat and would be mainly concerned with assessing the threat level. It has two child demons, a demon that is capable of finding and taking cover and another demon that is capable of shooting at an enemy (Figure 5). These child-demons can be considered behavior building blocks and may be used by other demons as well, thereby allowing for the reuse of code.

The *interaction* between the parent and its children consists of the parent controlling which children are active, and the children informing their parent of their progress. In our example: the parent demon (react-to-threat) may know that the first child that needs to be activated is the take-cover demon. This demon in turn informs the parent when it has finished taking cover. Once the parent demon is informed that cover is reached, it can de-activate the take-cover demon and activate the return-fire demon. Activating a child demon simply involves changing its shrieking level, but it does not necessarily imply that the child can and will affect behavior. The child may first need to compete with other, conflicting, demons (see section 3.3). The parent demon would usually activate a child by setting its shrieking level to its own shrieking level. Once this is done, the child has to compete autonomously with all other active demons in the pandemonium. A parent can de-activate child demons by setting their shrieking level to zero. Demons with a shrieking level of zero are still part of the pandemonium but do not take part in the competition.

3.3 Resources

In CHAOS, behavior can be triggered by *tasks*, which can be considered *goal directed behavior*, or by *stress or strain*, which can be considered *reactive behavior*. Note that "stress" should be interpreted very broadly here, ranging from exhaustion to stress caused by enemy fire. The influence of tasks and stress on behavior is depicted schematically in Figure 6, which also shows the central role *resources* play. These resources represent the *capabilities* of the system (human) that is modeled. Which capabilities are modeled depends on the specific application. Examples are attention, fine motor skills, or potential physical work rate. The demons in CHAOS are competing over these limited resources in order to produce behavior. The louder a demon shrieks (i.e. the more important he is), the bigger the chance is that he will get the resources he needs. Demons can only produce behavior if the capabilities (resources) needed for their specific type of behavior are available. This mechanism is represented by the arrow labeled 'used for' in Figure 6.

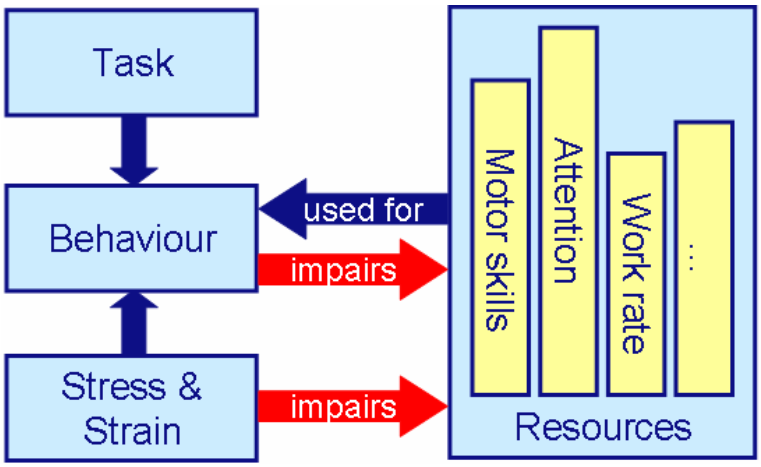

Figure 6: behavior is determined by stress, strain and/or the execution of tasks. Behavior can only be executed if the relevant capabilities ("behavior resources") are sufficiently available. Behavior and stress can both impair the capabilities of the system.

Multi-tasking

Given the role resources play in CHAOS, multi-tasking is controlled almost automatically by the fact that two or more *conflicting* behaviors (demons) will be competing over the *same* resources and can therefore not be performed simultaneously. We can also see in Figure 6 that behavior itself may *impair* the capabilities of the system (top 'impairs' arrow). This represents the fact that some behaviors that are not competing over the same resources still can not be performed simultaneously. This is another mechanism used to control multi-tasking. For instance, running will *affect* fine motor skills, even though fine motor skills are not *needed* for running. This is modeled by having any behavior involving running impair the capability "fine motor skills". Any behaviors that depend on this fine motor capability will in turn not be executed or will be executed badly.

