

Invited paper at the 2006 Workshop on Engineering Self-Organizing Agents, Hakodate, Japan.

## Hybrid Multi-Agent Systems: Integrating Swarming and BDI Agents

H. Van Dyke Parunak<sup>1</sup>, Paul Nielsen<sup>2</sup>, Sven Brueckner<sup>1</sup>, Rafael Alonso<sup>3</sup>

<sup>1</sup>NewVectors LLC, 3520 Green Court Suite 250, Ann Arbor, MI 48105  
{van.parunak, sven.brueckner}@newvectors.net

<sup>2</sup>28095 Hawberry Rd, Farmington Hills, MI, 48331  
paul\_eric\_nielsen@yahoo.com

<sup>3</sup>SET Corporation, 1005 North Glebe Road, 4th Floor, Arlington, VA 22201  
ralonso@setcorp.com

**Abstract.** The individual agents that interact in a multi-agent system typically exist along a continuum ranging from heavyweight cognitive agents (often of the “BDI” type) to lightweight agents with limited individual processing (digital ants). Most systems use agents from a single position along this spectrum. We have successfully implemented several systems in which agents of very different degrees of internal sophistication interact with one another. Based on this experience, we identify several different ways in which agents of different kinds can be integrated in a single system, and offer observations and lessons from our experiences.

### 1. INTRODUCTION

It has been said that to a small boy with a hammer, every problem looks like a nail. Technologists often seek to press every problem into the mold of their favorite mechanism. In the domain of multi-agent systems, a wide range of agent models have been developed. Some are highly sophisticated cognitive agents that aspire to individual human level intelligence, while other emulate insect-level cognition and exhibit intelligence only at the level of collective behavior.

For several years, we have been exploring different ways of combining heterogeneous models of cognition in a single system. Our experiences show that this approach is not only possible, but that it yields benefits that would be difficult to obtain within a homogeneous framework.

Just as there is no single best agent model, there is no single best way to combine different models. We exhibit a number of different architectures and discuss the situations in which they can be most profitably applied.

Section 2 describes the range of agent models that we hybridize in our work. Section 3 surveys and illustrates the different modes of integration that we have explored. Section 4 offers discussion and conclusion.

## 2. ALTERNATIVE AGENT MODELS

Software agents exist across a range of complexities. At some risk of oversimplification, we describe two extremes, and then illustrate some points in the middle. For expository purposes, we call the two extremes “heavyweight” and “lightweight” agents, but these titles are more mnemonic than definitive. Table 1 summarizes the differences between these two extremes.

### 2.1 Heavyweight Agents

Heavyweight agents are based on the cognitively-inspired AI programs developed in the heyday of artificial intelligence. Each such program then, and each individual agent now, aspires to human-level intelligence. In this domain, an intelligent agent system consists of a system composed of individually intelligent agents. Well-known researchers in the heavyweight tradition include Durfee [7], Jennings [16], Laird [19], Lesser [21], Sycara [34], and Wooldridge [37].

Heavyweight agents have inherited classical AI’s emphasis on symbolic representations manipulated with some form of formal logic. The symbols are intended to represent cognitive constructs that are meaningful to people, such as beliefs, desires, and intentions (thus the common rubric “BDI agent” [13, 30]). These constructs, and the logical entailments among them, are elicited by a process of knowledge engineering. This process seeks to capture human intuitions about the appropriate partitioning of a problem domain and self-reflective models of how people reason about the domain.

Because heavyweight agents are built around human-inspired cognitive constructs, they facilitate communication with their users. If an agent has a concept of “danger” that corresponds to a human’s concept, there is a good chance that when the agent tells the human that a situation is dangerous, the human will understand.

This benefit comes at a cost. The process of knowledge engineering is intensive and time-consuming. In addition, logical computation is subject to a number of

**Table 1: Two Extreme Types of Software Agents**

Class of Agent	Heavyweight	Lightweight
Origins	Artificial Intelligence/Cognitive Science	Artificial Life
Locus of intelligence	Within a single agent	In the interactions among agents
Internal representations and processing	Symbolic	Numeric: polynomials, neural networks, matrix manipulations
Concepts represented	Explicit beliefs, desires, intentions/goals, plans	Sensor states, actuator levels
Development approach	Knowledge engineering	Optimization
Strengths	Intelligible to humans	Computationally efficient Degrades gracefully
Weaknesses	Computationally intractable for large problems Brittle	Difficult to understand

limitations. For example, logical computations are often

- Intractable, their computational complexity increasing exponentially or worse in the size of the problem, so that problems of realistic size cannot be executed fast enough for the answer to be useful [10],
- Undecidable, so that some questions expressible in the logic simply cannot be answered [11], or
- Brittle, with performance that degrades rapidly (either in accuracy or speed) as one nears the limits of the domain.

