CHAPTER 13

An Overview of Agent-Oriented Programming

Yoav Shoham

I have been working in areas related to software agents for a number of years now, together with many students and other colleagues. Recently, terms such as "(intelligent) (software) agents," "knowbots," and "softbots" have become quite popular. The work taking place under this umbrella is diverse, varying in content, style, and quality sufficiently to render terms such as "software agent" meaningless in general. I have spent a fair amount of time in the past two years trying to understand various agent-related work in industry and academia. However, in this chapter I will not attempt to put any order into this area, nor position our own work at Stanford within it. This is the topic of another paper currently in the works. The discussion here will be confined to reviewing our own work on multi-agent systems in general and agent-oriented programming in particular.

Agent-Oriented Programming: Software with Mental State

In 1989 I coined the term *agent-oriented programming* (AOP) to describe a new programming paradigm, one based on cognitive and societal view of computation. Although new, the proposal was inspired by extensive previous research in Artificial Intelligence (AI), distributed computing, and other neighboring disciplines. This chapter will summarize some of the major ideas from previous research. A more detailed discussion of AOP appears in Shoham (1993).

What Is an Agent?

Most often, when people in AI use the term "agent," they refer to an entity that functions continuously and autonomously in an environment in which other processes take place and other agents exist. This is perhaps the only property that is assumed uniformly by those in AI who use the term. The sense of "autonomy" is not precise, but the term is taken to mean that the agents' activities do not require constant human guidance or intervention. Often certain further assumptions are made about the environment, for example that it is physical and partially unpredictable. In fact, agents are sometimes taken to be robotic agents, in which case other issues such as sensory input, motor control, and time pressure are mentioned.

Finally, agents are often taken to be "high-level." Although this sense is quite vague, many take some version of it to distinguish agents from other software or hardware components. The high level is manifested in symbolic representation and/or some cognitive-like function: agents may be "informable" (Genesereth 1989), may contain symbolic plans in addition to stimulus-response rules (Torrance 1991; Hayes-Roth et al. 1989; Mitchell 1990), and may even possess natural-language capabilities. This sense is *not* assumed uniformly in AI, and in fact a certain counter-ideology deliberately denies the centrality or even existence of high-level representation in agents (Agre and Chapman 1987; Brooks 1986).

Clearly, the notion of agenthood in AI is anything but crisp. I should therefore make it clear what *I* mean by the term "agent," which is precisely this: *An agent is an entity whose state is viewed as consisting of mental components such as beliefs, capabilities, choices, and commitments.* These components are defined in a precise fashion and stand in rough correspondence to their commonsense counterparts. In this view, therefore, agenthood is in the mind of the programmer: What makes any hardware or software component an agent is precisely the fact that one has chosen to analyze and control it in these mental terms.

The question of what an agent is is now replaced by the question of what entities can be viewed as having mental state. The answer is that *anything* can be so described, although it is not always advantageous to do so. This view is not original to me. For example, in Dennett (1987) and other publications, Dennett proposes the "intentional stance," from which systems are ascribed mental qualities such as intentions and free will. The issue, according to Dennett, is not whether a system really is intentional, but whether we can coherently view it as such. Similar sentiments are expressed by McCarthy (1979), who also distinguishes between the legitimacy of ascribing mental qualities to machines and its usefulness:

> To ascribe certain *beliefs, free will, intentions, consciousness, abilities,* or *wants* to a machine or computer program is *legitimate* when such an ascription expresses the same information about the machine that it expresses about a person. It is *useful* when the ascription helps us understand the structure of the machine, its past or future behavior, or how to repair or improve it. It is perhaps never *logically required* even for humans, but expressing reasonably briefly what is actually known about the state of the machine in a particular situation may require mental qualities or qualities isomorphic to them. Theories of belief, knowledge and wanting can be constructed for machines in a simpler setting than for humans, and later applied to humans. Ascription of mental qualities is *most straightforward* for machines of known structure such as thermostats and computer operating systems, but is *most useful* when applied to entities whose structure is very incompletely known.

