Aaron Sloman and Brian Logan

Building Cognitively Rich Agents USING THE SIM_AGENT TOOLKIT

Why your software agents will have emotions.

SYNTHETIC AGENTS WITH VARYING DEGREES OF INTELLIGENCE AND autonomy are being designed in many research laboratories. The motivations include military training simulations, educational software, games and entertainment, digital personal assistants, software agents managing Internet transactions or purely scientific curiosity. Different approaches are being explored, including, at one extreme, research on the interactions between agents, and at the other extreme, research on processes within agents.

The first approach focuses on forms of communication, requirements for consistent collaboration, planning of coordinated behaviors to achieve collaborative goals, extensions to logics of action and belief for multiple agents, and types of emergent phenomena when many agents interact, for instance, taking routing decisions on a telecommunications network.

The second approach focuses on the *internal* architecture of individual agents required for social interaction, collaborative behaviors, complex decision making, learning, and emergent phenomena within complex agents. Agents with complex internal structure may, for example, combine perception, motive generation, planning, plan execution, execution monitoring, and even emotional reactions.

We expect the second approach to become increasingly important for large multiagent systems deployed in networked environments, as the level of intelligence required of individual agents increases. This is particularly relevant to work on agents that must cooperate to perform tasks requiring planning, problem solving, learning, opportunistic redirection of plans, and fine

BOB CASE

judgement, in a partially unpredictable environment. In such contexts, important new information about something other than the current goal can arrive at unexpected times or be found in unexpected contexts, and there is often insufficient time for deliberation. These situations require reactive mechanisms. However, some tasks involve achieving new types of goals or acting in novel contexts, which may require deliberative mechanisms. Dealing with conflicting goals, or adapting to changing opportunities and cultures may require sophisticated motive processing mechanisms.

Motivations for such research include: an interest in modeling human mental functioning (emotions, for example), a desire for more interesting synthetic agents ("believable agents") in games and computer entertainments, and the need for intelligent agents capable of performing more complex tasks than those that are currently possible.

Many researchers are investigating relatively simple agents performing restricted tasks, such as sorting a manager's incoming email, or giving a novice user information about a software system. Some researchers believe that in the future a manager in a large company might be replaced by a team of software agents, some attempting to obtain and summarize information about other companies, some monitoring and evaluating activities in subsidiary companies, and others taking strategic decisions.

We believe cognitively rich internal mechanisms and structures are necessary to model complex communication and planning with (and between) agents. However, these may have unexpected side effects. It has been argued [10] that intelligent robots will need mechanisms for coping with resource limits and that interactions within these mechanisms will sometimes produce emotional states involving partial loss of control of attention, as already happens in humans.

Motivational and emotional processes may also be involved in attempts to persuade a character in an interactive story to adopt a particular belief, or in discussing the relative importance of various tasks with a personal assistant agent. Similar mechanisms may be involved when communication operates at several different levels, for example, when it involves humor, irony, or metaphor. Related claims about motivational and emotional requirements for intelligence can be found in [3, 7].

Different Kinds of Architectures

The internal architecture of an agent can take different forms, with varying degrees of intelligibility and modularity. At one extreme there may be very large numbers of low-level components (for example, neurons) all interacting in such a way that useful global behavior emerges, without any explicitly programmed intervening structure to account for the behavior or any architecture we can comprehend. At the other extreme, agents may have a clearly defined modular architecture with a hierarchical composition of separable components performing different tasks using different, though strongly interacting, mechanisms to perform those tasks. In between are architectures with varying degrees of functional decomposability and intelligibility.

