What kind of self-aware systems does the Grid need?

Miranda Mowbray, Alexandre Bronstein
Application Systems Department
HP Laboratories Bristol
HPL-2002-266(R.1)
February 28, 2005*

self-aware, grid

In this paper we draw on computer science, biology, psychology, and philosophy to examine what sort of self-awareness the Grid needs. Several influential Grid researchers have said that the Grid requires self-aware systems, without saying explicitly what "self-aware" means. We classify definitions of self-awareness from philosophy, cognitive science and computer science into two families. The first family is concerned with the external behaviour of the system, the second with its internal processes and structure. The commonest computer science definition is in the second family, and involves reflection - the ability of a program to manipulate as data something representing the state of the program during its own execution. We argue, with examples, that reflection is not necessary for the Grid: Grid systems should use self-awareness properties based on the external behaviour of the system, or no self-awareness at all, depending on the task.

* Internal Accession Date Only
Approved for External Publication

What kind of self-aware systems does the Grid need?

Miranda Mowbray and Alexandre Bronstein, HP Labs

Abstract

In this paper we draw on computer science, biology, psychology, and philosophy to examine what sort of self-awareness the Grid needs. The vision of the Grid is of an enormously large-scale heterogeneous distributed computing system that enables the transparent sharing of geographically distributed resources, including hardware, software, and data.

Several influential Grid researchers have said that the Grid requires self-aware systems, without being explicit about what "self-aware" means. We will identify properties – automatic resource allocation, self-configuration, robustness, fault handling and remediation, and security - that we believe are necessary for the Grid.

We outline several definitions of self-awareness from philosophy, cognitive science and computer science. These fall into two families; the first family is concerned with the external behaviour of the system, the second with its internal processes and structure. The commonest computer science definition of self-aware systems is in the second family, and involves the property of reflection [Baclawski, 2000] - the ability of a program to manipulate as data something representing the state of the program during its own execution. Some philosophers and cognitive scientists suggest a similar definition [Anderson, 1996]. The main theme of this paper is that reflection is in fact not necessary for the Grid.

We will show, with examples, that it is possible to have systems that exhibit properties that we have identified as being necessary for the Grid, without using reflection. We will argue that using reflection – or indeed, any self-awareness property concerned with the internal processes of the system - in the design of Grid systems might be inefficient. We will conclude that Grid systems should use self-awareness properties based on the external behaviour of the system, or no self-awareness at all, depending on the task, and will discuss the types of tasks for which self-awareness is most appropriate.

Some of this investigation may be useful to philosophers. By looking practically at what properties are needed for a system to operate effectively within a complex, dynamic environment, without assuming that this system is biological, we may indicate an interesting avenue of exploration, investigating whether living creatures – and in particular humans – themselves have the properties that we identify. Moreover, if we show that some properties of self-aware living creatures are not necessary for the Grid, this raises the question why these properties arise in living creatures. Finally, our analysis that self-awareness is useful for some tasks but should be avoided for others may provide confirmation for some observations in cognitive science.

The Grid

In December 2000, Laurence Smarr of the California Institute of Telecommunications and Information Technology gave a talk predicting that the Internet would evolve into the "information grid", a planetary-scale computer made from a vast number of interconnecting processors. He said, "The real question, from a software point of view, is: Will it become self-aware?" [Markoff, 2000].

Since then the plans for the Grid have become more concrete. Some influential Grid researchers have said that the Grid not only will become self-aware, but will require self-aware systems in order to achieve its aims [Newman, 1999; Astfalk, 2002] – without however being explicit about precisely what "self-aware" means.

The Grid's purpose will be to enable the transparent sharing of geographically distributed computing resources, including hardware, software, and data. Not only will the resources be geographically distributed, they will be heterogeneous, they may have different owners, different access and use policies, different loads, and different reliability, and all of these may alter dynamically. Moreover resources may become temporarily or permanently unavailable through faults, through policy changes, or because the resource capacity limit is reached.

