Report on DARPA Workshop on Self-Aware Computer Systems

Eyal Amir
Department of Computer Science
University of Illinois
Urbana, IL 61801-2302
eyal@cs.uiuc.edu

Michael L. Anderson
Department of Psychology
Franklin & Marshall College
Lancaster, PA 17604-3003
michael.anderson@fandm.edu

Vinay K. Chaudhri
Artificial Intelligence Center
SRI International
Menlo Park, CA 94025-3493
vinay@ai.sri.com

Abstract

Self-Aware Computer Systems is an area of basic research, and we are only in the initial stages of our understanding of what it means: what it means to be self-aware, what a self-aware system can do that other systems cannot do, and what are some of the immediate practical applications and challenge problems. This report captures some of the salient points discussed during the DARPA workshop on Self-Aware Computer Systems held on April 27-28, 2004 in Washington D.C.[1]

Introduction: What is Self-Awareness?

In humans, self-awareness is described and defined in many different ways. For instance, knowledge about one's own permanent aspects or of one's relationships to others; awareness of one's sensory experiences and their implications; awareness of one's beliefs, desires, intentions, and goals; knowledge about one's own knowledge or lack thereof; awareness of one's attitudes such as hopes, fears, regrets, and expectations; and the ability to perform mental actions such as forming or dropping an intention (McCarthy, 2004).

In machines, self-awareness is likely to be of interest only for long-lived programs—programs that operate over a period of time, and interact with the external world or other programs. Machines do not have to be self-aware in the same ways as humans are, but some forms of self-awareness will be useful for machines. For example, a machine may need to reason about what it can and cannot do; reason about ways to achieve new knowledge and abilities; know how it arrived at its current beliefs, maintain a critical view of them, and use this knowledge to revise its beliefs in light of new information; and regard its entire mental state up to the present as an object, and have the ability to transcend it and think about it.

Self-awareness also seems to have a social aspect. For instance, a system may have a theory of itself that it can use to interact with others. Such self-awareness may have uses in multiagent systems for dealing with errors in communication, argumentation, negotiation, and so on. Consider the kinds of self-awareness that would be involved in the ability not just to *say* "When I said 'X' I meant 'Y'", but to know when it would be appropriate to do so.

Such an ability suggests a degree of self-awareness that no system currently has. There are some simple forms of awareness that systems *do* have. For example, many systems monitor their battery level, or their amount of available memory; systems are able to produce a processing trace, to engage in garbage collection, and so on. Thus, to focus attention on the sorts of self-aware systems we have in mind, it is useful to make an explicit distinction between the following notions.

Explicit Self-Awareness. The computer system has a full-fledged self-model that represents knowledge about itself (e. g., its autobiography, current situation, activities, abilities, goals, knowledge, intentions, knowledge about others' knowledge of its knowledge) in a form that lends itself to use by its general reasoning system and can be communicated (possibly in some language) by a general reasoning system.

Self-Monitoring. The computer system monitors, evaluates, and intervenes in its internal processes, in a purposive way. This does not presuppose that the monitored information lends itself to general reasoning; in fact, there may be no general reasoning (e.g., operating systems, insects).

Self-Explanation. The agent can recount and justify its actions and inferences. In itself, this does *not* presuppose a full-fledged self-model, or integration of knowledge needed for self-explanation into a general reasoning capability.

[1] The position statements submitted by various participants are available online at http://www.ihmc.us/users/phayes/DWSAS-statements.html, and the presentations made during the workshop are available at http://www-formal.stanford.edu/eyal/aware.

The primary focus of the workshop, and the area that we suggest is fruitful for basic research, is the notion of explicit self-awareness: what it is, how to implement it, and what it is good for. Note that although one of the attractions of work in self-aware systems is the promise that self-awareness can play a role in helping systems be more adaptive and robust, these properties do not themselves presuppose or necessarily require explicit self-awareness, or even general reasoning. Many researchers expect, however, that self-awareness will be required for increasing *degrees* and different *kinds* of adaptivity and robustness over those that are displayed by current systems. Likewise, although properties such as self-monitoring, self-explanation, and goal-directedness are likely to be present in self-aware computer systems, these by themselves do not constitute self-awareness. The system may, however, exhibit self-awareness in the way it marshals these abilities, for example, in choosing what to monitor, or by knowing that it is garbage collecting (and why).