Stress and strain

Another important aspect of the role resources play in CHAOS has to do with stress and strain. Stress or strain can not only *trigger behavior* but may also directly *affect performance*. For instance: when a person gets tired while running he may decide to take a rest (stress triggers a change in behavior) or he may continue running, but will have to reduce his speed (stress affects the performance of the current behavior). The first type of effect is represented by the arrow from "Stress & Strain" to "Behavior" in Figure 6. The second type is represented by the bottom 'impairs' arrow, where stress directly "consumes" some of the resources (i.e. impairs some of the capabilities) of the system without producing any behavior in return.

3.4 Pandemonium

So far we have covered the resources, or capabilities of the system, and the demons that are responsible for behavior and that can affect resource levels. However, a demon is not allowed to control the behavior or influence the resource levels of the entity completely autonomously but it needs "permission" to do so by the pandemonium. In fact, every demon is

part of (contained in) the pandemonium, and the pandemonium controls when demons are updated and which demons get to influence the entity's behavior and resource levels. This process is shown schematically in Figure 7.

The procedure starts with resetting the capabilities (resources) to their initial levels. Next, all demons are ordered to update their "shrieking levels", given the current state of the simulation. After that, all demons representing some form of stress or strain are allowed to impair the capabilities. Finally all demons are requested to perform actions, but due to the limited resources only a small amount of demons (possibly just one or even none) will actually take some actions/produce behavior. In this last step the demon that is shrieking loudest will be the first to be asked to execute his actions. In order to do so, he will probably use some of the resources and perhaps impair others, leaving fewer resources for the next demon in line, etc.

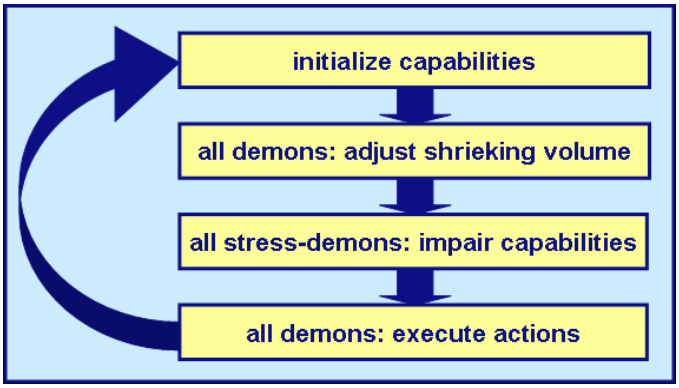

Figure 7: flow of control in the pandemonium. Iterating over demons is done on a sorted list, starting with the loudest shrieking demon.

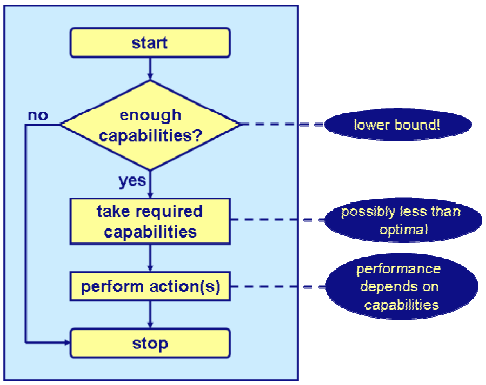

Figure 8: The procedure a demon follows when performing actions.

Figure 8 zooms in on the procedure a demon follows to produce behavior. A demon will start by checking if enough capabilities (resources) are available. If this is not the case it will return without taking actions or using resources, leaving the turn to the next demon in line. If enough (but possibly less than optimal) resources are available, the demon will take these resources and perform his actions. In case the resource levels were indeed less then optimal, the demon may alter his performance (whether this is the case depends on the specific implementation of the demon). This is how for instance stress may lead to a drop in performance: stress-demons take some of the resources before any behavior is "produced", this then leads to less-than-optimal resource levels, resulting in less-than-optimal performance.

So simply by *sorting* the demons according to their shrieking levels, the pandemonium can –without knowing anything about the specific behaviors or stressors that are active– have control over all the demons it contains. The fact that the demons require and make use of the limited set of resources assures that, if two or more behaviors are conflicting, only the most important one(s) will be performed. Neither the pandemonium nor the demons are fully responsible for dealing with exceptions, such as reacting to enemy fire while patrolling. Still, the system as a whole is capable of controlling when an exception should alter the "normal" (goal directed) behavior and when it should not. This principle is what makes the CHAOS framework suitable for complex behavior modeling.