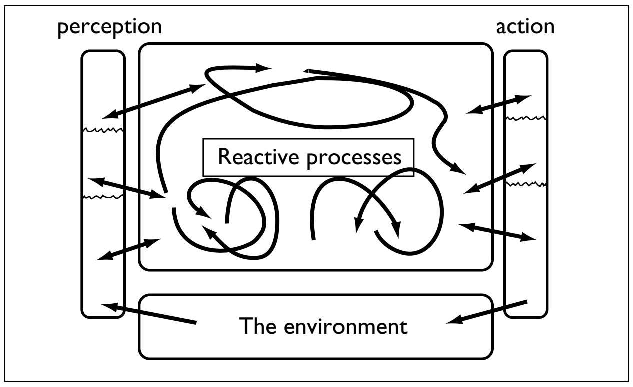


Figure 1. The reactive layer

Another dimension of variation concerns the extent to which the internal architecture is purely "reactive," where detection of internal or external conditions immediately generates new internal or external responses, which in turn could trigger new reactions (see Figure 1). The internal and external feedback loops and changes of state that alter a reactive agent's responses to new sensory inputs can combine to achieve quite sophisticated behavior. Examples could include software agents monitoring and reacting to sensor readings in a chemical plant to control complex and varied processes. More sophisticated reactive systems may require processing at different levels of abstraction in the sensory and motor subsystems, for example, detecting not only low-level signals but also more abstract patterns in the data, and triggering complex behaviors involving coordination of several effectors. However, this may prove inadequate in some contexts.

Deliberative architectures achieve even greater flexibility by allowing sets of possible actions to be assembled and evaluated without necessarily being selected for execution. A deliberative architecture (which might be implemented using internal reactive mechanisms) requires a large amount of stored knowledge, including explicit knowledge about which actions are possible and relevant in various circumstances, and what the effects of various actions are in those circumstances. It also requires a

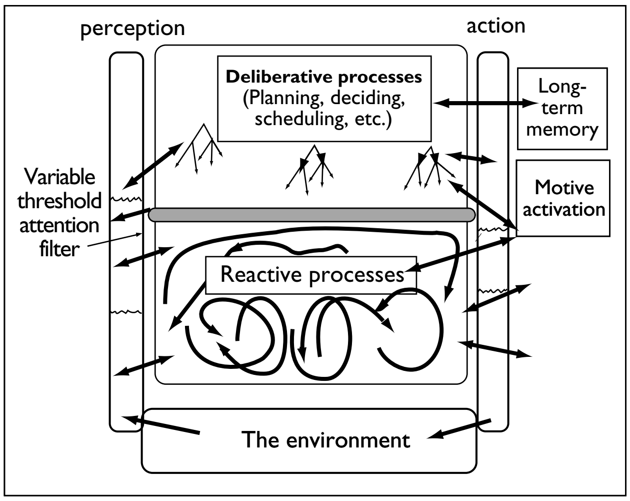


Figure 2. Reactive and deliberative layers

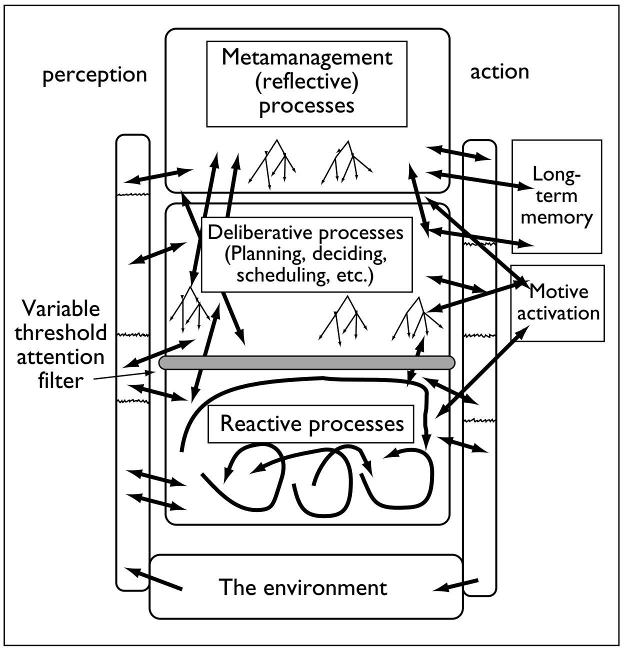

perception
Metamanagement (reflective) processes
action
Long-term memory
Deliberative processes
(Planning, deciding, scheduling, etc.)
Motive activation
Variable threshold attention filter
Reactive processes
The environment