## 2.2 Lightweight Agents

At the other extreme, lightweight agents draw their inspiration from computerized work in ethology, the study of animal behavior. Biologists often construct computer models of animals in order to study their interactions with one another. In many cases (such as ants), no one imagines that the individual agent has anything like human-level intelligence, but the society as a whole can exhibit impressive behavior that might be described as intelligent. In this context, an intelligent agent system is a system of agents that is collectively intelligent. Well-known researchers in this tradition include Bonabeau [3], Brueckner [4], Ferber [8], Ilachinski [15] and Parunak [22].

Lightweight agents do not rely on cognitively meaningful internal representations, for two reasons. First, biologists tend to resist anthropomorphizing the mental behavior of ants and termites. Second, even if it were appropriate to describe their mental operations in the same terms that emerge from human introspection, we would have no way to interrogate the organism about these constructs. What is accessible to the biologist is the entity's environment and its observed actions, so the representation tends to focus on sensory inputs and signals sent to actuators. These are customarily described in analog terms, leading to widespread use of numerical reasoning, usually as some form of matrix algebra (a framework that includes weighted polynomials and neural networks).

Programming such an agent is a matter of identifying the appropriate numerical parameters and setting their values. Knowledge engineering is of little use with a digital insect. Instead, one uses optimization methods such as evolutionary computation to explore the parameter space. We can compare the observed behavior of the agent either with the observed behavior of the domain entity (in a modeling application) or with the desired behavior (in a control application), and use the difference between the two as an objective function [32].

Because their internal processes are essentially numerical, lightweight agents are usually more computationally efficient than heavyweight agents, avoiding issues of tractability and decidability. Their representations extrapolate naturally, avoiding the challenge of brittleness. But they can be difficult for users to understand, for two reasons.

1. The mapping from internal numerical parameters to cognitively meaningful constructs may not be direct. An agent's behavior may be dominated by the fact that the weight between two nodes in a neural network is 0.375, but that knowledge is of little use to a human seeking to validate the agent.

2. Lightweight agents often yield useful behavior, not as individuals, but as a collective. The dynamic of emergence, by which global behavior arises from individual behaviors, is often counter-intuitive [31].

### 2.3 Intermediate Agents

The two categories of “heavyweight” and “lightweight” agents as described above are extreme cases, and a number of intermediate architectures have been used.

**Scripted agents** use a state machine to shift from one cognitively meaningful state to another, based on external stimuli. Thus they avoid some of the computational complexity issues associated with richer computational models such as theorem proving.

One mechanism for scripted agents is the Task Frame [5]. Task Frames are used in military simulations such as OneSAF, JSAF, and ModSAF to decompose tasks, organize information, and sequence through the steps of a process. Finite state machines are used to sequence through the task states, however the code within these states is unrestricted.

Sometimes scripted transitions are combined with lighter-weight mechanisms. MANA [20] is a combat model whose agents make decisions based on matrix multiplications, along the line of EINSTEIN [15]. However, the personality vector that weights the effect of environmental stimuli can be changed discontinuously by certain distinguished events, allowing the agent to change among different behavior patterns depending on environmental stimuli.

**Bayes networks** [29] combine symbolic and numeric processing, in a manner similar to iconic neural networks. Each node corresponds to a concept or proposition that is meaningful to a human, in a manner consistent with symbolic representations, but the links among the nodes represent conditional probabilities, and processing consists of numeric computations of the propagation of evidence through the network.

Bayes networks have proven most useful at interpretation of activity from observations. For example, seeing a person with wet hair enter the office could either imply that it is raining or they have just taken a shower. However, if we observe several people with wet hair the belief that it is raining would increase.

These intermediate agent architectures combine in a single agent mechanisms from different points in the spectrum *in a single agent*.

## 3. INTEGRATION MODES

In this section, we discuss why it is difficult to integrate agents with different cognitive levels in a single system, and then exhibit a number of different approaches that we have explored. Our list of examples is open-ended, and we invite other researchers to expand it on the basis of their experience.

### 3.1 Why is Integration Difficult?

The hybrid systems that we discuss here differ from the “intermediate agents” discussed in Section 2.3. Those examples combined mechanisms from different points in the spectrum *in a single agent*. Here, we explore patterns for combining *distinct agents* that differ in their cognitive mechanisms.

There are three challenges in developing a hybrid system: issues internal to individual agents, issues relating an individual agent to its external environment, and issues dealing with the overall structure of the system.

**Internal Issues:** Any agent, however simple or complex, is responsible to perceive its environment and take some action based on that perception. Each agent in the system must have the capacity to solve the problem with which it is tasked.

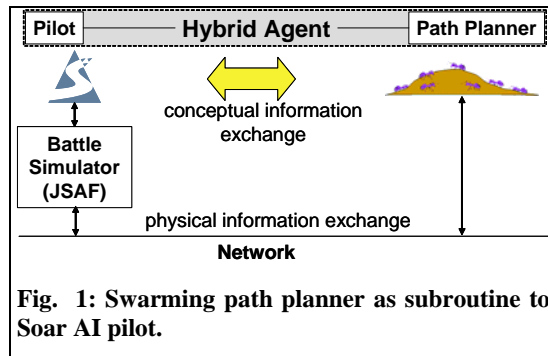
**External Issues:** The widespread use of heavyweight agents leads naturally to agent interactions that draw on the cognitive constructs that the individual agents are presumed to support. This assumption is the basis of messaging standards such as KQML/KIF and FIPA ACL. Lightweight agents interact through their sensors and actuators rather than through messages with explicit cognitive content. Thus communication between the two types of agents requires special attention.