3.5 Comparing CHAOS to COGNET

On the surface CHAOS and COGNET [6] seem similar since they both employ Selfridge's concept of shrieking demons. However, there is a number of differences between the two systems. Some of these differences have to do with the fact that COGNET is partially based on a functional decomposition. For instance, all information regarding the current situation is stored on a central blackboard as *declarative knowledge* [6]. This is the only source of information available to tasks, which are equivalent to our demons. Although such a concept of a blackboard that stores information may be adopted by a system that incorporates CHAOS (for instance, SCOPE contains a central component that represents some aspects of situation awareness), in CHAOS demons are also allowed to get their information from elsewhere. For instance, a demon representing "thermal stress" would probably get his information *directly* from a model of the human thermal system. If this were not the case the thermal system would have to post information about its variables on a central blackboard. This is a rather roundabout approach for the non-cognitive aspects of human modeling. Another difference related to the functional decomposition underlying COGNET is the fact that, even though tasks contain procedural knowledge, the actual "Action Knowledge" [6] needed to perform actions is separated from the tasks.

Perhaps the most important difference between the two systems is that COGNET does not support multi-tasking or performance degradation due to stress or strain. Since COGNET contains only a single resource ("attention"), and since all tasks require "attention", all tasks are considered conflicting. The result is that only one task can be active at any time. The same logic holds for stress or strain: since there is only one resource that is completely used by the active task, there is no way of implementing demons that can influence performance by impairing some of the capabilities of the system.

4. SCOPE

TNO has, as a partner of the Netherlands Ministry of Defense, developed the SCOPE simulation tool for operational analysis of dismounted soldier operations. A key characteristic of SCOPE is the level of human factors related modeling. These human factors models range from physiology to subjective threat perception and from visual perception to the CHAOS behavior model. Another key characteristic is that scenarios are executed in a single run instead of in a Monte Carlo fashion. This is possible since SCOPE will realize the "average outcome" in a single run. This average outcome can be achieved by virtue of abstractions made in the terrain representation and the probabilistic nature of some of the SCOPE models. For instance, it is possible to have a probability of hiding behind a tree (instead of being behind the tree or not). An entity is hit with a given probability, resulting in a decrease of the entity group size by the same, *fractional*, amount. This will in turn result in a proportional, fractional decrease of some of the group's capabilities. The single run feature also allows SCOPE to be used with much more complex scenarios than would be possible with traditional Monte Carlo simulations. Scenarios with multiple engagements that take several hours to complete are not a problem in SCOPE. Single run simulation also greatly simplifies analysis and understanding, since no statistical analysis is needed afterwards.

SCOPE contains many models, ranging from a radio propagation model to a model of visual perception. It is beyond the scope of this paper to give an extensive overview of all these models. However, we will briefly describe some models that are of special importance to the experiments described below, namely the models related to physiology and the SCOPE weapon effects model.

4.1 Physiology

As stated above, one of the key characteristics of SCOPE is the level of human factors related modeling. The physiological aspects of human performance are represented with models of the human thermal and metabolic systems. The thermal model is a two-node model that splits the body in a core and skin compartment, and a clothing layer that serves as the interface with the environment. The clothing ensemble is represented as a single layer, covered with the adjacent air layer. This allows us to include thermal and vapor resistance of the clothing and to include solar radiation. The combined physiological and clothing model is a simplified version of the model described and validated by Lotens [13]. The effects of acclimatization and fitness on sweating and blood flow are also taken into account, based on Havenith [14].

The metabolic model is based on textbook knowledge (see for instance [15]). It models glucose and lactate consumption and production, but ignores short term metabolic effects, making the model useful for tasks ranging in duration from 1 minute up to many hours. Factors influencing the glucose and lactate levels are the energy demand of the task, the fitness of the subject and the food regime. Fitness can range from 0 (unfit) to 1 (very fit) and affects the maximal aerobic work rate and percentage of anaerobic work. Fitness will also increase the muscle glycogen storage and the tolerable lactate concentration in the muscles.