Figure 3. The metamanagement layer

reusable short term memory for building structures representing possible action sequences in order to evaluate their consequences. The reuse of memory, the inability of long term storage to answer many questions in parallel, and the sequential nature of plan construction will typically make a deliberative system much slower than a reactive one. These resource limits might necessitate an attention filter to suppress low priority interruptions. New levels of abstraction in the perceptual and action subsystems may be required to match the more abstract processing in the deliberative system.

Such a deliberative architecture might be required for a personal assistant or a plant control mechanism able to cope in novel circumstances where routine reactions are sometimes inappropriate and where trial-and-error behavior is too dangerous or too time consuming.

Any externally observable behavior that can be produced by a deliberative architecture can, in principle, be produced by a purely reactive architecture, as long as an exhaustive collection of condition-action rules has been assembled in advance by a designer or perhaps by an evolutionary process. Producing a comprehensive set of reactive behaviors in advance may take an inordinately long time and may require excessive amounts of storage.

In deciding between reactive and deliberative architectures in an environment with a high degree of complexity and variability, there are complex trade-offs involving the time and effort required to precompile all the potentially relevant rules, the space required to store them, and the mechanisms required for efficiently finding and selecting between potential contenders at run time. Our conjecture is that evolution has solved this problem in natural systems with a hybrid architecture involving closely integrated concurrently active deliberative and reactive subarchitectures (see Figure 2).

Such a hybrid architecture may also include mechanisms for transferring some of the results of the deliberative layer to the reactive layer (for example, responses to stereotypical situations) to improve speed of performance or release the deliberative layer to attend to other things. Many learned human skills are like this, and similar types of skill acquisition may be needed in software agents and robots.

Self-Monitoring Systems

Work on multiagent systems often assumes that interacting agents are perfectly rational or have a fixed collection of motives (goals, preferences, and standards), requiring only simple cognitive mechanisms such as payoff matrices or utility functions. While this may be appropriate for systems consisting of simple agents with ant-like cooperation, it is inadequate for societies of intelligent human-like software agents or autonomous robots. In such situations the set of motives is not fixed, as new motives can be generated in new contexts, for instance the goal of helping a friend who is in trouble, or having to deal with unexpected conflicts between high-level objectives and standards or preferences. Realistic multiagent systems will have to take account of these internal within-agent conflicts as well as between-agent conflicts and negotiations. For example, members of a plant management software team may encounter conflicts between safety of