**System Issues:** When we provide our small boy with a screwdriver and a wrench in addition to his hammer, we have made his life much more complicated. Now he has to decide which tool to use in which situation. When we permit the use of multiple levels of agent cognition in a single system, we need to think about which kind of agent to use where, and why.

### 3.2 Swarm as Subroutine

Sometimes a community of lightweight agents can perform a specialized task in support of a heavyweight agent. We used this approach in an experimental extension of TacAir-Soar [17]. The basic TacAir-Soar system is a classic heavyweight architecture based on the Soar architecture for general AI. Geospatial reasoning such as path planning is cumbersome in such an architecture [9], but straightforward for a swarm of lightweight agents. Biological ants use a simple pheromone mechanisms to generate minimal spanning trees that connect their nests with food sources [12], and these mechanisms have been applied successfully in robotic path planning to approach targets while avoiding threats [33].

We merged these two classes of agents by having the Soar agent invoke a swarm of lightweight agents to plan paths. Fig. 1 shows the structure of the implemented system. Communication between the agents is at the cognitive level required by the pilot



agent. A wrapper around the path planning swarm handles the translation. In a typical dialog, the pilot reports its current location and its destination, and requests a route. The wrapper instantiates a nest of agents at the current location, a food source at the destination, and turns the swarming agents loose. They generate a field of digital pheromones whose crest indicates the desired path. The wrapper then translates the turning points in this path into a series of waypoints, which it reports to the pilot.

### **3.3 Polyagent**

Closely related to the previous example is the use of multiple lightweight agents to explore alternatives being considered by the heavyweight agent. We have formalized this construct as the polyagent [23], and other researchers have developed similar mechanisms under the rubric of “delegate Multi-Agent Systems” (because the heavyweight agent delegates its task to the lightweight agents) [14, 18].

The justification for using lightweight agents to explore alternatives is computational efficiency. If there were no constraint on computational resources, the heavyweight agent could just explore the alternatives itself (perhaps through parallel invocations of its environment). By using lightweight agents, it can explore more alternatives in shorter time.

The difference between the polyagent and the swarm as subroutine is that the heavyweight agent in the polyagent (the “avatar”) considers each of its lightweight agents (“ghosts”) individually in making use of their results, rather than simply consuming their aggregate output as in the swarming subroutine. As a result, instead of wrapping a translator around the lightweight agents to enable them to speak the language of the heavyweight agent, we require the heavyweight agent to examine the environmental changes (e.g., digital pheromones) produced by the actions of the lightweight agents.

### **3.4 Transitional Agents**

Transitional agents morph between heavy and light weight agents depending on the needs of the application.

For example, in a large scale simulation it is infeasible to simulate every entity at a very high level of fidelity, yet humans who participate in the simulation will not be challenged by the actions of an insect level intelligence. To provide both the desired breath of scale and richness of interaction, the majority of the entities can be controlled by lightweight agents, while those few agents who come into contact with humans can be controlled by heavyweight agents.

The simplest interface simply notes that an interaction is imminent, destroys the lightweight controlled entity and replaces it with a heavyweight controlled entity, or replaces a heavyweight entity with a lightweight when the entity is no longer within human interaction range.

One problem with this approach can be exemplified as “waking up in the middle of a dogfight.” The agent needs to acclimate itself to the situation, obtain historical information from the other controller, and plan a strategy for the encounter. During

this time the agent is helpless. To overcome this transition interval, an entity's new control method should be started while the entity is under control of the previous method. In order to facilitate the transition, each controller must incur an overhead of logging additional information that is useful to the other controller.

### 3.5 Swarming Integration of Cognitive Reasoners

Sometimes it is desirable to integrate the results of multiple reasoning engines operating on a single domain. Within the paradigm of symbolic AI, the standard approach is to construct yet another reasoner that does meta-level reasoning over the knowledge produced by each of the component reasoners. This process is liable to the same challenges of tractability, decidability, and brittleness as any symbolic process. It is often exacerbated by the increased size of the problem (involving the union of the results of multiple reasoners) and the difficulty of ensuring semantic compatibility across the reasoners.

Instead of *reasoning* about the intermediate results, an alternative approach is to treat them as *constraints* that are imposed on an emulation of the domain. This approach uses the semantics of the emulation as the interface standard across the different reasoners. Whether the emulation is tractable or not depends on its underlying technology. An equation-based model of the domain is completely tractable and decidable, but has a number of shortcomings compared with an agent-based model [28]. If the agents in an agent-based model are heavyweight agents, the emulation itself may be no more efficient than metalevel reasoning. The highly efficient execution of lightweight agents enables us to run an emulation fast enough to support real-time fusion of results from multiple reasoners. This approach is in some sense a mirror image of the "Swarm as Subroutine" model of Section 3.2. There, a swarm of lightweight agents solved a specific problems for a heavyweight agent. Here, the heavyweight agents are the focused problem-solvers, and it is the swarm that integrates their results.