In Shoham (1989), I illustrate the point through the light-switch example. It is perfectly coherent to treat a light switch as a (very cooperative) agent with the capability of transmitting current at will, who invariably transmits current when it believes that we want it transmitted and not otherwise; flicking the switch is simply our way of communicating our desires. However, while this is a coherent view, it does not buy us anything, since we essentially understand the mechanism sufficiently to have a simpler, mechanistic description of its behavior. In contrast, we do not have equally good knowledge of the operation of complex systems such robots, people, and, arguably, operating systems. In these cases it is often most convenient to employ mental terminology; the application of the concept of "knowledge" to distributed computation, discussed below, is an example of this convenience.[1]

Agent- Versus Object-Oriented Programming

I mentioned previously that the ascription of mental constructs must be coherent and useful. The application of the logic of knowledge in distributed computation, given there as an example, used the mental construct "knowledge" in a particular way: it mapped it onto an existing computational framework (a distributed network of processors) and used it to reason about the system. The use we will make of mental constructs is different: rather than use them for mere analysis, we will employ them to *design* the computational system. The various mental categories will appear in the programming language itself, and the semantics of the programming language will be related to the semantics of the mental constructs. This is similar in spirit to a development within the distributed computation community, where a proposal has been made to include tests for epistemic properties in the protocols themselves (Halpern and Zuck 1987); however, up till now there has been no follow up on the proposal.

I have proposed a computational framework called *agent-oriented programming* (AOP). The name is not accidental, since from the engineering point of view AOP can be viewed as a specialization of the *object-oriented programming* (OOP) paradigm. I mean the latter in the spirit of Hewitt's original Actors formalism (Hewitt 1977), rather than in some of the senses in which it used today.

Intuitively, whereas OOP proposes viewing a computational system as made up of modules that are able to communicate with one another and that have individual ways of handling incoming messages, AOP specializes the framework by fixing the state (now called *mental state*) of the modules (now called *agents*) to consist of precisely defined components called beliefs (including beliefs about

	OOP	AOP
Basic unit	object	agent
Parameters defining state of basic unit	unconstrained	beliefs, commitments, capabilities, choices...
Process of computation	message passing and response methods	message passing and response methods
Types of message	unconstrained	inform, request, offer, promise, decline...
Constraints on methods	none	honesty, consistency...

Figure 1. OOP versus AOP.

the world, about themselves, and about one another), capabilities, choices, and possibly other similar notions. A computation consists of these agents' informing, requesting, offering, accepting, rejecting, competing, and assisting one another. This idea is borrowed directly from the speech act literature (Grice 1989; Searle 1969; Austin 1962).

Speech act theory categorizes speech, distinguishing between informing, requesting, offering and so on; each such type of communicative act involves different presuppositions and has different effects. Speech-act theory has been applied in AI, in natural language research as well as in plan recognition. To my knowledge, AOP and McCarthy's Elephant2000 language are the first attempts to base a programming language in part on speech acts. Figure 1 summarizes the relation between AOP and OOP.[2]

On the Responsible Use of Pseudo-Mental Terminology

The previous discussion referred to mentalistic notions such as belief and commitment. In order to understand the sense in which I intend these, consider the use of logics of knowledge and belief in AI and distributed computation. These logics, which were imported directly from analytic philosophy first to AI and then to other areas of computer science, describe the behavior of machines in terms of notions such as knowledge and belief. In computer science these mentalistic-sounding notions are actually given precise computational meanings and are used not only to prove properties of distributed systems, but to program them as well. A typical rule in such a knowledge-based systems is "if processor A does not *know* that processor B has received its message, then processor A will not send the next message." AOP augments these logics with formal notions of choices, capabilities, commitments, and possibly others. A typical rule in

the resulting systems will be "if agent A *believes* that agent B has *chosen* to do something harmful to agent A, then A will *request* that B change its choice." In addition, temporal information is included to anchor belief, choices, and so on in particular points in time.

Here again we may benefit from some ideas in philosophy and linguistics. As in the case of knowledge, there exists work in exact philosophy on logics for choice and ability. Although they have not yet had an effect in AI comparable to that of logics of knowledge and belief, they may in the future.

Intentional terms such as knowledge and belief are used in a curious sense in the formal AI community. On the one hand, the definitions come nowhere close to capturing the full linguistic meanings. On the other hand, the intuitions about these formal notions do indeed derive from the everyday, common sense meaning of the words. What is curious is that, despite the disparity, the everyday intuition has proven a good guide to employing the formal notions in some circumscribed applications. AOP aims to strike a similar balance between computational utility and common sense.