Performance effects related to metabolism and the thermal system are modeled within the CHAOS framework. For the thermal model three demons are implemented, representing heat stress, dehydration and shivering. For the metabolic model glucose depletion is modeled by two demons (glucose concentration in blood and muscles) and lactate concentration is represented by a single demon. These demons are modeled analogously to the examples given in Figure 4 and can affect performance by impairing the capability "work rate". This capability represents the potential physical work rate of the soldier, and can thus for instance affect walking speed. The connection between walking speed and work rate is based on the model by Pandolf [16], with an extension based on the work of Minetti et al. [17]. Other influences taken into account, according to the Pandolf model, are terrain type, slope and load carried. The resulting work rate is fed into the thermal and metabolic models and will thereby affect the physiological condition. A controlling feedback mechanism exists since increase in physiological stress reduces the work rate, which will reduce the stress levels.

4.2 Weapon effects and ballistic protection

Two types of weapons effects (powers) are modeled in SCOPE: direct and indirect. Direct powers must be aimed (directed) at their target (e.g. a rifle) while an indirect power delivers a threat in a circular area surrounding the detonation point (e.g. a hand grenade). Indirect powers can be fragmenting, doing damage by spreading fragments over the impact range; non-fragmenting indirect powers deliver a continuous impact. This classification accommodates virtually any lethal and many non-lethal weapon.

The representation of ballistic protection in SCOPE is based on the four main protection classes as defined in the NIJ Standard 0101.04 [18]. This ballistic protection classification assumes that bullets hit at their maximum speed. This is of course not always the case; the distance from which the shot is fired determines the speed at impact. This means that the distance is a parameter in determining the effect of protection. In the model this is handled by determining the maximum effective range of the weapon for each protection class, which is shorter for heavier protection. When a person in SCOPE is wearing ballistic protection, only a percentage of his body surface will be protected. The weapon effects model uses this percentage to determine the amount of ballistic damage on the whole surface.

5. EXPERIMENTS

Recently, TNO has performed a study with SCOPE on the effects on operational performance of different types of ballistic protection, fitness and acclimatization. A typical peacekeeping scenario was used, involving several engagements and displacements. Two different weather conditions represented the Afghan and Dutch climates. The CHAOS framework facilitated the human behavior and performance modeling that was needed for this study.

Four clothing and protection configurations were compared (see Table 1). These configurations have a twofold effect on operational performance. Firstly, the level of ballistic protection reduces the probability of incapacitation as calculated by the weapon effects model. Secondly, the weight associated with each configuration will affect the physiological condition of the person, possibly resulting in stress and strain. Even without stress or strain, additional weight will reduce walking speed. The fitness level was also varied in the study and affects some physiological variables, as is explained in section 4.1. The level of acclimatization was varied as well, affecting the probability of heat stress in hot climates. An overview of all conditions is given in Table 2.

G1	Helmet	2.0 kg.				
	Ammunition	2.5 kg.				
	Clothing + Shoes	3.0 kg.	7.5 kg.			
G2	Vest	5.0 kg.		12.5 kg.		
G3	Ball. Plates	7.0 kg.			19.5 kg.	
G4	Marching Kit	15.0 kg.				34.5 kg.

Table 1: The four clothing and protection conditions. Each configuration adds to the previous configuration. So G2 = G1 + a ballistic vest. G4 adds no protection, just weight by introducing a 15 kg backpack.

Category	Condition	Description
Clothing	G1	Basis
Protection	G2	Vest
Marching kit	G3	Plates
	G4	Marching kit
Fitness	0.50	Normal
	0.75	Fit
	1.00	Very fit/Athlete
Climate	AFG	Afghanistan
	NL	Netherlands
Acclimatisation	2	Bad
	30	Good

Table 2: All conditions: clothing and equipment, fitness, climate, and acclimatization (in days).

5.1 Results

The results showed that the experimental conditions influenced performance according to expectation. Acclimatization improved physical performance in the hot Afghan climate. In the worst case scenario (G4, fitness: 0.5), acclimatization reduced the time used to complete the scenario from 04h:11m to 04h:01m. Fitness could improve performance by another ten minutes in the Dutch climate. In the hot climate however, being well trained was less important since the physiological bottleneck was of thermal rather than energetic nature. The effects of ballistic protection were also as expected with increased protection reducing casualties, but negatively affecting other operational measures (time needed to complete the mission, post-action readiness). The time to complete the scenario ranged from two hours and fifty minutes (G1, Dutch climate, very fit), to well over four hours (G4, Afghan climate, unfit, not acclimatized).