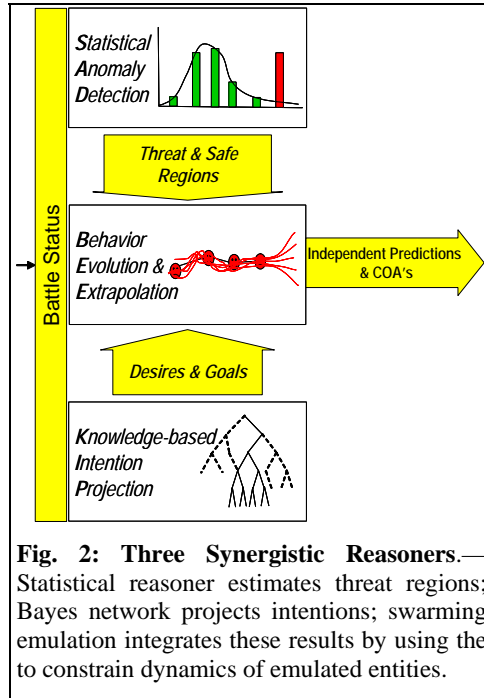
An important challenge to this approach is the question of fidelity. Whether a model can accurately fuse constraints imposed by component reasoners depends on how faithfully it represents the domain, and *a priori* one might expect lightweight agents, with their simplified internal modeling, to yield less accurate models of behavior than heavyweight agents. In many domains, the constraints imposed by the environment are more important to determining the system-level outcome of a model than the internal reasoning of the agents, a phenomenon that we have termed "universality" [25]. In such domains, lightweight agents are equal to the task.

We have taken this approach in fusing multiple reasoners in a system for predicting battlefield behavior. Fig. 2 shows the structure of this system. The central element, Behavioral Evolution and Extrapolation (BEE), is a swarming model of the battlespace that itself predicts entity behavior using polyagents (Section 3.3). It also fuses the results of two other reasoners. Statistical Anomaly Detection (SAD) identifies regions of the battlespace that are likely to be perceived as threatening, and thus repulsive, to domain entities, while Knowledge-based Intention Projection (KIP) uses Bayesian networks to suggest entity goals, which are attractors. Fig. 3 shows how the swarming BEE ghosts form predictions of entity movement that satisfy these constraints. This prediction technology outperforms human experts, and also other technical approaches (such as game-theoretic predictors) with which it has been compared [24].

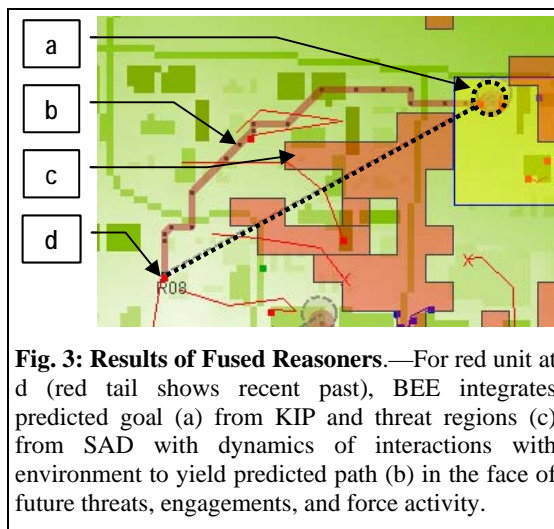
In the example shown in Fig. 2, SAD and KIP interface with the swarming simulation by depositing pheromones reflecting their results. In principle, they could also modulate the simulation by changing the structure of its underlying topology, or by modifying the personalities of the swarming agents.

### 3.6 Cognitive Interface, Swarming Processing

The cognitive transparency of heavyweight agents makes them a natural candidate for a user interface to an underlying swarming system. In this approach, the heavyweight agent maintains



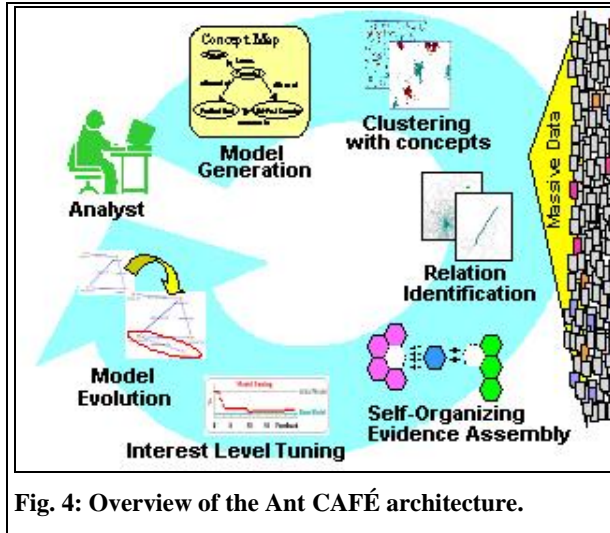
**Fig. 2: Three Synergistic Reasoners.**—Statistical reasoner estimates threat regions; Bayes network projects intentions; swarming emulation integrates these results by using the to constrain dynamics of emulated entities.