Agents in Tank Battle Simulations

Jeremy Baxter and Richard Hepplewhite

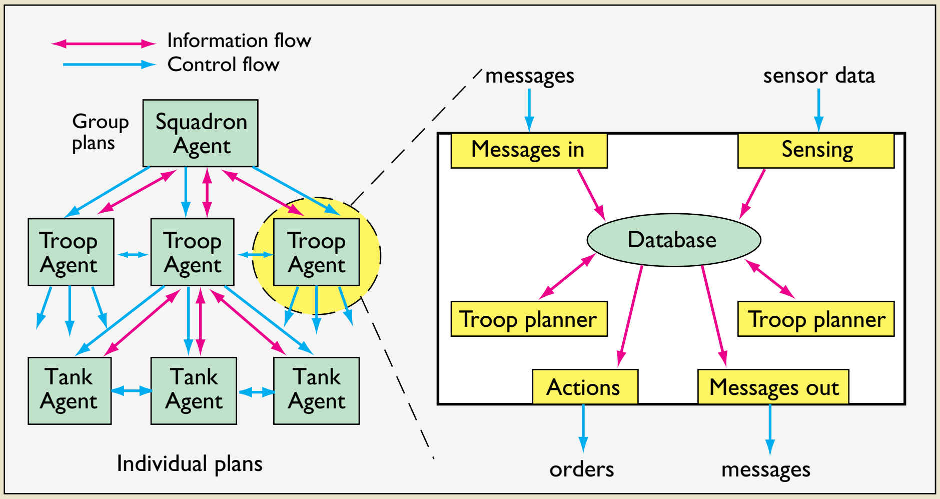
Networks of computers can be used to produce a digital virtual environment (DVE) where multiple participants can interact. This technology is extremely attractive to the military to provide training simulations. By the use of mock-up vehicles and high-fidelity visual systems, trainees get a window onto a virtual world populated by simulated vehicles interacting over a realistic terrain surface. Some of these vehicles are controlled by human trainees, others by computers. It is essential that trainees find the behavior of the computer-controlled vehicles realistic. Currently most computer forces are semiautomated using finite state machines or rule bases to govern their behavior, but requiring constant supervision by a human controller [1]. However, the vehicles can become increasingly autonomous as AI and agent techniques develop, thus reducing the number of human controllers as well as the hefty manpower bill associated with running big training simulations [2, 3].

We are concentrating on developing agents to control tanks within ground battle simulations. Here, tactical behavior is governed by two main factors—the terrain over which the tanks are moving and their beliefs about the enemy. In trying to produce battlefield behavior that mimics a human tactician, it is advantageous to model the command structure used by the army. This helps with the gathering of knowledge from subject matter experts and enables a hierarchical decomposition of the problems. The figure appearing in this sidebar shows the hierarchy of agents—high-level commanders are given objectives that are used to produce lower-level objectives for their subordinates. Information flows both up and down the command chain and agents need to cooperate with their peers to achieve the overall goal set by their commander. This natural decomposition of the problem allows higher-level agents to work on long-term plans while the individual tank agents carry out orders designed to achieve more immediate objectives.

The Agent Toolkit. To provide the framework within which agents operate, we use the SIM_AGENT toolkit (see the article by Sloman and Logan in this section). It allows multiple agents to be run and controls their communication with each other and with the physical simulation of the battlefield. Internally, these agents run a number of processes which share data held in a central database; as shown in the figure. The processes are scheduled to run for a few steps at a time and each performs a different task, for example, assessing incoming sensor data, monitoring the progress of a plan, or communicating with other agents. This allows agents to pursue many mental tasks simultaneously. Scheduling ensures each agent and process gets a fair share of the available processing power and enforces real-time operation.

Agents need to incorporate fast reactions to their environment to cope with unexpected events while at the same time perform complex reasoning about the terrain. Our agents, therefore, combine the use of anytime planning techniques with reactive plan execution systems designed to operate in a real time environment [2]. These are implemented as separate processes within the agents allowing the combination of reactive and deliberative behavior.

For example, when considering how to place forces to block enemy movement through an area, the squadron commander has to consider a number of factors. Positions must be identified that give protection to the defending forces, but also provide a good view of the potential enemy approach routes and are close enough to other



Hierarchy of agents and internal structure of an agent groups to offer mutual support. The squadron defensive planner identifies candidate positions for the defending troops by analyzing the protection afforded by the terrain. Combinations of these positions are ranked in terms of the overall breadth and depth of the engagement area which can be seen (and fired upon) from them. During this optimization process, the best deployment identified so far is cached, so that it can be executed if the time for planning runs out.

The battlefield is so dynamic that detailed individual tank plans are unlikely to remain valid for long, so tanks operate by selecting actions from a recipe book covering general situations. These short-term actions combine the agent's goals with reactions to the enemy and the terrain. Plans are developed as sequences of these actions and are assessed by carrying out an internal simulation of their probable effects to identify how well they would perform in the present situation. The use of an internal simulation also allows assumptions about the future state of the world to be incorporated into the plan. During execution, the agent can identify cases where the assumptions turn out to be false and a new plan is required.