Two Scenarios

Below are two scenarios. The first is fairly complex and serves to illustrate the type of future applications envisioned. The second is a toy example and serves three purposes: it illustrates a number of AOP ideas more crisply; it is implementable in the simple AGENT-0 language described later in the chapter; and it illustrates the fact that agents need not be robotic agents.

Manufacturing Automation. Alfred and Brenda work at a car-manufacturing plant. Alfred handles regular-order cars, and Brenda handles special-order ones. The plant has a welding robot, Calvin. The plant is controlled by a coordinating program, Dashiel. The following scenario develops, involving communication between Alfred, Brenda, Calvin and Dashiel. It contains communication acts such as informing, requesting, committing, permitting, and commanding and requires agents to reason about the beliefs, capabilities, and commitments of other agents.

8:00: Alfred requests that Calvin promise to weld ten bodies for him that day; Calvin agrees to do so.

8:30: Alfred requests that Calvin accept the first body, Calvin agrees, and the first body arrives. Calvin starts welding it and promises Alfred to notify him when it is ready for the next body.

8:45: Brenda requests that Calvin work on a special-order car which is needed urgently. Calvin responds that it cannot right then but that it will when it finishes the current job, at approximately 9:00.

9:05: Calvin completes welding Alfred's first car, ships it out, and offers to weld Brenda's car. Brenda ships it the car, and Calvin starts welding.

9:15: Alfred inquires why Calvin is not yet ready for his (Alfred's) next car. Calvin explains why and also that it (Calvin) expects to be ready by about 10:00.

9:55: Calvin completes welding Brenda's car and ships it out. Brenda requests that it reaccept it and do some painting, but Calvin refuses, explaining that it does not know how to paint. Calvin then offers to weld another car for Alfred and proceeds to weld Alfred's cars for a while.

12:15: Brenda requests that Calvin commit to welding four more special-order cars that day. Calvin replies that it cannot, since that conflicts with its commitment to Alfred, who still has six unwelded cars. Brenda requests Alfred to release Calvin from its commitment to Alfred. Alfred refuses. Brenda requests that Dashiel (remember Dashiel?) order Calvin to accept her important request and revoke its commitment to Alfred. Dashiel orders Calvin to weld two of Brenda's cars, and then as many of Alfred's as time allows.

3:30: Calvin senses it is low on lubricant, and, without interrupting its work, asks Alfred for when the next lubrication is scheduled. Alfred responds that he doesn't know, but that Dashiel should know. Calvin asks Dashiel, who informs it that the next lubrication is scheduled for 4:00. Since that is sufficiently soon, Calvin continues to weld.

Airline Reservation. This example is a minor modification of one due to John McCarthy (1990), who uses it to illustrate his Elephant2000 programming language. The example involves three agents: a passenger P, an airline clerk C, and C's supervisor S. P and S are human, and C is a program (or is it the other way around? Or perhaps all three are programs?). Note that by confirming a reservation, the airline enters into a commitment to issue a boarding pass to the passenger at the appropriate time. Again, the following exchange involves requesting (a query is a request to inform), informing, permitting, and denying, as well as mental components such as knowledge and commitment.

March

P to C: Please inform me what flights you have from San Francisco to New York on April 18.

C to P: Flight 354 departs at 08:30, flight 293 departs at 10:00, flight 441 departs at noon.

P to C: Please book me on 354.

C to P: That is sold out.

P to C: Please book me on 293.

C to P: That is confirmed; your reservation number is 112358.

P to C: Please book me also on 441.

C to P: That conflicts with 293; I am not allowed to double book a passenger.

P to C: Please get permission to do so.

C to S: I request permission for the following double booking.

S to C: Permission denied.

C to P: Sorry, I cannot get approval.

April 18, at the airport

P to C: My name is P; I have a reservation for flight 293.

C to P: Here is your boarding pass.

Overview of the AOP Framework

A complete AOP system will include three primary components:

- A restricted formal language with clear syntax and semantics for describing mental state. The mental state will be defined uniquely by several modalities, such as belief and commitment.
- An interpreted programming language in which to program agents, with primitive commands such as REQUEST and INFORM. The semantics of the programming language will depend in part on the semantics of mental state.
- An "agentifier," converting neutral devices into programmable agents.