**Fig. 3: Results of Fused Reasoners.**—For red unit at d (red tail shows recent past), BEE integrates predicted goal (a) from KIP and threat regions (c) from SAD with dynamics of interactions with environment to yield predicted path (b) in the face of future threats, engagements, and force activity.

a model of the user's interests. It focuses the behavior of the swarm in accordance with those interests, and then translates the swarm's results into terms understandable to the user. Metaphorically, the heavyweight agent constructs the DNA that guides the behavior of the lightweight agents.

An example of this architecture is the Ant CAFÉ, a system that supports Indications



**Fig. 4: Overview of the Ant CAFÉ architecture.**

and Warnings analysts as they try to connect clues gleaned from massive quantities of complex data [35]. The system is an iterative loop: analysts ask the system to find evidence that supports a hypothesis, the system returns assemblies that organize relevant evidence, and the analyst reviews the evidence and in the process improves her understanding of the problem. The analyst-system interaction leads to a revised representation of the hypothesis, and the loop iterates repeatedly in this manner as the investigation advances. Fig. 4 provides a high level overview.

Hypotheses are represented as concept maps [6]. The concept maps are utilized in every stage of processing; they essentially act as templates for the construction of evidence assemblies. Concept maps are graphs with labeled nodes and edges. The nodes are nouns and the edges are verbs or verb-prepositions. We call the nouns and verbs *concepts*. We call two nodes and their connecting edge, together, a *relation*. We consider concept maps to be a low-commitment form of ontology-like symbolic knowledge representation. They are becoming quite ubiquitous for modeling domain knowledge, and are now widely taught in middle schools and elsewhere.

The left side of Fig. 4 (the Analyst Modeling Environment, or AME) involves modeling the analyst's interests as represented in the concept maps, using a heavyweight agent that explicitly manipulates high-level concepts. Issues include initial acquisition of concept maps, tuning weights associated with concepts and relations to reflect analyst interests by observing their behavior, and evolving concept maps in semi-automatic ways to capture increasing understanding as investigations progress [1]. The use of a heavyweight agent to model the analyst and interpret the results returned by the Ant Hill is necessary to enable humans to use the system effectively.

The right side of Fig. 4 is the Ant Hill, which uses lightweight agents in swarming processes to cluster data, identify relations, and assemble evidence. Each of these stages employs a distinct swarming mechanism. They are sequential in terms of logical data flow, but execute concurrently. All of the processes use *anytime*

algorithms: where some answer is available at any time, and the quality of the answer improves as time passes. Lightweight swarming agents are appropriate for the Ant Hill because they can efficiently process the massive dynamic data that the system must handle.

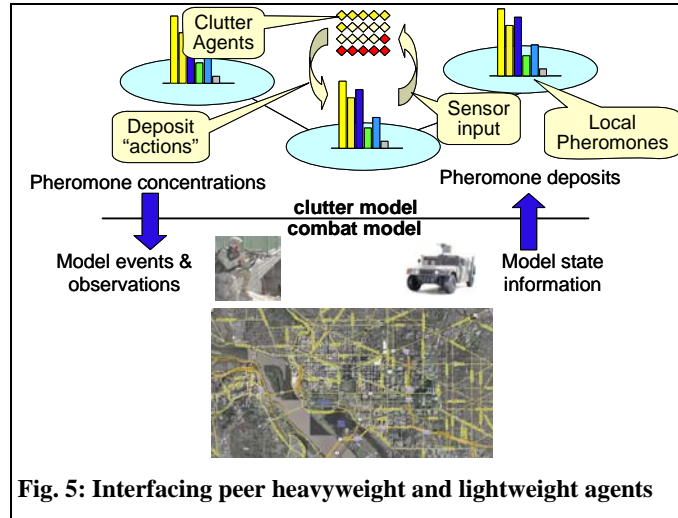
The concept maps form the interface between the Analyst Modeling Environment and the Ant Hill. The AME can interpret them as symbolic structures to capture the analyst's interests and explain the structure of results, while the Ant Hill can use the concepts and their associated weights to define a feature vector that supports low-level self-organizing processes such as dynamic distributed hierarchical clustering [27].

### **3.7 Peer Interactions**

Because lightweight agents do not require the degree of knowledge engineering that heavyweight agents do, and because of their relative efficiency of execution, they can often be attractive as a way to increase the population of a multi-agent model without unduly increasing its cost or its computational expense. In this approach, lightweight and heavyweight agents function as peers in an agent-based model.