It will not be practical for human users to specify goals for Grid computations at a detailed level. So goals will be specified at a higher level (e.g. "run this task within this timescale with this level of security at the cheapest cost found") and the system will work out, for example, which geographically distributed hardware and software to use. As a result of the dynamic changes within the Grid this decision might change while the task is being carried out, leading to an automatic reallocation of resources.

For a system of the size envisioned, central control is utterly impractical, and so is complete knowledge in one place (or in one human brain) of the current state or performance of the Grid. Therefore the Grid needs to be capable of being built, reconfigured and managed automatically. Grid systems and processes need to be self-configuring. There need to be local mechanisms for monitoring changes in local system performance and changes in the environment, and the systems must adapt automatically to these changes. Grid systems need to be able to effect some self-repair in the case of damage or process failures. In the cases where self-repair does not solve the problem, self-decommissioning of processes or equipment will be needed.

Security is a vital issue, as the Grid will enable resource sharing across organizational boundaries.

In summary, the Grid needs distributed control algorithms implementing automatic resource (re)allocation, self-(re)configuration, robustness to changes during operation, fault detection and remediation, and security.

Human autonomic systems, and why they are relevant

The Grid needs to be able to run distributed computations without detailed explicit instructions from human users: the Grid processes work out the detail themselves. If the users of the Grid are considered as part of the system, this means that the Grid is

precisely *not* fully self-aware - indeed the Director of the UK e-Science program has described the Grid as requiring "vegetative computing" [Hey, 2002]. The adjective "self-aware" reflects not awareness by the human users but awareness by the Grid software and middleware.

Most of the biological systems that control the operation of the human body do so without human awareness. These include the systems that control heart rates, monitor blood acidity, protect the body from disease, and so on. These systems involve sensing, and reacting on, conditions within the body and sense information coming from the environment, but they do not produce sensations.

Several computer scientists, particularly within IBM [IBM, 2002] have identified these human autonomic systems as a useful metaphor for the design of distributed computer algorithms. Indeed, autonomic systems need several of the same properties that a Grid system requires. They need to operate without conscious involvement. They need to be able to adjust automatically to changes in the environment - for example, the iris adjusts automatically to light levels. Since they cannot rely on conscious intervention to repair faults, they need to be fault-tolerant (several human biological systems achieve fault tolerance by overprovision and parallelism) and if possible self-repairing. Finally they need to be secure from attacks by dangerous pathogens.

Definitions of self-awareness

This section is concerned with existing definitions of self-awareness. Unfortunately, there are considerable differences of opinion as to what "self-awareness" is or how it arises. These differences exist not only between the different disciplines of computer science, philosophy and cognitive science, but also within these disciplines. Since there are so many definitions we will not attempt to produce a comprehensive list of them: we will just outline some that we find particularly interesting or pertinent.

Definitions from philosophy and cognitive science

Some philosophers and cognitive scientists suggest that all information processing may give rise to awareness - so that any computing system, indeed any thermostat, can have subjective experience [Chalmers, 1995]. More discriminatory definitions of self-awareness lie in two families.

The **first family** of definitions emphasises the externally observable behaviour of the self-aware system. An advantage of this is that they that can be applied on an object whose construction is unknown. An approach that fits into the first family is to use a definition of self-awareness similar to the Turing Test definition of artificial intelligence: a system is self-aware if it exhibits behaviour which, if exhibited by a living creature, would lead us to think that the creature was self-aware [Baker, 1997]. Under this approach, a system that gave pre-set, purely syntactic responses to requests could count as self-aware if the responses convinced human testers.

The Turing Test approach requires an idea of what constitutes self-awareness in creatures. To what extent are living creatures self-aware? Minsky [1982] and others question whether humans are self-aware, on the grounds that humans are not aware of much that goes on in their own minds. Two tests for self-awareness in living creatures

that appear to have wide acceptance are Gallup's test [Gallup, 1970], and the false belief test [Wimmer & Perner, 1983], which in one version is called the "Sally-Anne test".

Gallup's test detects whether an animal can recognize itself in a mirror. Most species of animals fail this test, as do humans under 18 months. An analogy for computer systems is the ability to discriminate self from non-self, or normal from anomalous behaviour. Computer systems with this ability [Forrest et al, 1994; Maxion & Tan, 2002] can use it to detect security violations or system faults.

