Oracle: An Agent-Based, Reference Architecture

Henry Hexmoor* and Jody Little~
*Computer Science and Computer Engineering Department
Engineering Hall, Room 340A
University of Arkansas
Fayetteville, AR 72701
~ Sierra Nevada Corporation
4801 NW Loop, 410
San Antonio, Tx 78205
{hexmoor@uark.edu, jody.little@sncorp.com}

Abstract
This paper introduces a reference architecture for developing agent-based systems that preserves core concepts of agenthood while minimizing cumbersome features found in other agent architectures. This parsimony has been found useful in addressing complex problems.

1. Introduction

With the proliferation of agent-based systems in every type of complex, mission critical system there is a need for a reference architecture that capitulates essential and salient ingredients of agent-based systems. This reference architecture can be used as a guide to organization and design of specific agent-based systems (Gazi, et. al., 2001). In this paper we primarily focus on the salient components and interfaces of such an architecture as we aim to offer a reference architecture in the spirit of James Albus' RCS. A reference architecture is not a blow by blow specification but rather an educated guide for conceptualization. Specific methodologies and operational notions that elaborate our reference architecture will vary among classes of problems. To illustrate this, we briefly outline two case studies in the latter part of this paper.

We take an Agent oriented software engineering approach in exposition of our reference architecture. Agent-oriented software engineering is a proper subset of software engineering (Wooldridge, 2000) with emphasis placed on interactions and relationships among entities we'll call agents. To make explicit agents we offer five guiding principles. First, agents are entities in the problem domain as opposed to models of functional abstractions. More specifically, agents encapsulate computational units that determine plans and actions as well as the process of exhibiting acting. Second, agents are properly sized so they model entities that are rather modest in mass and time. We are not suggesting to model midgets. Rather, we are arguing to consider acting units in such a way that the system being modeled will map to a finite number of agents in order to allow us to meaningfully focus on modeling interesting interactions and relationships. If the agent granules are fairly large we would not have the opportunity to examine intricacies of interagent interface. Since an aim in designing agent-based systems is distributed intentionality, our third principle is that agents should be considered to own their local intentionality. System intentionality should be properly divided into a series of proper sets so they can be mapped to intentionality of agent communities and the smallest units map to individual agent intentionality. Our fourth agent design principle is coherent dissemination of information, knowledge, and wisdom. Agents should be provided with methods for caching and sharing results of individually (i.e., locally) processed and fused data. Properly designed dissemination methods will offer cohesion so that the set of agents will act as a whole, i.e., a hive-mind. Although it is beyond the scope of this article, we need to point out the need for shared or disparate ontologies among groups of agents (NCOR). A more elaborated consideration needs to account for delineate ontology mediation by agents themselves or designated agents. Our fifth principle suggests to consider agents operating in sufficient independence as if they operate in parallel. Despite the obvious need for interdepedendence among agents in any complex system and hence models of relationships and interactions suggested herein, agents need to be designed to operate concurrently as opposed to sequentially. The large number of agent architectures in existence does not imply maturity in the discipline. Instead, it reflects a rush to capture and document features of interest. In the following section we will review agent architectures and propose a need for parsimony and a return to original conceptions of modeling agents.

2. Agent architectures

One of the earliest and the most influential agent architectures is the BDI paradigm. The Stanford group of researchers in mid-1980s suggested capturing mentalistic notions such as belief, desire, intention (Bratman, 1987, Muller, 1997, Rao and Georgeff, 1998). There was a heavy leaning toward grounding BDI in formal modal logics partly to inherit the properties of soundness and completeness and partly to gain expressive power of treating BDI as modalities. The expressive power gained came at the expense of lack of tractability. Along with many researchers we have implemented a limited form of BDI in our labs with partial satisfaction. The best known implementation is often attributed to Kinney and Georgeff, 1991). BDI shortcomings are well-documented and we will avoid repeating them here. Instead, we point to the need to preserve Bratman's claim that rational agents strive to adopt and maintain conflict-free intentions. All rational agent reasoning will service for avoid detraction from adopted intentions and on methods for manifesting desired objects of intention. We wish to explicate the primacy of *intention* with the need for *attention* as we will see in the following section. A more pragmatic agen t architecture is MaSE (DeLoach, et.al., 2003). Rooted in BDI, DeLoach has not only provided expressive power of modeling roles and communication in MaSE but also provided a blow by blow methodology that was lacking in BDI. Tropos is a recent agent architecture that offers both a methodology and rich expressivity (Giorgini, et.al., 2004).