Military operations unfold within the constraints of a particular physical environment. Operational effectiveness is a product of the joint "system" of the military operations and the environment. These emergent dynamics often vary widely as the state of the environment varies, to the point where the dynamics and thus the outcome of particular military operations may be completely dominated by the environment in which they are embedded. Recognizing this relationship between the operations and their environment is important at different levels. At the level of sensors and communication networks, local variations in environmental factors such as weather, soil conductivity, and foliage can lead to wide variation in system performance. At the level of fighting units such as urban combat with small distributed teams, FCS system of systems, or highly distributed mixed-initiative systems, the outcome depends on the availability of information and the interaction of numerous entities and the complex environment in which they are embedded.

The interaction of various noncombatants with different states within the environment of a military operation may also strongly determine the outcome of the operations. Noncombatants interact with the operations in two ways. On the one hand, they are simply affected by the ongoing operations (e.g., seeking refuge, being injured, etc.), but on the other hand, they may also influence the progress of the operation directly or indirectly. A direct impact occurs when some of the objectives of the operation are contingent on the state of the noncombatants (e.g., minimize casualties, control refugee flows). Noncombatants indirectly influence the progress of the operations as they may provide cover for combatants or obstruct the line-of-sight to targets.



**Fig. 5: Interfacing peer heavyweight and lightweight agents**

We have implemented this approach to add swarming non-combatants to the scripted combatants in an urban combat modeled in COMBAT XXI [2]. Fig. 5 shows how we interface them. The swarming clutter agents live on a graph model whose nodes maintain digital pheromone concentrations. The scripted combatants live on a map. We add a process to the computational environment that translates between pheromone concentrations on the swarming side and events and state information on the heavyweight side. Such a process is an instance of the kind of services that make it important to consider the environment a first-class object in multi-agent systems [36].

## 4. DISCUSSION AND PROSPECTUS

We can now return to the challenges to integration that we raised in Section 3.1, and also outline some directions for further research.

### 4.1 Internal and System Issues

Fig. 6 summarizes the relative strengths and complementarity of heavyweight (e.g., BDI) and lightweight (typically swarming) agents. In one way or another, each of our examples seeks to combine the strengths of the two models.

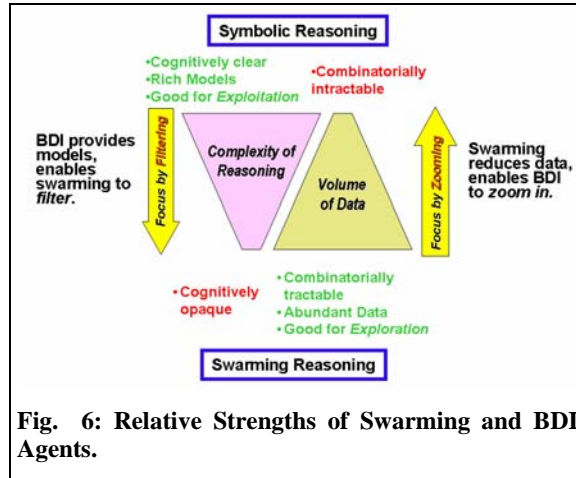
This prototypical summary of the capabilities and weaknesses of the different models is moderated by the principle of agent universality [25]. As the level of constraint imposed by the environment increases, differences in the cognitive capability of agents become less important. In effect, there is a trade-off between the information that agents can gain from the environment and the information they must generate by their own reasoning. Highly-constrained environments may provide so much exogenous information that even very simple agents are all that is needed, while

in environments with no constraints, swarming lightweight agents may not be able to function effectively.

#### 4.2 External Issues: Agent Interfacing

Our examples show a number of different approaches to agent interfacing. Lightweight agents are by their nature representationally impoverished relative to

heavyweight agents, so it is natural that heavyweight agents will need to learn to “speak their language” rather than the other way around. This may take the form of depositing and reading pheromones that are visible by stigmergic agents (as KIP and SAD do in the swarming integration example). Because lightweight agents are so simple, it is possible to have heavyweight agents custom-craft them for specific tasks (as in the Polyagent, or along the lines of the cognitive interface example).



**Fig. 6: Relative Strengths of Swarming and BDI Agents.**

#### 4.3 Future Research

Once we begin to think in terms of using lightweight and heavyweight agents together in the same system, a number of research questions become evident.

To what extent can we construct a unified development methodology that supports both extremes (and thus intermediate agent types as well)? It is clumsy to have to use one methodology [38] for BDI agents, and a completely separate one [26] for swarming agents. Hybrid systems will only become commonplace when both kinds of agents can be developed within an integrated framework.

Sometimes the cognitive complexity of an agent may need to differ during its lifetime. For example, imaging a crowd of lightweight agents representing non-combatants. At some point, an intelligence operative might want to engage one of them in conversation about what she has seen recently, a process that requires the cognitive capabilities of a heavyweight agent. One approach is the transitional agent approach described in Section 3.4. Another would be to add an interpreter to the lightweight agent along the lines of Section 3.6. Such a dynamic change in agent complexity has not been explored to any great degree, and poses a number of interesting research challenges.