As a concrete example of what a self-aware computer system can do that a system without self-awareness cannot do, consider (McCarthy, 1996):

- A fish cannot, while not swimming, review its previous swimming performance so as to swim better next time.
- A fish cannot take instruction from a more experienced fish in how to swim better.
- A fish cannot contemplate designing a fish better adapted to certain swimming conditions than it is.

A self-aware swimming robot would be able to perform all three of these tasks.

Neuroscientific Viewpoint on Self-Awareness

The neuroscience view on self-awareness is driven by the observation that at any given time humans seem to be aware of only a small fraction of what they know; most cognitive processing seems to go one below the threshold of awareness. Moreover, human self-awareness seems to be largely mismatch driven: we become more aware of our thoughts, behavior, and so on when our expectations for experience are violated.

One way of implementing some of these features of self-awareness in a machine would be with an architecture called the Global Workspace (GW)—a massively parallel society of specialized processors coupled with a central information exchange (the "workspace") (Baars, 1988). Executive controls ("self systems") can be implemented to monitor and control large sets of specialized processors. These processors would normally operate semiautonomously, outside of central awareness, but expectation violations might cause their operations to rise above the threshold and attract the "attention" of higher-level control systems.

The neuroscientific view raises a number of unanswered questions, for example, Can awareness be caused by complexity in situations and not just by mismatches? Can we distinguish among species based on consciousness? Why does the brain appear to be neurologically serial?

Psychological Viewpoint on Self-Awareness

In psychology, two main current areas of research are relevant to self-awareness. The first, deriving primarily from the work of Piaget, focuses on the development of a sense of self over one's lifetime (Rochat, 2003). Research suggests that infants are born with a rudimentary concept of self, manifested in such simple things as differentiating a self-touch from the touch of another. At around 18 months infants manifest a more "conceptual" sense of self, supporting the ability to recognize themselves in a mirror. Other milestones include the ability to recognize that what they believe and what someone else believes may be different, which generally happens by 4 to 5 years.

The other line of research originates in more recent cognitive psychology, and focuses on the monitoring and executive control of cognitive processes (Nelson, 1992). In the general account of metacognitive monitoring and control, cognition occurs at two levels: object and meta. Object-level cognition is "normal" cognition performed in direct support of achieving some goal: planning a birthday party, memorizing vocabulary words, and the like. Meta-level cognition monitors and controls object-level processes. Monitoring involves functions like judging the completeness of a plan, or the degree of confidence that one has learned something (making a "judgment of learning"). Control involves using this information to do such things as starting or stopping processes, selecting or altering strategies, and allocating time between processes.

Having surveyed some of the general questions and issues involved in understanding self-awareness, we will devote the rest of this report to focusing on the challenges involved in formalizing and implementing self-awareness in computational systems. The next section outlines a few examples of existing systems and discusses the degree to which they implement self-awareness. This is followed by a section discussing open research questions, and a section on challenge problems that we hope will spur interest in self-aware systems, and result in significant advances in the state of the art.

Implemented Systems

There are examples of implemented systems that exhibit features of self-aware computer systems. We give here a description of each system and what makes it self-aware.

Several systems have been developed at the University of Maryland based on the idea of a Meta-Cognitive Loop (MCL) (Anderson and Perils, 2005). The meta cognitive loop has three main steps (1) monitor events, to note possible anomalies (deviations from expectations), (2) assess the type and possible strategies for dealing with any anomalies, and (3) guide one or more strategies into place while continuing to monitor (looping back to step (1) for new anomalies that may arise either in the normal course of operation, or as part of the repair strategy. MCL allows systems to reason about their own performance failures, and to make targeted changes to their own modules (including learning new things) to improve performance. In one study it was shown that the performance of standard reinforcement-learning algorithms, which tend to recover slowly when there is a sudden change in reward structure, can be significantly enhanced when coupled with even a minimal form of MCL.