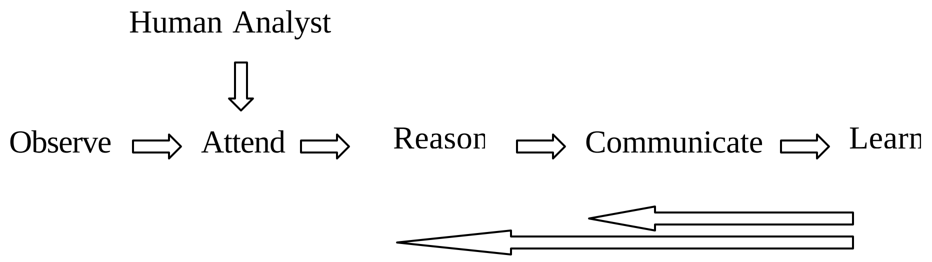

Figure 1. OARCL Components

3. OARCL Reference Architecture

OARCL is comprised of Observe, Attend, Reason, Communicate, and Learn (OARCL) components. Similar to the OODA loop concept proposed by Col. John Boyd (Richards, 2004), it is complex and involves many internal loops and nontrivial processes that supervise and keeps the system inline with various measures of effectiveness and performance. There are abundant asynchronies and nonlinearities (see Figure 1). One of our aims in introducing Oracle is to urge a return to the original conceptions of agents, which were anthropomorphic modeling of a sense of acting. Another aim is to put in the spotlight the salient properties of agency by Oracle components. At a metaphorical level, an OARCL agent is a cognitive entity with functionalities suggested by pre-attentive functions captured in the module Observe, attention generation and maintenance in the module Attend, inferential capabilities in the component Reason, intentional message generation in the module Communicate, and abilities to modify perception, attention, and reasoning processes in the component we call Learn. Next we provide general description for each component.

In order to coarsely filter out irrelevant data, the Observe component performs agent input data distribution management functions. Agent who live in a highly distributed environments need to control the volume and the nature of input they process. We choose to consider the tasks of what kind and how much data to process as a pre-attentive function of an agent. The best example is human visual processing where visual input is rapidly and automatically filtered. Supported by recent findings in human visual processing that suggests *attention* plays an important cueing role even in early, *preattentive* visual stage we have selected that as our next component (Healey, et.al,.1993) .

Attend receives references from the human supervisor, all agent requests from the Observe component as well as from the Communicate module. References might be associated with specific sensory-identified objects or targets or it might be in terms of features, patterns, and conditions. The selection process, which includes nontrivial reasoning is modeled in this component. We will deliberately leave out discussion of the obvious connection between attention to awareness and consciousness (Crick, 1996). Despite this omission, we point out that this connection is the most elusive, intriguing, yet essential character of agency. There is an inextricable relationship between defining characteristics of independence and pro-activity of computational agents and self-awareness encapsulated in attention. The notion of individuality in attention plays a crucial role in guiding as well as controlling inference and logical reasoning. The rationality principles of Allen Newell and Nick Jennings are addressed in our reason module (Boman and Van deVelde, 1997). The *reason* component performs the primary inference in OARCL agents.

Communicate performs external information management including the speech act using standard agent communications language (ACL) for outputs (e.g., requests) and input information and requests from other agents.

Finally, the Learn component generates performance assessment that drives its process management of all agent components. Next, we will discuss applications that lead to development of our reference architecture.