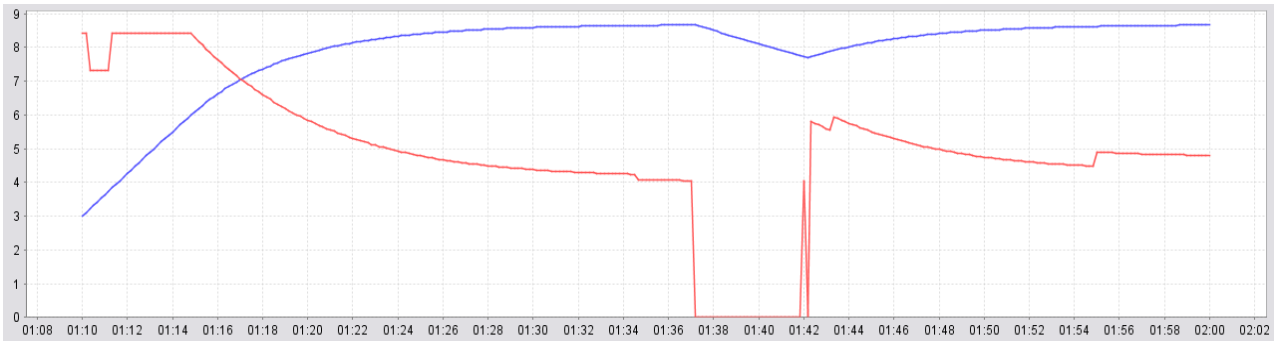

Figure 9: typical example of performance degradation, taken from the scenario described above. The blue (top) series shows the "body heat content", which is considered stressful between a value of 5 and 10 (see Figure 4). This form of thermal stress affects performance, in this case by limiting the walking speed as represented by the red (bottom) series (km/h). The x-axis shows simulation time.

The role of CHAOS in this experiment is twofold; firstly it provided the means for (low level) autonomous decision making. Secondly, CHAOS facilitated the modeling of performance degradation. All differences in measures of effectiveness (number of casualties, time to complete the scenario, post-action readiness) resulting from the different conditions are related to the performance degradation mechanisms as implemented in CHAOS. A typical example of performance degradation is shown in Figure 9, where the effect of thermal stress on walking speed is displayed. As stress rises (01:10-01:37), walking speed decreases smoothly (thermal stress impairs the physical work rate capability which is needed for all physical activity). When the demon that represents thermal stress starts to shriek louder than the demon that represents the movement, the entity will stop walking altogether to rest (01:37). After a period of resting (the duration of the recovery period is determined dynamically; recovery will stop when stress has dropped with a predetermined percentage), the entity will resume walking (01:42).