In the remainder of this document I will start with a short discussion of mental state. I will then present a general family of agent interpreters, a simple representative of which has already been implemented as AGENT-0. Related work is described in the fifth section. I will end with a summary of recent research results related to AOP.

On the Mental State of Agents

The first step in the enterprise is to define agents; that is, to define the various components of mental state and the interactions between them. There is not a unique "correct" definition, and different applications can be expected to call for specific mental properties.[3] In this section I summarize what could be viewed as a bare-bones theory of mental state, a kernel that will in the future be modified and augmented.

Components of a Language for Mental State

In related past research by others in AI, three modalities were explored: belief, desire, and intention (giving rise to the pun on BDI agent architectures). Other similar notions, such as goals and plans, were also pressed into service. These are clearly important notions; they are also complex ones, however, and not necessarily the most primitive ones.[4]

By way of motivation, here is an informal view of the world which underlies the selection. At any point in time, the future is determined by two factors: the past history, and the current actions of agents. For example, past history alone does not (in this view) determine whether I raise my arm; that is determined by whether in fact I take the appropriate action. The actions of an agent are determined by its *decisions*, or *choices*.[5] In other words, some facts are true for natural reasons, and other facts are true because agents decided to make them so. Decisions are logically constrained, though not determined, by the agent's beliefs; these beliefs refer to the state of the world (in the past, present or future), to the mental state of other agents, and to the capabilities of this and other agents. For example, given that the robot believes that it is incapable of passing through the narrow doorway, it will not decide to go through it. Decisions are also constrained by prior decisions; the robot cannot decide to be in Room 5 in five minutes if it has already decided to be in Room 3 at that time.

In the first instantiation of AOP, a language called AGENT-0 (Torrance 1991), we too started with quite basic building blocks, in fact much more basic than those mentioned so far. We incorporated two modalities in the mental state of agents: *belief* and *obligation* (or *commitment*). We also defined *decision* (or *choice*) as an obligation to oneself. Finally, we included a third category which is not a mental construct *per se: capability*.

By restricting the components of mental state to these modalities we in some informal sense excluded representation of motivation. Indeed, we did not assume that agents are "rational" beyond assuming that their beliefs, obligations and capabilities are internally and mutually consistent. This assumption stands in contrast to the other work mentioned above, which makes further assumptions about agents acting in their own best interests, and so on. Such stronger notions of rationality are obviously important, and I am convinced that in the future we will wish to add them. In fact, in her dissertation, Thomas introduced an AOP language that includes the notions of intending and planning (Thomas 1993).

Properties of the Various Components

I have so far not placed any constraints on the various modalities defined, and therefore have not guaranteed that they in any way resemble their common sense counterparts. We will now place such constraints. Just as there is no objectively "right" collection of mental categories, there is no "right" list of properties for any particular mental category. I have already stated that the correspondence between the formal definition and common sense will always be only approximate and that I would like to strike a balance between common sense and utility. Indeed, I expect different applications of AOP to call for different properties of belief, commitment, and capability. In this section I will briefly and informally define a number of properties I assume about the modalities. Formal definitions of these properties may be found in Shoham (1993).

These properties are quite weak, but they are sufficient to justify the terminology, and necessary for the design of the interpreter. The weakness of the assumptions ensures that the interpreters apply to a wide variety of applications. Still, even these assumptions will be inappropriate for some purposes, in which case a new type of interpreter will be required.

Internal consistency. I assume that both the beliefs and the obligations are internally consistent.

Good faith. I further assume that agents commit only to what they believe themselves capable of, and only if they really mean it.

Introspection. Although in general I do not assume that agents have total introspective capabilities, I *do* assume that they are aware of their obligations. On the other hand, I do not assume that agents are necessarily aware of commitments made to them.

Persistence of mental state. I have only placed restrictions on mental attitudes at a single instant of time. I conclude this section by discussing how mental states change or persist over time. Unlike with the previously discussed properties, precise constraints cannot currently be specified, but only informal guidelines.