In the Sally-Anne test, the person being tested watches a puppet show in which puppet Sally puts a marble into a basket and leaves the stage, and then puppet Anne moves the marble from the basket into a box. Sally returns, and the person being tested is asked where Sally will look for her marble. Most young (human) children and autistic adults indicate the box, and so fail the test. The ability to pass the false belief test is perhaps better described as "other- awareness" rather than "self-awareness". In the case of the Grid, it is not clear whether there is a belief-holding "other" of which the Grid needs to be aware – although it is interesting to consider what this might be. One possibility is that Grid processes might compensate in some cases for a Grid user's demonstrably false belief. Another is that Grid security processes might try to "fool" security-attacking software by taking advantage of facts known to the security processes but not the attacking software. At a lower level of granularity, one Grid system or process might compensate for wrong actions of another Grid system or process caused by lack of information. However these applications could all be implemented without giving Grid systems any general capacity to reason about others' beliefs. Throughout this paper we will show that using self-awareness (of any type) is just one approach to designing Grid systems with the required properties, but does not preclude non self-aware solutions.

Common "pop" psychology defines self-awareness as the ability to observe and reflect on one's complete behaviour [Keyes & Keyes, 1975].

Jolly [2002] defines a self-aware organism as one that can modify its behaviour based on inputs from its environment. Which brings us back to the self-aware thermostat.

The **second family** of definitions and approaches emphasises the system's inner processes, representations, and structure. Some philosophers believe that self-awareness may somehow emerge in a system with sufficiently complex internal structure [MacLennan, 1997; Chalmers, 1996].

Cole [1998] describes a hypothetical self-aware machine as "a machine capable of treating sensory information as states of itself." He gives as an example of sense without sentience (and hence, for him, without self-awareness) a modern car engine control system, which senses temperatures, pressure, oxygen levels and so on and acts on this to minimize pollution and maximize efficiency. Human autonomic systems also have sense without sentience.

Damasio [1999] argues that self-awareness is caused by the representation within an organism's brain of the continuity of the organism and its interactions with external

objects. The definition of reflection for computer systems also relies on the system's ability to reify its own current state.

For Nichols and Stich, self-awareness is the ability to attribute mental states to oneself. In [Nichols & Stich, 2002] they suggest that human self-awareness is achieved by a special monitoring mechanism that takes a mental representation of a belief *p* and produces a representation "I believe that *p*". Smart [2000] discusses the example of a robot that could scan its own perceptual processes, and thus would be aware of its own awareness. Along similar lines, but requiring more than monitoring, the members of the PT-Project at Illinois State University define self-awareness as the ability not just to access one's own mental states, but also to critically reflect on these states and make judgements about them [Anderson, 1996].

Definitions from Computer Science

An online job advertisement described a vacancy in July 2002 for a junior Java ® professional for whom the duties were "development of distributed, intelligent and self-aware systems". Unless Science Fiction is becoming true even faster than usual, the words "intelligent" and "self-aware" are being used here in a way that is different from their use in everyday conversation, and also from the way many philosophers use them.

In fact, many computer scientists agree that the definition of self-awareness for software and middleware is related to reflection. Reflection is the ability of the program to manipulate as data something representing the state of the program during its own execution [Bobrow et al., 1993]. This is similar to the cognitive science definition given by the PT-Project. For some computer scientists, in particular Java programmers, self-awareness just means reflection. For others, in addition to reflection a self-aware system has to have a representation of its own purpose [Baclawski, 2000]. Different branches of Computer Science have, however, interpreted the details of reflection differently: for a good comparative study, see [Demers & Malenfant, 1995].

An example of a reflective architecture designed for the Grid is the Java-based PAGIS architecture for building geographical information systems [Webb & Wendelborn, 2002]. The functional concerns of the geographical information systems are dealt with by the base-level program, and meta-level programs deal with issues of implementation strategy such as locality, adaptation and performance monitoring. One meta-level program maps processes and channels to physical resources using the Globus Grid middleware.