The schemes summarized in this paper are the result of surveying projects that we have executed. A further stage of analysis might yield a more systematic theory of how one can integrate different agent models, which in turn might suggest further approaches to this important problem.

## 5. REFERENCES

- [1] R. Alonso and H. Li. Model-driven Information Discovery for Intelligence Analysis. In *Proceedings of In preparation*.
- [2] Army MSRR. Combined Arms Analysis Tool for the XXIst Century (COMBAT XXI). 2006. [http://www.msrr.army.mil/index.cfm?RID=MNS\\_A\\_1000185](http://www.msrr.army.mil/index.cfm?RID=MNS_A_1000185).
- [3] E. Bonabeau, M. Dorigo, and G. Theraulaz. *Swarm Intelligence: From Natural to Artificial Systems*. New York, Oxford University Press, 1999.
- [4] S. Brueckner. *Return from the Ant: Synthetic Ecosystems for Manufacturing Control*. Dr.rer.nat. Thesis at Humboldt University Berlin, Department of Computer Science, 2000. <http://dochostrz.hu-berlin.de/dissertationen/brueckner-sven-2000-06-21/PDF/Brueckner.pdf>.
- [5] A. Ceranowicz, P. E. Nielsen, and F. Koss. Behavioral Representation in JSAF. In *Proceedings of Ninth Annual Computer Generated Forces and Behavior Representation Conference*, Orlando, FL, 2000.
- [6] J. W. Coffey, R. R. Hoffman, A. J. Cañas, and K. M. Ford. A Concept Map-Based Knowledge Modeling Approach to Expert Knowledge Sharing. In *Proceedings of IASTED International Conference on Information and Knowledge Sharing*, 2002. <http://www.ihmc.us/users/acanas/Publications/IKS2002/IKS.htm>.
- [7] E. H. Durfee. *Coordination of Distributed Problem Solvers*. Boston, MA, Kluwer Academic Press, 1988.
- [8] J. Ferber and E. Jacopin. The framework of eco-problem solving. In *Decentralized Artificial Intelligence 2*, pages 181-193. Elsevier/North-Holland, Amsterdam, 1991.
- [9] K. D. Forbus, P. Nielsen, and B. Faltings. Qualitative Spatial Reasoning: The Clock Project. *Artificial Intelligence*, 51(1-3):417-471, 1991.
- [10] M. R. Garey and D. S. Johnson. *Computers and Intractability*. San Francisco, CA, W.H. Freeman, 1979.
- [11] K. Gödel. Über formal unentscheidbare Sätze der Principia Mathematica und verwandter Systeme, I. *Monatshefte für Mathematik und Physik* 38:173-198, 1931.
- [12] S. Goss, S. Aron, J. L. Deneubourg, and J. M. Pasteels. Self-organized Shortcuts in the Argentine Ant. *Naturwissenschaften*, 76:579-581, 1989.
- [13] A. Haddadi and K. Sundermeyer. Belief-Desire-Intention Agent Architectures. In G. M. P. O'Hare and N. R. Jennings, Editors, *Foundations of Distributed Artificial Intelligence*, pages 169-185. John Wiley, New York, NY, 1996.
- [14] T. Holvoet and P. Valckenaers. Exploiting the Environment for Coordinating Agent Intentions. In *Proceedings of Third International Workshop on Environments for Multi-Agent Systems (E4MAS06)*, Hakodate, Japan, Springer, 2006.
- [15] A. Ilachinski. *Artificial War: Multiagent-Based Simulation of Combat*. Singapore, World Scientific, 2004.
- [16] N. R. Jennings and J. R. Campos. Towards a Social Level Characterisation of Socially Responsible Agents. *IEE Proceedings Software Engineering*, 144(no 1):11-25, 1997.
- [17] R. M. Jones, J. E. Laird, P. E. Nielsen, K. J. Coulter, P. Kenny, and F. V. Koss. Automated intelligent pilots for combat flight simulation. *AI Magazine*, 20(1 (Winter)):27-41, 1999.
- [18] J. Kant and S. Thiriot. Modeling one Human Decision Maker with a MultiAgent System: the CODAGE Approach. In *Proceedings of Fifth International Conference on Autonomous Agents and Multi-Agent Systems (AAMAS06)*, Hakodate, Japan, ACM, 2006.
- [19] J. E. Laird, A. Newell, and P. S. Rosenbloom. Soar: An Architecture for General Intelligence. *Artificial Intelligence*, 33(1): 1-64, 1987.
- [20] M. K. Lauren and R. T. Stephen. Map-Aware Non-uniform Automata (MANA)—A New Zealand Approach to Scenario Modelling. *Journal of Battlefield Technology*, 5(1 (March)):27ff, 2002. <http://www.argospress.com/jbt/Volume5/5-1-4.htm>.