The CASSIE system, developed at SUNY Buffalo, is an autonomous agent with a natural language front end (Shapiro, 1989). Its representation includes a term to represent itself (an "I"), and it has the ability to represent and refer to its own beliefs and actions. It uses a first order logic representation called SNePS. Functionally, Cassie has beliefs about herself and others, knows with whom she is talking, can remember and report on her actions using appropriate tense and aspect, can be informed about the actions of others, and can properly use certain deictic terms such as I, you, and here.

The KRAKEN system, developed by Cycorp, is a knowledge acquisition front end to the Cyc knowledge base (Witbrock, et al. 2003). The acquisition proceeds by the system asking the user questions. In the initial versions of KRAKEN, the system had limited self-awareness capabilities: it did not know why or how it was asking questions; it did not substantially distinguish the current discourse context from the general knowledge base; and it was only dimly aware of the person with whom it was communicating. This substantially reduced the effectiveness of the knowledge formation process. Since the system had no represented goals, it was unable to decide whether asking a particular question best satisfied those goals. Its questions to the user had no context, because of which it could not choose the best way to ask them, or frame its questions in terms of preceding and subsequent interactions. Finally, the system could not recognize that its questions may be annoying the user, because it did not really know what its questions are, or what effect they might have.

To address these shortcomings, Cycorp is taking some first steps toward implementing self-awareness in KRAKEN. They are approaching this problem by representing and reasoning about the system itself within the knowledge base. KRAKEN now keeps a record of user utterances, their interpretations, the information presented by the system to the user, and so on. In short, it keeps a formal representation of the entire interaction that can be reasoned with. These facilities will serve as the basis for structuring interactions with the user.

Research Issues

We consider some scientific challenges that need to be addressed in the development of self-aware computer systems.

Knowledge Representation

The basic ability to represent knowledge is clearly necessary for building self-aware computer systems. More importantly, self-aware systems need to represent knowledge *about* their knowledge. Aspects of self-awareness that require such representations include planning under uncertainty, problem diagnosis, decision explanation, collaborative or adversarial interactions with multiple agents, and exploration of new domains.

More technically, the sorts of knowledge representation required to support self-awareness include the ability to represent knowledge and beliefs as modal operators; the representation of self and other agents; the representation of sensing and observing; memory operations such as forgetting, recalling, and recognition; the ability to represent conceptual objects (such as beliefs *per se*); the ability to represent and reason about emotions; and the ability to represent a naïve or folk psychology. Specific research in knowledge representation under the following themes seems to be central for self-aware computer systems:

- A focus on under-researched issues such as memory, focus of attention, contradiction resolution, expansion of vocabulary with new information and situations, changes in the system and its beliefs, emotion, and folk-psychology
- Integration of traditional KR areas (e.g., sensing, knowledge and non-monotonicity)

Self and Social Agency

Much of human self-knowledge seems to involve thinking of the self as a social agent, an agent that makes promises, commitments, and so on. This involves a kind of self-awareness that depends in part on awareness that the agent is itself perceived by other 'selves'. This observation raises the following questions: Is this social dimension important or central, and how does it relate to other self-notions, such as the bodily self? What kind of analogous social assumptions would be appropriate for an artificial self-aware system?

We can distinguish two notions of *self-aware*, roughly distinguished as the *social-agent* sense and the *monitoring-executive* sense. The first sees the person as primarily an agent in a community of agents and the self as the locus of social commitment. The second sees the person as an isolated body/mind with the self as a locus of internal observation (introspection) and executive control. In humans these are identified, but are conceptually distinct and so could be distinguished in artificial systems. For example, consider a Question-Answering system with no body or bodily location, or a distributed robotic system.