Simulations using these agents have been shown to military experts who have confirmed their terrain-related behavior is more realistic than that produced by simple, finite state machine-based approaches. The agent-based approach to this problem has several advantages, including decomposition of the problem, natural distribution between machines, and permitting clear comparisons with reality by human experts. Future work will focus on the deficiencies in overall group behavior and a need to plan to gather information. C

References

1. Courtemanche, A.J., and Ceranowicz, A. ModSAF development status. In *Proceedings of the 5th Conference on Computer Generated Forces and Behavioural Representation.* (May 9–11, Orlando Fla.). Institute for Simulation and Training, 1995, 3–13.
2. Hepplewhite, R.T., and Baxter, J.W. Broad agents for intelligent simulation. In *Proceedings of the 6th Conference on Computer Generated Forces and Behavioural Representation*. (July 23–25, Orlando, Fla.) Institute of Simulation and Training, 1996, 319–327.
3. Hill, R., Chen, J., Gratch, J., Rosenbloom, P., and Tambe, M. Intelligent agents for the synthetic battlefield: A company of rotary wing aircraft. In *Proceedings of AAAI97 and IAAI97* (July 27–31, Providence, R.I.). AAAI Press, 1997, 1006–1012.

Jeremy Baxter (jbaxter@signal.dera.gov.uk) is a senior scientist at the UK Defence Evaluation and Research Agency (DERA), Malvern, UK.
Richard Hepplewhite (rth@signal.dera.gov.uk) is a senior scientist at the UK Defence Evaluation and Research Agency (DERA), Malvern, UK.

human workers and economy of plant operation. Simulated battlefield commanders or simulated antiterrorist strategists may have to detect and handle conflicts between protecting civilians and capturing opponents.

In these contexts an agent may benefit from an additional architectural layer with the ability to monitor, evaluate, and possibly modify internal processes of various kinds (see Figure 3). This could be based on a mixture of internal reactive and deliberative processes directed at the agent's internal sensory buffers, its goals and preferences, deliberative strategies, records of recent decisions, and the like. Such a system might discover conflicts of motivation and devise a strategy for dealing with them. Or it may notice that certain problem solving processes are taking too long, so that a less cautious but more speedy strategy is required.

We expect that some synthetic agents, including agents working in groups or teams, will require a similar variety of internal processing. For instance, metamanagement is required to decide when to stop trying to solve a problem alone and ask for help.

In humans, this "metamanagement" capability seems to be used for a variety of purposes including social control via inculcation of ethical and other standards of self-assessment, and also some kinds of learning in which deliberative processes are evaluated, found wanting, and improved. It also seems to become impaired during times of stress or in certain emotional disturbances.

Some of the internal processes, including both deliberative processes and reactive processes in which chains of associations are needed to solve a problem, may be too slow for an environment where opportunities and dangers requiring immediate action can turn up unexpectedly. A *global alarm system* using fast pattern-recognition mechanisms able to trigger stereotyped internal and external responses could help in dealing with such situations. The need for rapid reactions could make such a mechanism sometimes produce erroneous responses. This trade-off between speed and intelligence can be found in some human emotions and may also occur in synthetic agents. A trainable alarm system might reduce the frequency of mistakes.

Multiprocessing Within Agents

Sophisticated agents will include many concurrent processes performing various kinds of tasks, such as:

Realistic multiagent systems will have to take account of within-agent conflicts as well as between-agent conflicts and negotiations.