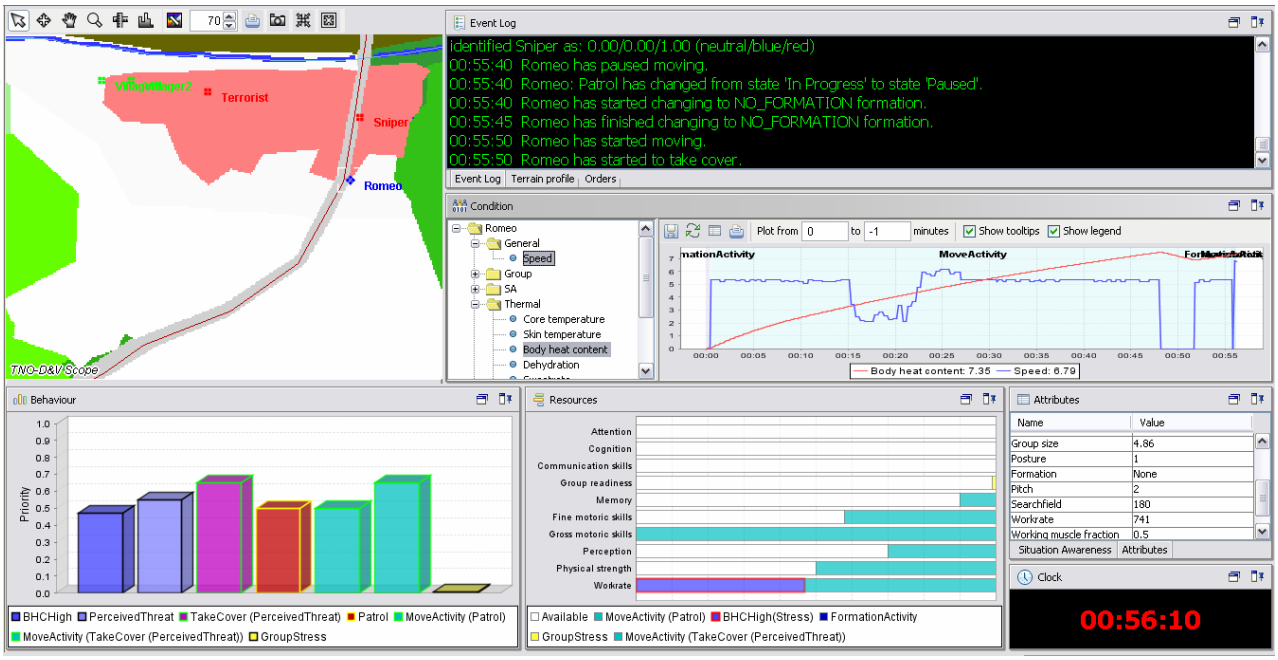

Figure 10: Screenshot of the SCOPE simulation environment. The left top panel shows a 2D map, with each color representing a specific terrain type. The right top panel shows an event log. The other panels display information regarding the selected entity, Romeo. The graphs below the event log provide a view on some of the SCOPE models. The leftmost bottom panel shows the demons currently active, with the height of the bar charts representing the demons' shrieking levels. The panel in the middle displays the resources in CHAOS, with colors corresponding to the colors of the demons that are using resources and white representing available resources. The rightmost panels show some attributes of Romeo, and a clock that displays simulation time.

An example of the autonomous decision making process is illustrated in Figure 10, that shows a number of demons competing to control behavior. Each bar in the bottom left bar chart represents an activated demon, with the height of the bars representing shrieking level. Just before this screen was captured, the blue entity (Romeo), which represents a group of 5 soldiers, was patrolling along the gray road (top left panel in Figure 10). When Romeo is fired at by the Sniper, located in the village (red area), Romeo decides to suspend the patrol task and takes cover. The mechanism behind this decision is as follows: the sniper firing leads to increased perceived threat (represented by the second demon from the left in Figure 10), making it more important than the current patrol task (fourth from the left). The 'PerceivedThreat' demon takes over by activating its 'TakeCover' child demon (third from the left). This demon decides on the best cover location and determines a route towards this location. It then initializes and activates a 'MoveActivity' (second from the right) that is responsible for moving to the cover location.

5.2 Discussion

The quality and validity of the behavior produced with CHAOS depends largely on the actual implementation of the demons and capabilities. The CHAOS system will only function correctly if the relevant capabilities are identified, implemented and utilized. Furthermore, the demons should be capable of determining their own importance (shrieking level). Meeting these modeling requirements can be hard, but similar requirements will have to be met in *any* architecture for human behavior modeling. Once these requirements are met, the CHAOS system will facilitate the dynamic combination of stress, strain and goal directed behaviors.

The multi-dimensional nature of operational simulations, including the described SCOPE experiment, complicates validation, but also shows the added value of simulation. Although a formal, predictive validation of SCOPE or CHAOS is not feasible due to the complexity involved, it is possible to validate the subcomponents and underlying mechanisms of a complex model such as SCOPE. The physiological models in SCOPE are based on existing, validated models and research. These models play an important part in the experiment described in this paper. The CHAOS stress mechanism forms the interface between low-level (physiological) models and performance and behavior. In the SCOPE implementation, this stress mechanism facilitates, at the very least, *content validity,* by preventing unrealistic performance or unrealistic physiological model values. This claim is backed by the experimental data that were all within realistic ranges, while showing the significant impact additional weight can have on performance, especially in hot climates. All variations in the examined variables resulted in expected changes in operational performance and are considered realistic by subject matter experts.