Consider, for example, belief. The previously discussed restrictions allow agents which at one time believe nothing at all, shortly afterwards to have a belief about *every* sentence, and then again to become quite agnostic. Common sense suggests that beliefs tend to be more stable than that, and it would indeed be difficult to rely on the behavior of agents with such volatile beliefs. I will now place a strong condition on belief: I will assume that agents have perfect memory of and faith in their beliefs, and only let go of a belief if they learn a contradictory fact. Beliefs therefore persist *by default*. Furthermore, I will assume that the *absence* of belief also persists by default, although in a slightly different sense: if an agent does not believe a fact at a certain time (as opposed to believing the negation of the fact), then the only reason he will come to believe it is if he learns it.

How to formally capture these two kinds of default persistence is another story, which touches on issues that are painfully familiar to researchers in nonmonotonic temporal reasoning and belief revision. In fact, a close look at the logical details of belief (or knowledge) persistence reveals several very subtle phenomena, which have so far not been addressed in the literature.

In addition, obligations should persist—otherwise they wouldn't be obligations. As in the case of belief, however, the persistence is not absolute. Although by default obligations persist, there are conditions under which obligations are revoked.

These conditions presumably include explicit release of the agent by the party to which it is obligated, or alternatively a realization on the part of the agent that it is no longer able to fulfill the obligation. (In their discussion of the persistence of commitment, Cohen and Levesque [1990] actually propose a more elaborate second condition, one that requires common knowledge by the committer and committee of the impossibility; however, further discussion of their position and arguments against it would be too long a detour.)

Since decision is defined in terms of obligation, it inherits the default persistence. Notice, however, an interesting point about the persistence of decision: while an agent cannot unilaterally revoke obligations it has towards others, it can cancel obligations held towards it—including obligations it holds towards itself, namely decisions. An agent is therefore free to modify an existing decision, but unless he explicitly does so the decision will stand.

Finally, capabilities too tend not to fluctuate wildly. In fact, in this document I assume that capabilities are fixed: What an agent can do at one time it can do at any other time. However, I will allow to condition a capability of an action on certain conditions that hold at the time of action.

The Contextual Nature of Modal Statements

I have throughout the discussion talked of "unequivocal" statements regarding beliefs, obligations, and capabilities. Common sense, however, suggests that each of these modalities is context sensitive: I can print the document right now, but only in the context of the network being up; I am obligated to you to finish the work by tomorrow, but if my child has just been rushed to hospital then all bets are off (even though I am still capable of finishing the work). Indeed, McCarthy has argued that all statements, not only modal ones, should be viewed in context. Although I agree in principle and discuss it further in Shoham (1991), in this article I will ignore the issue of context sensitivity.

A Generic Agent Interpreter

In the previous section I discussed the first component of the AOP framework, namely the definition of agents. I now turn to the programming of agents and will outline a generic agent interpreter.

The behavior of agents is governed by programs; each agent is controlled by its own private program. Agent programs themselves are not logical entities, but their control and data structures refer to the mental state of the agent using the logical language.[6]

The Basic Loop

The behavior of agents is, in principle, quite simple. Each agent iterates the following two steps at regular intervals:

1. Read the current messages and update your mental state (including your beliefs and commitments);

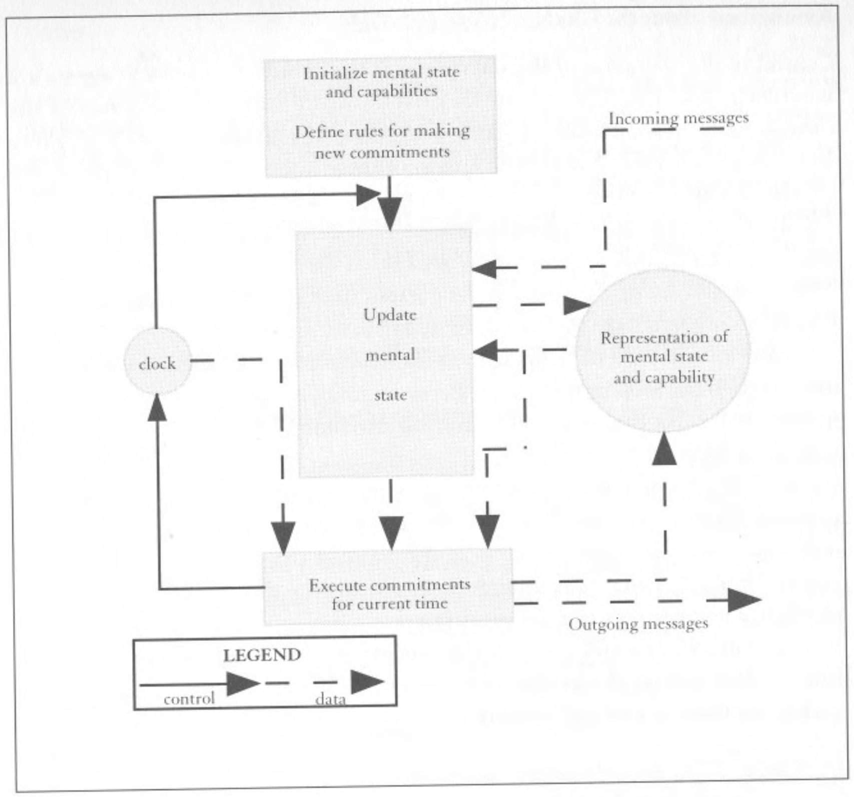
Initialize mental state and capabilities
Define rules for making new commitments
Incoming messages
Update mental state
clock
Representation of mental state and capability
Execute commitments for current time
Outgoing messages
LEGEND
control
data

Figure 2. A flow diagram of a generic agent interpreter.

2. Execute the commitments for the current time, possibly resulting in further belief change. Actions to which agents are committed include communicative ones such as informing and requesting.

The process is illustrated in figure 2; the dashed arrows represent flow of data, while the solid arrows show temporal sequencing.

Assumptions about Message Passing

Agent programs will include, among other things, communication commands. In order that those be executable, I will assume that the platform is capable of passing messages to other agents addressable by name, whether those reside in the same machine or in others. The programming language will define the form of these messages, and the interpreter will determine when messages are sent.

Assumption about the Clock

Central to the operation of the interpreter is the existence of a clock; agents are inherently "real time" (to use another overloaded term). The main role of the clock is to initiate iterations of the two-step loop at regular intervals (e.g., every 10 milliseconds, every hour). The length of these intervals is determined by the settable variable "time grain."

I do not discuss the implementation of such a clock, which will vary among platforms, and simply assume that it exists. I also assume a variable "now," whose value is set by the clock to the current time in the format defined in the programming language (e.g., an integer, date:hour:minute).

In previous work, I have made the very strong assumption that a single iteration through the loop lasts less than the time grain; in future versions of the language I will relax this assumption and correspondingly will complicate the details of the loop itself.

Of course, the fact that agents use the same temporal language does not ensure that their clocks are synchronized. If all are agents are running on the same machine; there will be no problem, but otherwise the possibility of clock drift exists. Although synchronization does not impact the design and programming of single agents, it is crucial for ensuring that a society of agents is able to function usefully. Fortunately, there exist synchronization protocols which ensure limited drift among clocks (for an overview, see Schneider [1987]), and we expect to use these in our applications.

AGENT-0: A Simple Language and its Interpreter

Agent interpreters may vary along many dimensions and in general pose many challenging problems. We have implemented a particular programming language called AGENT-0, whose interpreter is an extremely simple instance of the generic agent interpreter. In fact, the simplifications embodied in AGENT-0 are so extreme that it may be tempting to dismiss it as uninteresting. However, it was recognized early on that one would not gain good insight into the strengths and weaknesses of AOP without writing actual programs. It was decided therefore to implement a simple interpreter first, and design more complex languages and interpreters based on this experience. It turned out the design of AGENT-0 itself posed some challenges, and we have been surprised by the diversity of applications that even this simple language admits. Furthermore, AGENT-0 is designed in a way that suggests obvious extensions; a few are being currently pursued and are described in the final section.

The implemented interpreter is documented in Torrance (1991). A second, more complex interpreter was designed and implemented in collaboration with the Hewlett Packard Corporation.

Related Work

So far, I have not discussed related work in any depth. The body of related work is in fact so rich that in this section I will mention only the most closely related work, and briefly at that. I will omit further discussion of past work on logics of knowledge and belief, which the logic of mental state extends, since I already did that in the introduction. For the same reason, I will not discuss object-oriented programming and Hewitt's work. The following is ordered in what I see as decreasing relevance to, and overlap with, AOP. The order (or, for that matter, inclusion in the list) reflects no other ranking, nor is it implied that researchers high up on the list would necessarily endorse any part of AOP.