- Analysis and interpretation of sensory data at different levels of abstraction;
- Monitoring for "alarm" conditions, sometimes producing rapid global redirection of processing;
- Generation of new motives;
- Assessment of motives and deciding whether to adopt them;
- Construction of plans for achieving a motive (either after deciding to adopt it or as part of the Process of deciding whether to do so);
- Deciding which currently adopted plans should be executed now, which postponed, which abandoned;
- Learning new patterns and categories, evaluating past performance;
- Attempting to interpret the observed behavior of other agents;
- Understanding explicit (and implicit) communications from others; and
- Deciding what to say to others and how to say it, and many other tasks and behaviors.

In addition to relatively high-level processes, a robot with a complex physical body, like an animal, may also require a large number of reactive processes monitoring and controlling various aspects of bodily function, posture, gait, fine control of movement of hands, and so forth. Software agents dealing with high-volume information streams requiring fast decisions may also need a well-trained reactive layer.

Multiple Programming Paradigms: The SIM_AGENT Toolkit

For some years the Cognition and Affect project at the University of Birmingham has been exploring the problems sketched previously [1, 9]. To support this research we designed and implemented a software toolkit, SIM_AGENT, to allow exploration of ideas through implementation using rapid prototyping [8, 11]. We found it necessary to provide support for a variety of programming paradigms in addition to widely used object-oriented techniques.

For instance, the need for flexible reactive internal mechanisms within agents is supported by a rule-based programming style, in which diverse mechanisms within an agent are implemented in different condition-action rulesets.

To allow a wide range of conditions to trigger internal processes and in order to support sophisticated internal reasoning, a list-processing paradigm is used to implement the internal databases and communication channels between parts of an agent and between agents.

To support "sub-symbolic" processing, the rule-based system has interfaces to procedural programs and both conditions and actions can be linked to mechanisms like neural nets when appropriate.

The need for fast low-level processes is met by using compiled code. More flexible processes subject to self-monitoring can use interpreted internal languages. This also allows users to vary the relative speeds of different components to explore hypotheses about resource-limited systems. The presence of an incremental compiler allows rapid development and testing, as well as supporting easy experimentation by students in a teaching environment. Use of indirect references allows rules to be redefined easily or procedures to be recompiled during a pause in an experimental run of the system. Automatic store management and garbage collection prevents memory leaks.

The implementation language is Pop-11, a sophisticated extendable Lisp-like language with a Pascal-like syntax, in the Poplog environment. (Pop-11 is also at the heart of Clementine, a commercially successful datamining product.)

The SIM_AGENT toolkit allows construction of sets of agents in which each agent has a multicomponent architecture, in which different components operate in parallel, using different kinds of mechanisms. These can include neural nets or other reactive mechanisms alongside AI symbol manipulating mechanisms, used for solving complex planning problems, or for certain metamanagement tasks. For example, a system may use a neural net to obtain evidence of fraud in credit transactions based on subtle combinations of data, together with deliberative mechanisms to analyze suspect cases in greater detail. A factory automation system could use a set of reactive procedures to handle routine management and a deliberative planner to handle novel problems, such as the production of a new product. A metamanagement layer could decide whether to change the criteria being used by the planner, or whether to suspend planning and begin acting because of shortage of time or because of the need to collect more information by acting on a partial plan.

Many agent toolkits are based on a commitment to a particular architecture, for example, SOAR [4]. Because SIM_AGENT has no such commitment it has proved attractive to researchers and students wishing to explore alternative designs in a variety of fields, including simple computer games, battlefield simulations in training software (at DERA Malvern, UK), modeling aspects of human cognition and emotions [2, 12], agents planning in a dynamic domain [5] and telecommunications. The ease of development in an AI environment makes it particularly useful for teaching, using simple demonstrations that students can modify.

Planned extensions arising out of our recent work on metamanagement tasks include improved facilities for self-monitoring and self-modification by agents. Recent work on requirements for agent toolkits is reported in [6].