Although there may be many systems that do not require much in the way of social awareness—for example, a toaster oven does not need to consider social aspects of its actions (toasting)—there are also examples of systems, such as medical advisers for long-term medical care, artificial pets, personal assistants, medical 'companions' for elderly people, where it seems essential to model the self as a social agent. And even the toaster oven may find some social awareness useful, if for instance it uses indicators (e.g., a red light or a bell), it might use social awareness to choose between them, or even use other indicators, if it knows that the target of a given indication does not know its meaning or is not aware of it.

Engineering Self-Sustainable Systems

A self-sustaining system is any combination of hardware and software that is able to notice errors, inefficiencies, and other problems in its own behavior and resources, and to repair them on the fly. We imagine that such systems could be quite large and that they might exhibit both hardware and software failures. This means that some of the failures will be intermittent or based on coincidences—of placement, of timing, and of naming.

Self-sustainability and self-awareness appear to be related at an implementation level, but perhaps also conceptually. A self-sustaining system performs some degree of reflection and can alter its own behavior based on what it sees, and this is what a self-aware system does.

We can draw inspiration from how biological systems sustain themselves. Biological systems maintain diversity, redundancy, biological modularity (codes can change), spatial compartmentalization (cells keep things physically together), stigmergy (making changes, observing, and making changes again, but still having state), symbiogenesis (combining genetic material), growing, staying alive, and so on. Software systems face a different set of challenges; for example, they need to maintain modularity in the face of changing context, and they need to dynamically adapt. Most organisms require different types of feedback and so will programs.

One approach to engineering self-sustaining programs is to implement an observational, programmable layer that not only observes, interrupts, and alters, but that consists of decaying persistent memory. For example, consider a tightly packed multiprocessor that can suffer heat-related failures when heavy-duty computations take place on it. Imagine an observational layer that is monitoring heat and memory or computation failure rates. If the observational memory were to hold data about the heat or failure rate and treat it like dissipating heat or dissolving chemicals, then adjacent memory and computational elements would "inherit" some of the characteristics of the problems and computations would not be scheduled onto them as frequently, thereby reducing errors by moving computations away from overheated areas.

Engineering self-sustainable systems can be a concrete practical application of self-awareness. It is not apparent if such systems will require notions such as beliefs, and representations of self that were traditionally studied in formal knowledge representation, but they certainly can use concepts such as the Meta-Cognitive Loop.

Cognitive Architecture

One can think of the architecture of a self-aware computer system from three perspectives: (1) an autonomous agent view, (2) an information processing view, and (3) a biological view. Each view suggests a different set of components.

From an autonomous agent view, a self-aware system must have sensors, effectors, memory (including representation of state), conflict detection and handling, reasoning, learning, goal setting, and an explicit awareness of any assumptions. The system should be reactive, deliberative, and reflective.

From an information processing view, a self-aware system worries about provenance of information (where the information came from (self, other)), reference of information (what the information pertains to (self, other)), the use of information (who can use the information (self, other)), and the intention of use (what is the information used to affect (self, other)).

From the biological viewpoint, a self-aware system is a decentralized system (as opposed to centralized e.g., a parliament or theater model), and components have an executive-slave relationship. The influence of this view on the engineering of the system is not obvious.

The unifying theme among these three perspectives is that the system can use the information about self, environment, and the history of its actions to reason about its future actions, giving different performance for different degrees of self-awareness.

In addition to these general architectural considerations, there are some specific issues for the design of *memory* in self-aware systems. The first issue is the question of what information to acquire. It is pretty clear that information acquisition in humans is highly selective, goal-driven, and based on judgments of relevance. Might it be possible to leverage self-awareness and self-models to implement these properties in computer memory systems? For instance, a system might decide what to learn based on its assessment of current needs.

The next issue concerns retrieval. A great deal of work needs to be done to understand different content-based organizational schemes (episodic memory based on individual narrative experience, semantic memory based on conceptual relations) to allow efficient retrieval of related information. One research question is how a self-model might be useful as an organizing framework for, or play some other role in, structuring memory.