6. CONCLUSIONS

CHAOS is currently part of the SCOPE (dismounted soldier simulation) and SAFE (simulation of firefighter operations) projects. For these projects, CHAOS has shown itself to be a suitable framework for human behavior representation and human performance modeling. The decentralized design, based on a behavioral rather than functional decomposition, simplifies the modeling process. By incorporating multiple resources, CHAOS allows for modeling of multi-tasking and stress and strain.

Although CHAOS is just a framework, the study performed in SCOPE has shown that, provided with the proper implementation, CHAOS is capable of producing autonomous behavior in dynamic environments. Within the study several forms of stress and strain resulted in performance degradation and alternative behaviors. The conclusion is therefore that CHAOS provides a useful methodology for modeling behavior and performance of embodied and situated entities.

REFERENCES

[1] Selfridge, O.G., "Pandemonium: a Paradigm for Learning", Proc. Symposium on Mechanization of Thought Processes: National Physical Laboratory. London: HM Stationery Office, 1959.

[2] Pew, R.W., Mavor, A. S. (editors), "Modeling Human and Organizational Behavior: Application to Military Simulations", National Academy Press, Washington, D.C. 1998.

[3] Ritter, F. E., Shadbolt, N. R., Elliman, D., Young, R., Gobet, F., and Baxter, G. D., "Techniques for modeling human and organizational behaviour in synthetic environments: A supplementary review.", Human Systems Information Analysis Center, Wright-Patterson Air Force Base, OH, 2003.

[4] Anderson, J. R., Bothell, D., Byrne, M. D., Douglass, S., Lebiere, S. & Qin, Y., "An integrated theory of the mind", Psychological Review, 111(4). 1036-1060 (2004).
[5] Lehman, J.F., Laird, J., Rosenbloom, P., "A Gentle Introduction to SOAR, an Architecture for Human Cognition: 2006 Update", http://ai.eecs.umich.edu/soar/sitemaker/docs/misc/GentleIntroduction-2006.pdf
[6] Zachary, W., Ryder, J.M., Hicinbothom, J.H, "Cognitive task analysis and modeling of decision making in complex environments". In Cannon-Bowers, J. A., & Salas, E. (Eds.), "Making decisions under stress: Implications for individual and team training" (pp. 315–344). Washington, DC: American Psychological Association (1998).
[7] Brooks, R.A, "Intelligence without Reason", Proceedings of the International Joint Conference on Artificial Intelligence, pp. 569-595, 1991.
[8] Wooldridge, M., "Reasoning About Rational Agents", MIT Press, 21-46 (2000).
[9] Howden, N., Röhnnquist, R., Hodgson, A., Lucas, A., "JACK Intelligent Agents™: Summary of an Agent Infrastructure", Workshop on Infrastructure for Agents, MAS, and scalable MAS at The Fifth International Conference on Autonomous Agents, Montreal, Canada, 2001.
[10] "IWARS Methodology Guide", July 2006, AMSAA
[11] Evertsz, R., Ronnquist, R., "Agent-Oriented Control of Battlefield Simulations", SimTecT conference, Sydney, Australia, 1998.
[12] Rönnquist, R., Lucas, A., Howden, N., "The Simulation Agent Infrastructure (SAI) – Incorporating Intelligent Agents into the CAEN Close Action Simulator", SimTecT conference, Sydney, Australia, 2000.
[13] Lotens, W.A., "Heat transfer from humans wearing clothing". Thesis Delft Univ of Technology, 1993.
[14] Havenith, G., "Individual heat stress response". Thesis, Univ Nijmegen, ISBN 90-9010979-X, 1997.
[15] McArdle, W.D., Katch, F.I and Katch, V.L., "Exercise physiology: Energy, Nutrition and Human Perfromance.", Lappingcott Williams & Wilkins, 2001
[16] Pandolf, K.B., Givoni, B, & Goldman, R.F., "Predicting energy expenditure with loads while standing or walking very slowly". Journal of Applied Physiology, 43 (4) 577-581 (1977).
[17] Minetti, A.E., Moia, C., Roi, G.S., Susta, D., Ferretti, G., "Energy cost of walking and running at extreme uphill and downhill slopes", Journal of Applied Physiology, 93 (3), 1039-1046 (2002).
[18] http://en.wikipedia.org/wiki/Bulletproof_vest#Performance_standards.