Conclusion

The trade-offs between different types of architectures are not clear and much research is still required. Although more efficient toolkits dedicated to particular types of agent architectures will be more appropriate in tasks in which the architectural requirements are narrowly specified in advance, we expect that, increasingly, toolkits with the flexibility of SIM_AGENT will be needed as research focuses strongly on agents with more sophisticated multi-functional architectures, which raise many of the hardest unsolved problems in AI. Currently these are ignored in much multiagent systems research, but that cannot last. C

References

1. Beaudoin, L.P. and Sloman, A. A study of motive processing and attention. In A. Sloman, D. Hogg, G. Humphreys, D. Partridge, and A. Ramsay, Eds., *Prospects for Artificial Intelligence*, IOS Press, Amsterdam, 1993, pp. 229–238.
2. Davis, D.N. Reactive and motivational agents: Towards a collective minder. In J.P. Mueller, M.J. Wooldridge, and N.R. Jennings, Eds., *Intelligent Agents III—Proceedings of the Third International Workshop on Agent Theories, Architectures, and Languages.* Springer-Verlag, 1996.
3. Goleman, D. *Emotional Intelligence: Why It Can Matter More than IQ.* Bloomsbury Publishing, London, 1996.
4. Laird, J.E., Newell, A., and Rosenbloom, P.S. SOAR: An architecture for general intelligence. *Artificial Intelligence, 33* (1987), 1–64.
5. Logan, B. and Alechina, N. A* with bounded costs. In *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98)*, AAAI Press/MIT Press, Menlo Park CA and Cambridge MA, 1998. pp. 444–449.
6. Logan, B., and Baxter, J., Eds. *Software Tools for Developing Agents: Papers from the 1998 Workshop.* Technical Report WS-98-10. American Association for Artificial Intelligence. Menlo Park, CA, 1998.
7. Picard, R. *Affective Computing.* MIT Press, Cambridge, MA 1997.
8. Poli, R., Brayshaw, M., and Sloman, A. A hybrid rule-based system with rule-refinement mechanisms. In *Proceedings of Expert Systems'95 Conference*, Cambridge UK. British Computer Society Expert Systems Group, 1995.
9. Sloman, A. What sort of control system is able to have a personality. In R. Trappl and P. Petta, Eds., *Creating Personalities for Synthetic Actors: Towards Autonomous Personality Agents.* Springer (Lecture Notes in AI), Berlin, 1997, pp. 166–208.
10. Sloman, A. and Croucher, M. Why robots will have emotions. In *Proceedings of the 7th International Joint Conference on AI*, Vancouver, 1981.
11. Sloman, A. and Poli, R. SIM_AGENT: A toolkit for exploring agent designs. In M.J. Wooldridge, J.P. Mueller, and M. Tambe, Eds., *Intelligent Agents II—Proceedings of the Second International Workshop on Agent Theories, Architectures, and Languages.* Springer-Verlag, 1996, pp. 392–407.
12. Wright, I.P. and Sloman, A. *Minder1: An implementation of a protoemotional agent architecture.* Technical Report CSRP-97-1. The University of Birmingham, School of Computer Science, 1997.

Aaron Sloman (a.sloman@cs.bham.ac.uk) is a professor of Artificial Intelligence and Cognitive Science in the School of Computer Science at the University of Birmingham, UK.

Brian Logan (b.s.logan@cs.bham.ac.uk) is a lecturer in Artificial Intelligence in the School of Computer Science at the University of Birmingham, UK.

Further Information and Availability

The toolkit and additional information about it are freely available via FTP and WWW:

- A brief overview with some MPEG movies can be found at www.cs.bham.ac.uk/~axs/cog_affect/sim_agent.html
- The Cognition and Affect FTP repository of theoretical papers is at ftp://ftp.cs.bham.ac.uk/pub/groups/cog_affect/
- The Birmingham Poplog directory is at the following location, where the README file lists information about SIM_AGENT and its supporting libraries: ftp://ftp.cs.bham.ac.uk/pub/dist/poplog
- SIM_AGENT requires Poplog, available from Integral Solutions Ltd.: www.isl.co.uk/