Finally, there is the question of forgetting. Assuming limited storage capacity and a time-cost to retrieval that increases with KB size, some information is going to need to be deleted sometimes. A self-aware system should be able to make intelligent *choices* about what to retain and what to discard.

Challenge Problems

Challenge problems identify specific measurable tasks that can be used to drive the research and to measure progress. Here are some example problems that will require a system to have self-awareness.

Expertise identification. A self-aware system is capable of reasoning about its own knowledge, and about the knowledge of others. Such ability can be of great practical interest in locating expertise on specific topics.

For example, consider the following question:

Did anyone use radar to identify landmarks in the DARPA vehicle challenge problem? A self-aware system may reason with it as follows: (1) I do not know, (2) I need to find someone who does know, (3) Sebastian Thrun may know or may know someone who does know.

An example of a practical system where such queries will make lot of sense is the LinkedIn network (http://www.linkedin.com), which is a social network where people advertise their interests and capabilities as well as who their contacts are.

Conversation. Extended human conversation seems to require representation of one's intentions, of the expected effects of one's utterances, and the state of one's conversational partner, of one's own comprehension of their utterances, and so on. It also seems to involve modeling of the self as a social agent, that is, an agent for whom there are expectations, such as conversational norms and etiquette. This is especially true in areas where success depends on establishing an enduring relationship with human participants, for example, medical advisers for long-term medical care, artificial pets, personal assistants, and 'companions'.

Kriegspiel Chess. Kriegspiel Chess is a variant of chess that requires participants to reason about what they do and do not know, and what their opponent does and does not know. The game requires three boards—one for each player and one for a referee. The main idea of Kriegspiel is that players see only their own pieces, and do not know exactly what moves the opponent has made—they have only some partial information (see below) that allows them to guess where the opponent's pieces are. Only the referee knows exactly the real position of both sets of pieces.

Players move turnwise, just as in normal chess. At each turn, a player attempts a move. When this move is legal, the referee announces that the player has moved, and the turn is done. When the move is not legal, the referee announces that the player attempted an illegal move, and the player must make a new attempt to move, until he makes a legal move. All announcements by the referee are heard by both players.

When a piece captures another piece, the referee announces this, and also the field where the capture has taken place. For instance, the referee could announce: *White has captured on d3*. The referee announces neither the type of piece captured, nor the piece executing the capture. There is one exception to this rule, namely en-passant capture, in which a pawn takes another pawn. In this case, the referee would announce, for instance: *Black has taken en-passant on f3*.

To avoid players having to make long series of wrong guesses about pawn captures each turn, a player may ask: *Are there any pawn captures?* The referee either answers *No*, if the player cannot capture a piece with a pawn, or *Try!*, if there are one or more possible capturing moves with a pawn. In the latter case, the player must make at least one attempt to capture with a pawn (if unsuccessful, the player may continue such attempts or attempt other moves at will).

If a player makes moves that he knows are illegal (for instance, asking about pawn captures when he has no pawns left), the referee says *Impossible*, so that the opponent is not confused by this.

When a move gives check, the referee announces this, and also announces the direction in which check is given: either on the row, on the column, on the small diagonal, on the large diagonal, or by a knight. However, the location of the checking piece is not announced.

Kriegspiel chess is a focused challenge problem for reasoning about knowledge and lack of knowledge that can serve a role that traditional chess did in early days of AI.

Semantic Web services. These services can retrieve information from World Wide Web sources when they need to, and use this knowledge to plan and execute actions that contribute to goal achievement. Self-awareness is required in such services because the knowledge that is available in a web source can be about the availability of other knowledge in a different location. For example, one source (e.g., Yahoo!'s website) can tell the system that it can find out more about computers if it goes to manufacturers' websites. In turn, those can refer the system to another source, depending on the information that is needed more specifically (e.g., technical or purchase information). The system needs to be aware of its own knowledge, the effects of receiving and keeping new knowledge, and how changes to its knowledge should be retracted if it turns out that the new knowledge is inaccurate or false.