Another form of self-awareness focuses solely on the behaviour of a computing element, rather than on its internal state or structure. This corresponds to the first family of definitions from philosophy and cognitive science. Management software for example could use this extensional view to assess the health of 'black box' entities, just like a psychologist observes the behaviour of patients in order to assess (and eventually improve) their mental health.

This leads us to the following classification of self-awareness approaches seen in system design, approximately corresponding to the second and first families of philosophical approaches, and the approach of not using meta reasoning at all.

A Classification

Reflective self-awareness	Observation and computation over the structure and run-time state of computing element/system. Logically, an intensional approach. (Java reflection is the archetype.)
Objective self-awareness	Observation and computation over the externally visible behaviour of a computing element/system. Logically, an extensional approach.
No self-awareness	Neither of the above. Logically, no meta reasoning.

Examples

In the section on the Grid we identified five task areas for which the Grid needs distributed control algorithms. We will now give examples of control algorithms within each of these areas that do not use reflection. These examples show that important problems in each of the areas that might use self-awareness can be solved without reflection. It is unproven that reflection is needed for the Grid.

Automatic resource allocation: ant-based algorithms

Foraging ants leave pheromone trails for other ants indicating the direction of good food resources. Several researchers have developed computer algorithms inspired by this behaviour, originally for stochastic optimisation [Dorigo et al, 1996] and then for load balancing in networks [Schoonderwoerd et al, 1996].

Andrzejak et al [2002] have built on this work to produce an ant-based algorithm that determines where to host a particular service within a Utility Data Centre linked to the Grid. The service may be new, or may need to be moved from an overloaded server. In their algorithm, a software “ant” associated with the service travels from server to server on a probabilistic walk with a fixed maximum number of steps. At each server, the ant evaluates how suitable the server is to host the service, based on the resource requirements for the service and the suitability of neighbouring servers to host services with which the new service needs to cooperate. The ant updates a data structure on the server and its neighbouring servers with a score for suitability as host for the service itself and for cooperating services. The ant then moves to a new server, with its choice of the next server influenced by scores left by previous ants. At the end of the walk, the ant sends a message to the managing agent of the highest-scoring server found on the walk, telling it to install the service.

This algorithm achieves automatic allocation of resources without using self-awareness. It may not find optimal solutions, but is robust both to changes in the environment and to failures of individual ants. The same goes for the foraging strategy used by biological ants.

Self-configuration: SmartFrog configuration management environment

HP's SmartFrog ("Smart Framework for Object Groups") technology [Goldsack, 2001] is a system description and deployment framework. SmartFrog provides a declarative language for describing the “configuration” of complex, distributed systems as sets of related components. Configurations are instantiated by a runtime

deployment environment, which ensures that the components are correctly located, configured, sequenced and bound to one another. The runtime environment allows for inspection of the set of running components, and runtime configuration control such as addition, deletion and reconfiguration of components. This can be thought of as a form of reflection at the granularity of the component, however at the granularity of individual code instructions (which is the granularity at which reflection is usually considered) the environment is not reflective.

Robust, Fault-Tolerant Operation: Two-phase commit protocol

The two-phase commit protocol is a well-known solution getting several distributed machines to agree to do something or not, atomically. For a description of the protocol, see [Vin, 2001].

It is fundamental in database systems [Garcia Molina, Ullman, & Widom, 2001] and quite robust, and is used in large-scale real distributed systems, for example banking systems. Yet it has no self-awareness whatsoever: the processes involved simply follow an agreed-upon messaging script and low-level logging and state changing actions. The correctness proof for the protocol involves some meta level reasoning, but the algorithm itself does not. Nevertheless it achieves robust distributed coordination with some fault-tolerance.

Fault detection and remediation: self-aware services

The self-aware services approach [Bronstein et al., 2001] intentionally attempts to equip a computing element with machinery that observes and reflects on its extensional behaviour. This is implemented by adding a simple probabilistic reasoning engine (Bayesian network) over an adaptive measurement gathering and evaluating architecture. The purpose is to perform automatically the health assessment task that is usually done by human system administrators; and secondly, to communicate that health assessment to any other interested service element, including possibly the element itself, in order to facilitate fault location and enable fault remediation. The architecture was demonstrated on an email firewall service, but is mostly domain independent. In that work, the authors define their notion of “self-awareness” as the ability of an element to autonomously detect deviations in its behaviour that are meaningful. This corresponds to “objective” self-awareness in our classification.