McCarthy's (1990) work on Elephant2000

This language under development is also based on speech acts, and the airline-reservation scenario I have discussed is due to McCarthy. One issue explored in connection with Elephant2000 is the distinction between illocutionary and perlocutionary specifications, which I have not addressed. In contrast to AOP, Elephant2000 currently contains no explicit representation of state, mental or otherwise. Conditional statements therefore refer to the history of past communication rather than to the current mental state.

Distributed AI

There is related work within Distributed AI community (cf. MCC [1990]). Although AOP is, to my knowledge, unique in its definition of mental state and the resulting programming language, others too have made the connection between object-oriented programming and agenthood (Ferber and Carle 1990; Hewitt 1990).

The Intelligent Communicating Agents Project (1987-1988)

This ambitious project, carried out jointly at Stanford, SRI and Rockwell International (Nilsson, Rosenschein, Cohen, Moore, Appelt, Buckley, and many others) had among its goals the representation of speech acts and connection between the intensional level and the machine level. See discussion of some of the individual work below.

Cohen and Levesque's Work on Belief, Commitment, Intention, and Coordination

These two researchers (Cohen and Levesque 1997, 1990) have also investigated the logical relationships between several modalities such as belief and choice. Although they have not approached the topic from a programming-language perspective as I have, they too have been interested in speech acts and mental state as building blocks for coordination and analysis of behavior. Their work has its roots in earlier work in natural language understanding by Allen, Cohen, and Perrault (Allen 1983; Cohen and Perrault 1979). Despite some similarities, crucial differences exist between the mental categories employed by Cohen and Levesque and ours.

Contract Nets

AOP shares with early work on contract nets (Smith 1980) the computational role of contracts among agents. However, the similarity ends there. Contract nets are based on broadcasting contracts and soliciting bids, as opposed to the intimate communication in AOP. Contract nets had no other notion of mental state, no range of communicative speech acts, nor any aspect of the asynchronous, real-time design inherent in AOP.

Situated Automata

Rosenschein and Kaelbling's situated automata (Kaelbling 1988; Rosenschein and Kaelbling 1986; Rosenschein 1985) is relevant in connection with the process of agentification. We adopt their idea of decoupling the machine language from the programmer's intensional conceptualization of the machine, but differ on the specific details.

Coordination

Several researchers have been concerned with the process of coordination in modern environments. For example, as a part of their more global project, Winograd and Flores have developed a model of communication in a work environment (Winograd and Flores 1986). They point to the fact that every conversation is governed by some rules, which constrain the actions of the participants: a request must be followed by an accept or a decline, a question by an answer, and so on. Their model of communication is that of a finite automaton, with the automaton states corresponding to different states of the conversation. This is a *macro* theory, a theory of societies of agents, in contrast to the *micro* theory of AOP. In related work, Malone and his associates are aiming towards a general theory of coordination, drawing on diverse fields such as computer science and economics (Malone 1991).

Informable Agents

Genesereth's work on informable agents (see Genesereth chapter, also in this volume). Genesereth's interest lies primarily in agents containing declarative knowledge that can be informed of new facts and that can act on partial plans. In this connection, he has investigated also the compilation of declarative plans and information into action commands. Genesereth uses the term "agents" so as to include also low-level finite-automaton-like constructs. AOP's structure of mental state is consistent with Genesereth's declarative regime but is not required by it.

Plan Representation and Recognition

Work on plan representation and recognition by Kautz, Pollack, Konolige, Litman, Allen, and others (e.g., Kautz 1990, Litman and Allen 1990, Pollack 1990, and Bratman 1987) also addresses the interaction between mental state and action, but it is usually concerned with finer-grained analyses, involving the actual representation of plans, reasoning limitations, and more complex mental notions such as goals, desires, and intentions.

Nilsson's Action Nets

ACTNET is a language for computing goal-achieving actions that depends dynamically on sensory and stored data. The ACTNET language is based on the concept of action networks. An action network is a forest of logical gates that select actions in response to sensory and stored data. The connection to AOP, albeit a weak one, is that some of the wires in the network originate from database items marked as "beliefs" and "goals." The maintenance of these databases is not the job of the action net.