Automated tutoring systems. Such systems display a sequence of lessons to a student, and adjust their teaching level and questions/answers to the student level, as perceived by the system. The conversation between the human and the system is not always in natural language, but instead could use different modalities, such as sounds and graphical interfaces. The perception of the system and its decision on its actions depend to a large extent on the state the user is in, and the student's state depends to a large extent on its perception of the system's state. Thus, the system should take into account its own internal state, the state of its knowledge, and its belief about the state of its knowledge as perceived by the user.

Poker playing. In poker (e.g., Texas Hold'em) there are multiple players who need to bid given their observations of their own cards and other players' bids. It is important to maintain a view of the likely cards held by the opponents given their bids, as well as the system's perception of its opponents' beliefs about its own cards. The level to which the system can be explicit about its beliefs, its opponents beliefs about its beliefs, and its beliefs about its opponents beliefs about its beliefs is crucial to success in this game. For example, 'bluffing' in its simplest form is the act of making your opponents believe you have stronger (or weaker) cards than you do. If you believe that your opponents are undecided about the strength of your cards, then bluffing can be a good idea. It is a bad idea to bluff if you believe that your opponents are certain about the strength of your cards.

Summary

The vision of a completely general-purpose theory and architecture for self-aware systems is certainly not yet the state of the art. It is, however, an excellent long-term vision in that it idealizes a strong thread of ongoing activity that is of both theoretical and practical interest.

Machines do not need to be self-aware in the same way as humans do, but some forms of self-awareness seem to be useful, for example, the ability to determine what a system knows and does not know about what it can do and cannot do, and how it can be driven over a period of time in a way that is consistent with its goals. Self-awareness can make the system more robust and self-repairing over a period of time.

There have been some implementations of self-aware systems that have explicit representations of the self and keep a history of its beliefs, actions, and interactions for use both in everyday reasoning and planning and in diagnosing and fixing failures.

Research is needed to explore the forms of knowledge representation necessary to support self-aware computer systems, and to better understand the role(s) that self-awareness can play in practical applications.

Several simple applications of self-awareness to specific tasks are achievable in the near future, but some additional work is necessary to define a focused challenge problem that can carry the field forward in a more substantive way. Areas that might provide materials for the definition of a challenge problem include representing and reasoning about an agent's knowledge (as is required by poker, Kriegspiel chess, and expert identification), and conversational systems and other systems that interact with humans in a social way, such as companions and automated tutors.

Acknowledgments

This material is based on work supported by the Defense Advanced Research Projects Agency (DARPA) under Contract No. NBCHD030010, and FA87500520286. Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of DARPA, or the Department of Interior-National Business Center (DOI-NBC).

References

Anderson, M. L. and Perlis, D. R. 2005. Logic, self-awareness and self-improvement: The metacognitive loop and the problem of brittleness. *Journal of Logic and Computation*, 15(1): 21-40.

Baars, B. J. 1988. *A Cognitive Theory of Consciousness*. Cambridge University Press.

McCarthy, J. 1996. Making Robots Conscious of their Mental States. In S. Muggleton (Ed.), *Machine Intelligence 15*. Oxford University Press.

McCarthy, J. 2004. Notes on Self-Awareness. www-formal.stanford.edu/jmc/selfaware/selfaware.html

Nelson, T. O. 1992. *Metacognition: Core readings*. Boston: Allyn & Bacon.

Rochat, P. 2003. Five levels of self-awareness as they unfold early in life. *Consciousness and Cognition* 12: 717–731.

Shapiro, S. C. 1989. The CASSIE projects: An approach to natural language competence. In: J. P. Martins and E. M. Morgado, editors, *EPIA 89: 4th Portuguese Conference on Artificial Intelligence Proceedings, Lecture Notes in Artificial Intelligence* 390: 362–80.

Witbrock, M., Baxter, D., Curtis, J. et al. 2003. An interactive dialogue system for knowledge acquisition in Cyc. In: *Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence*.