- [21] V. Lesser and D. D. Corkill. Functionally accurate, cooperative distributed systems. *IEEE Transactions on Systems, Man, and Cybernetics*, SMC-11:81-96, 1981.
- [22] H. V. D. Parunak. 'Go to the Ant': Engineering Principles from Natural Agent Systems. *Annals of Operations Research*, 75:69-101, 1997. <http://www.newvectors.net/staff/parunakv/gotoant.pdf>.
- [23] H. V. D. Parunak and S. Brueckner. Modeling Uncertain Domains with Polyagents. In *Proceedings of International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS'06)*, Hakodate, Japan, ACM, 2006. <http://www.newvectors.net/staff/parunakv/AAMAS06Polyagents.pdf>.
- [24] H. V. D. Parunak, S. Brueckner, R. Matthews, J. Sauter, and S. Brophy. Real-Time Evolutionary Agent Characterization and Prediction. In *Proceedings of Social Agents: Results and Prospects (Agent 2006)*, Chicago, IL, Argonne National Laboratory, 2006.
- [25] H. V. D. Parunak, S. Brueckner, and R. Savit. Universality in Multi-Agent Systems. In *Proceedings of Third International Joint Conference on Autonomous Agents and Multi-Agent Systems (AAMAS 2004)*, New York, NY, pages 930-937, ACM, 2004. <http://www.newvectors.net/staff/parunakv/AAMAS04Universality.pdf>.
- [26] H. V. D. Parunak and S. A. Brueckner. Engineering Swarming Systems. In F. Bergenti, M.-P. Gleizes, and F. Zambonelli, Editors, *Methodologies and Software Engineering for Agent Systems*, pages 341-376. Kluwer, 2004. <http://www.newvectors.net/staff/parunakv/MSEAS04.pdf>.
- [27] H. V. D. Parunak, R. Rohwer, T. C. Belding, and S. Brueckner. Dynamic Decentralized Any-Time Hierarchical Clustering. In *Proceedings of Proceedings of the Fourth International Workshop on Engineering Self-Organizing Systems (ESOA'06)*, Hakodate, Japan, Springer, 2006. <http://www.newvectors.net/staff/parunakv/SODAS06.pdf>.
- [28] H. V. D. Parunak, R. Savit, and R. L. Riolo. Agent-Based Modeling vs. Equation-Based Modeling: A Case Study and Users' Guide. In *Proceedings of Multi-agent systems and Agent-based Simulation (MABS'98)*, Paris, FR, pages 10-25, Springer, 1998. <http://www.newvectors.net/staff/parunakv/mabs98.pdf>.
- [29] J. Pearl. *Probabilistic Reasoning in Intelligent Systems*, Morgan-Kaufmann, 1988. San Francisco, CA, Morgan-Kaufmann, 1988.
- [30] A. S. Rao and M. P. Georgeff. Modeling Rational Agents within a BDI Architecture. In *Proceedings of International Conference on Principles of Knowledge Representation and Reasoning (KR-91)*, pages 473-484, Morgan Kaufman, 1991.
- [31] M. Resnick. *Turtles, Termites, and Traffic Jams: Explorations in Massively Parallel Microworlds*. Cambridge, MA, MIT Press, 1994.
- [32] J. A. Sauter, R. Matthews, H. V. D. Parunak, and S. Brueckner. Evolving Adaptive Pheromone Path Planning Mechanisms. In *Proceedings of Autonomous Agents and Multi-Agent Systems (AAMAS02)*, Bologna, Italy, pages 434-440, ACM, 2002. [www.newvectors.net/staff/parunakv/AAMAS02Evolution.pdf](http://www.newvectors.net/staff/parunakv/AAMAS02Evolution.pdf).
- [33] J. A. Sauter, R. Matthews, H. V. D. Parunak, and S. A. Brueckner. Performance of Digital Pheromones for Swarming Vehicle Control. In *Proceedings of Fourth International Joint Conference on Autonomous Agents and Multi-Agent Systems*, Utrecht, Netherlands, pages 903-910, ACM, 2005. <http://www.newvectors.net/staff/parunakv/AAMAS05SwarmingDemo.pdf>.
- [34] K. Sycara. Intelligent Software Agents. 2001. Web Page, <http://www-2.cs.cmu.edu/~softagents/retsina.html>.
- [35] P. Weinstein, H. V. D. Parunak, P. Chiusano, and S. Brueckner. Agents Swarming in Semantic Spaces to Corroborate Hypotheses. In *Proceedings of AAMAS 2004*, New York, NY, pages 1488-1489, ACM, 2004. <http://www.newvectors.net/staff/parunakv/AAMAS04AntCAFE.pdf>.
- [36] D. Weyns, A. Omicini, and J. Odell. Environment as a first-class abstraction in multiagent systems. *Journal of Autonomous Agents and Multi-Agent Systems*, 17, 2007.

- [37] M. Wooldridge. *Reasoning about Rational Agents*. Cambridge, MA, MIT Press, 2000.
- [38] M. Wooldridge, N. R. Jennings, and D. Kinny. The Gaia Methodology for Agent-Oriented Analysis and Design. *International Journal of Autonomous Agents and Multi-Agent Systems*, 3(Forthcoming), 2000.