Security: computer immune systems

The human immune system is a sophisticated fault-tolerant self-adapting repair system. It operates without human awareness. Several research teams [Forrest, 2002; Dasgupta, 1999; Kephart, 1994; Kim & Bentley, 1999] have aimed to design computer security systems based on the human immune system, whether for intrusion detection or for protection from computer viruses. IBM has built a "commercial-grade immune system" along these lines [White et al, 1998]. (However, during its development IBM weakened the similarity of their computer immune system to the human immune system, perhaps for ease of management: see [Williamson, 2002] for a discussion of this.)

Discussion: Self-Awareness to What End?

The question that should be asked when deciding what kind of self-awareness (if any) to build into a system is: self-awareness to what end? From the discussion of the examples above it appears that different tasks (both in computing systems and in humans) have different requirements for self-awareness.

Some tasks work well with no self-awareness. For example, distance vector routing and other non self-aware distributed routing algorithms can deal comfortably and robustly with highly complex, changing environments. Self-awareness, especially reflective self-awareness, is expensive in terms of time and processing power, and therefore should be avoided if possible for routine operations of the system. In fact, some non-routine tasks (such as fault handling) can also be accomplished by non self-aware approaches, as demonstrated by the two-phase commit protocol.

Many of the tasks performed by the human autonomic systems are routine tasks. Humans can also perform routine learned processes semi-unconsciously, even very high level processes. Golfers practice their swings until they are no longer aware of their technique – and are thus less likely to interfere with it. Drivers can discover that their car is driving them home although they meant to go somewhere else: although their processes for reacting to traffic are (we hope!) operating in a self-aware fashion, their larger-scale navigation is not. A crisp analysis of when self-awareness is useful and when it is not, in a particularly high level task such as human combat, was given several centuries ago by the Zen master Takuan Soho in "The Unfettered Mind" [Soho, 1986]: self-awareness is a stepping stool for acquiring skills and techniques, i.e. improving the self, which must then be disposed of in order to perform masterfully.

The length of time between changes in a computer system is critical to the amount of model information that it is sensible to use for a task that reacts to these changes. In the case of fault handling and repair tasks where there is a relatively long time between changes, the extra burden of self-awareness can be justified.

Conclusion

We conclude that the Grid does not need reflective self-awareness. It does need distributed control algorithms, but these should either use objective self-awareness, or no self-awareness at all, depending on the stage in the lifecycle of the controlled process that the control task falls into. For resource allocation and configuration

(birth) and on-going operation the processes can be efficiently handled without using self-awareness, but self-awareness is appropriate for fault detection and remediation. If remediation fails or is not a possibility, shutting down (death) and upgrade (reincarnation?) can be viewed as a special case of fault handling, and so might benefit from self-awareness at the appropriate level of granularity. Until Grid systems get into the business of learning new skills and techniques on their own, the tasks where self-awareness seems most appropriate are fault handling and repair.

To end this paper, here are some questions for philosophers and biologists. If reflective self-awareness is not necessary - and may be inefficient - for Grid systems, what purpose does it serve for humans that is not relevant to the Grid? (Assuming, of course, that we humans *are* reflectively self-aware.) The types of task for which self-awareness is most appropriate appear at first glance to be similar for Grid systems (using our classification) and for humans (according to some of the definitions from philosophy). Is there a general result relating to the amount of information about the self that it is sensible to use for a particular type of task? Finally, are there other notions of self-awareness in philosophy that could be useful for the design of the Grid in particular, and computer science in general?

Acknowledgement

Thanks to everyone whose ideas and feedback we have used here. Thanks especially to Marsha Duro, whose suggestions significantly improved the paper.