One of the problems that motivated Oracle was the SIGINT man on the loop with the aim to design an agent-based software that aids and automates intelligence analysis, technically known as SIGINT analysis (Hollywood, et.al, 2004). The SIGINT activity encompasses all of command, control, communications, human intelligence, surveillance, and reconnaissance. The second motivating problem was modern disaster response with the aim to design an agent-based software that fuses information at varying levels of abstraction in order to rapidly assess situations at a high level. Our approach preserved the highly distributed and disparate loci of information gathering and synthesis.

Next, we review Oracle modules and briefly discuss the software engineering tasks therein. For Observe one must broadly gather techniques for selecting data sources and channels with details beyond the scope and interest of this paper. Attend for both problems must be flexible to allow for changes in the human analyst's goals and targets. Generically, a human analyst will set and revise conditions and targets of interest for the remainder of the system (shown in Figure 1). The reference points selected in attend will be used to place filters on data gathered by Observe. Reason will need to identify threats and opportunities of interest n the command and control, which is the core function of a typical human analyst.

Communicate will consist of (a) all conditions for triggering messages to other system functions in support of SIGINT as well as disaster response, and (b) types and formats of messages corresponding to triggering conditions.

Finally, Learn will embody metrics and sets of adjustments for internal functions as well as interactions among Oracle modules.

Conclusion

Oracle is both a call to return to original conceptions of agenthood and agent-based research as well as a reference architecture for designing agents in an agent-based system.

Our outline here is very concise and does not offer a detailed methodology. It also does not provide properties found in aspect –oriented paradigm.

Acknowledgements

Prof. Henry Hexmoor wishes to acknowledge support and encouragement of staff at Sierra Nevada Corporation's San Antonio Operations as well as the Center for Multisource Fusion at SUNY Buffalo.

References

1. M. Boman, M. and Van de Velde, W. Multi-Agent Rationality: Proceedings of the 8th European Workshop on Modelling Autonomous Agents in a Multi-Agent World, MAAMAW'97, Ronneby, Sweden, May 13-16, 1997.

2. Bratman.M.E., *Intentions, Plans, and Practical Reason*. Harvard University Press: Cambridge, MA, 1987.

3. Crick, F. Visual perception: rivalry and consciousness. *Nature* 379:485-486, 1996

4. DeLoach, S.A., Matson, E.T., Li, Y.. Exploiting Agent Oriented Software Engineering in the Design of a Cooperative Robotics Search and Rescue System. The International Journal of Pattern Recognition and Artificial Intelligence, 17 (5), pp. 817-835, 2003.

5. Hollywood, J., Snyder, D., McKay, K. N., and Boon, J.E. Out of the Ordinary: Finding Hidden Threats by Analyzing Unusual Behavior, Rand pub, ISBN: 0-8330-3520-7, 2004.

6. Gazi, V., Moore, M.L., , Passino, K.M., Shackleford, W.P., Proctor, F., Albus, J.S. The RCS Handbook: Tools for Real Time Control Systems Software Development, Wiley, 2001.

7. Giorgini, P., Kolp, M., Mylopoulos, J., and Pistore, M. The Tropos Methodology: an overview. In F. Bergenti, M.-P. Gleizes and F. Zambonelli (Eds) Methodologies And Software Engineering For Agent Systems, Kluwer Academic Publishing, 2004.

8. Healey, C. G., Booth, K. S., and Enns, J. T. Harnessing preattentive processes for multivariate data visualization. *Proceedings Graphics Interface '93* (Toronto, Canada, 1993), pp. 107–117, 1993.

9. Kinny, D. and M. Georgeff., M. Commitment and effectiveness of situated agents. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI-91)*, pages 82–88, Sydney, Australia, 1991.

10. Muller, J.P. *The Design of Intelligent Agents (LNAI Volume 1177).* Springer-Verlag: Berlin,
11. Germany, 1997.

12. Rao, A.S. and Georgeff, M. Decision procedures of BDI logics. *Journal of Logic and*

13. *Computation*, 8(3):293–344, 1998.

14. Richards, C. Certain To Win: The Strategy Of John Boyd, Applied To Business, Xlibris Corporation publisher, 2004.