BIBLIOGRAPHY

Anderson, D. L., (1996) “What Is Self-Awareness?” In Building an Artificial Person, PT-Project online paper, Illinois State University. Online: http://www.ptproject.ilstu.edu/prsnbld1.htm

Andrzejak, A., Graupner, S., Kotov, V. and Trinks, H. (2002) Self Organizing Control in Planetary-Scale Computing. IEEE International Symposium on Cluster Computing and the Grid (CCGrid), 2nd Workshop on Agent-based Cluster and Grid Computing (ACGC), May 21-24, 2002, Berlin. Online: http://www.inf.ethz.ch/personal/andrzeja/my-papers/CCGrid-2002-subm.pdf

Astfalk, G. (2002) Why Grids? Singapore Grid Symposium, 2002, slide 16.

Baclawski, K. (2000) Overview – Ontology and Abstract Model for Real-Time Distributed Resource Management. Self Controlling Software Group, Northeastern University. Online: http://www.ece.neu.edu/groups/scs/overview_documents/ontology.html

Baker, M. (1997) "Re: Semantic Frameworks, Sociological Impact, and More (fwd)" on *dist-obj* list, 3 Mar 1997. Online: http://www.distributedcoalition.org/mailing_lists/dist-obj/msg00087.html

Bobrow, D.G., Gabriel, R.P and White, J.L. (1993) “CLOS in Context – The Shape of the Design Space.” In A. Paepeke, ed., Object Oriented Programming – The CLOS Perspective, ch. 2. MIT Press.

Bronstein, A., Cohen, I., Das, J., Duro, M., Friedrich, R., Kleyner, G., Mueller, M., and Singhal, S. (2001). Self-Aware Services: Using Bayesian Networks for Detecting Anomalies in Internet-Based Services. (Report No. HPL-2001-23-R1). Palo Alto, California, USA: Hewlett-Packard Labs (IMSL). Online: http://www.hpl.hp.com/techreports/2001/HPL-2001-23R1.html

Chalmers, D. J. (1995) "Facing Up to the Problem of Consciousness", Journal of Consciousness Studies. Online: http://www.u.arizona.edu/~chalmers/papers/facing.html

Chalmers, D. J. (1996) The Conscious Mind: In Search of a Fundamental Theory, p.248. Oxford University Press.

Cole, D. (1998) Sense and Sentience, Jan 1998. Online: http://www.d.umn.edu/~dcole/sense5.html

Damasio, A. R. (1999) The Feeling of What Happens: Body and Emotion in the Making of Consciousness. Harcourt Brace.

Dasgupta, D. (1999) “Immunity-Based Intrusion Detection System: A General Framework.” In Proc. 22nd National Information Systems Security Conference (NISSC), October 18-21, 1999.

Demers, F-N., and Malenfant, J. (1995) “Reflection in Logic, Functional and Object-Oriented Programming: a Short Comparative Study”. In Workshop of IJCAI'95 : On Reflection and Meta-Level Architecture and their Application in AI, pages 29-38, August 1995. Online: http://citeseer.nj.nec.com/106401.html

Dorigo, M., Maniezzo, V., and Colorni, A. (1996) “The Ant System: Optimization by a Colony of Cooperating Agents.” IEEE Transactions on Systems, Man and Cybernetics, Part B, 26(1):1-13, 1996.

Forrest, S., (2002) Computer Immune Systems. Online: http://www.cs.unm.edu/~immsec/research.htm

Forrest, S., Perelson, A.S., Allen, L., and Cherukuri, R. (1994) “Self-Nonself Discrimination in a Computer.” In Proc. 1994 IEEE Symposium on Research in Security and Pricacy, 120-128, IEEE Computer Society Press.

Gallup, G. G., Jr. (1970) “Chimpanzees: Self-recognition.” Science 167: 86-87.

Garcia-Molina, H., Ullman, J. D., & Widom, J. D. (2001). Database Systems: The Complete Book. Prentice Hall. ISBN 01-303-19-953.

Goldsack, P (2001) “SmartFrog: A Framework For Configuration”. Large Scale System Configuration Workshop, National e-Science Centre, Edinburgh, November 2001.

Hey, A. (2002) Talk on the UK Grid, HP Labs Bristol, 10 July 2002

IBM (2002) Autonomic Computing. Online:
http://researchweb.watson.ibm.com/autonomic/

Jolly, T. (2002) E-mail exchange between Tom Jolly and Curtis Anderson, reported on the letters page of The Games Journal, Jan 2002. Online:
http://www.thegamesjournal.com/letters/Jan2002.shtml

Kephart, J. O. (1994) "A Biologically Inspired Immune System for Computers". In Artificial Life IV, Proceedings of the Fourth International Workshop on Synthesis and Simulation of Living Systems, Rodney A. Brooks and Pattie Maes, eds., MIT Press, Cambridge, Massachusetts, 1994, pp. 130-139. Online:
http://www.research.ibm.com/antivirus/SciPapers/Kephart/ALIFE4/alife4.distrib.html

Keyes, K., & Keyes, K. Jr. (1975). Handbook to Higher Consciousness. Love Line Books. ISBN 0960068880.

Kim, J., & Bentley, P. (1999). "The Human Immune System and Network Intrusion Detection", 1999. Online, via http://citeseer.nj.nec.com/kim99human.html

Maclennan, B. (1997) "The Elements of Consciousness and their Neurodynamical Correlates". In Explaining Consciousness: The Hard Problem, ed. Jonathan Shear, MIT Press. Online: http://www.cs.utk.edu/~mclennan/ElConsc/JCS2.html

Markoff, J. (2000) The Soul of the Ultimate Machine, December 10, 2000. Online:
http://www.racematters.org/selfawarecomputer.htm

Maxion, Roy A. & Tan, Kymie M. C. "Anomaly Detection in Embedded Systems." IEEE Transactions on Computers, 51(2) 108-120, February 2002.

Minsky, M. (1982) "Why People Think Computers Can't". AI Magazine, vol. 3 no. 4, Fall 1982. Online:
http://www.ai.mit.edu/people/minsky/papers/ComputersCantThink.txt

Newman, H. (1999) MONARC Status, and Data Analysis Grid Projects. SCTB Meeting, CERN June 14, 1999. Online:
http://l3www.cern.ch/~newman/scbjune99_monarc/tsld032.htm

Nichols, S. & Stich, S. (2002) "How to Read Your Own Mind: A Cognitive Theory of Self-Consciousness." In Consciousness: New Philosophical Essays, eds. Q. Smith and A. Jokic. Oxford University Press. Early version online:
http://www.umsl.edu/~philo/Mind_Seminar/New%20Pages/papers/Stich/Self-awareness.html

Schoonderwoerd, R., Holland, O., Bruten, J., and Rothkrantz, L. (1996) "Ants for Load Balancing in Telecommunications Networks", Adaptive Behaviour 2:169-207

Smart, J.C.C. (2000) "The Identity Theory of Mind", in Stanford Encyclopedia of Philosophy. Online: http://plato.stanford.edu/entries/mind-identity/

Soho, T. (1986). The Unfettered Mind - Writings of the Zen Master to the Sword Master. Tokyo: Kodansha International. ISBN 4-7700-1276-4.

Vin, H. (2001) Reliability and Two-Phase Commit Protocol. Online: http://www.cs.utexas.edu/~vin/Classes/CS372Fall01/Slides/20.TwoPhaseCommit.pdf.

Webb, D.L. & Wendelborn, A.L (2002), Campus Grid and PAGIS Application Builder. Online: http://www.cs.adelaide.edu.au/users/darren/files/demogrid.pdf

White, S.R., Swimmer, M., Pring, E.J., Arnold, W.C., Chess, D.M., Morar, J.F. (1998) Anatomy of a Commercial-Grade Immune System. Online: http://www.research.ibm.com/antivirus/SciPapers/White/Anatomy/anatomy.html

Williamson, M. M. (2002) Biologically Inspired Approaches to Computer Security. Technical Report HPL-2002-131, Hewlett Packard Laboratories. Online: http://lib.hpl.hp.com/techpubs/2002/HPL-2002-131.html

Wimmer, H. & Perner, J. (1983). “Beliefs about Beliefs: Representation and Constraining Function of Wrong Beliefs in Young Children's Understanding of Deception”. Cognition, 13, 103-128.