Summary of Results and Ongoing Research

Work on mental state is proceeding on different fronts. Here are some pointers to ongoing research:

- In Moses and Shoham (1993) we provide some results on the connection between knowledge and (one kind of) belief.
- Thomas (1993) tackles the notions of capability, plan, and intentions.
- In Lamarre and Shoham (1994) we argue for the three-way distinction between knowledge, certainty, and belief.
- Brafman and Tennenholtz (1992) lay out a framework in which beliefs, preferences, and strategy are maintained in a form of "rational balance."
- Del Val and Shoham (1994) argue that the properties of belief update should be derived methodically from a theory of action and that doing so reveals some limitations of the KM postulates.
- Del Val and Shoham (1994) propose to reduce the notion of belief revision to that of belief update, and thus also to theories of action.
- In Shoham and Cousins (1994) we provide an initial survey of logics of mental state in AI.

In parallel with the logical aspects of action and mental state, we have investigated algorithmic questions:

- We have proposed a specific mechanism for tracking how beliefs change over time, called *temporal belief maps* (Isozaki and Shoham 1992).
- In Brafman, Latombe, and Shoham (1993) and Brafman et al. (1993) we show that, similar to distributed systems, the formal notion of knowledge can be applied to algorithmic robot motion planning. Recently, we proposed knowledge complexity as a useful general complexity measure in robotics, with an application to automating the distribution of robotic algorithms.

We have recently begun contemplating the role of agents in the context of digital libraries, whether or not they are of the AOP variety. We have so far conducted one experiment:

- There is an experiment to deploy adaptive agents that perform automated browsing of the World Wide Web on behalf of the user.

Finally, we are interested in how multiple agents can function usefully in the presence of other agents. In particular, we are interested in mechanisms that minimize conflicts among agents and have been investigating the utility of social laws in computational settings:

- In Shoham and Tennenholtz (1992) we propose a general framework for representing social laws within a theory of action and investigate the computational complexity of automatically synthesizing useful social laws. We also study a special case of traffic laws in a restricted robot environment.
- In Shoham and Tennenholtz (1995) we study ways in which such conventions emerge automatically in a dynamic environment. Early results were reported on in Shoham and Tennenholtz (1992).
- In Kittock (1994), he refines these results to take into account the topology of the agent network and the existence of asymmetric interactions among agents.

Notes

1. Elsewhere, I discuss how the gradual elimination of animistic explanations with the increase in knowledge is correlated very nicely with both developmental and evolutionary phenomena. In the evolution of science, theological notions were replaced over the centuries with mathematical ones. Similarly, in Piaget's stages of child development, there is a clear transition from animistic stages around the ages of 4-6 (when, for example, children claim that clouds move because they follow us around) to the more mature later stages.

2. There is one more dimension to the comparison, which I omitted from the table, and it regards inheritance. Inheritance among objects is today one of the main features of OOP, constituting an attractive abstraction mechanism. I have not discussed it since it is not essential to the idea of OOP, and even less so to the idea of AOP. Nevertheless a parallel can be drawn here, too. In OOP, specialized objects inherit the methods of more general ones. One analogous construct in AOP would be group agents, that is, agents that are made up of a group of simpler agents. If we define the beliefs of this composite agent as the "common beliefs" of the individual agents and the commitments of the composite agents as their "common commitments," then the mental attitudes of the group are indeed inherited by the individual.

3. In this respect our motivation here deviates from that of philosophers. However, I believe there exist sufficient similarities to make the connection between AI and philosophy mutually beneficial.

4. Cohen and Levesque (1990), for example, propose to reduce the notion of intention to those of goal and persistence. Their pioneering work introduces mental categories that are different from ours. The two frameworks share the essential view of belief and time. They each introduce modalities absent from the other: obligation and capability in our framework, goals and intentions in theirs. However, even two notions that at first appear to be similar—such as our "decision" and their "choice"—turn out to be quite different.

5. The term choice is somewhat ambiguous; I discuss various senses of choice later.

6. However, an early design of agent programs by Akahani was entirely in the style of logic programming; in that framework program statements themselves were indeed logical sentences.

References

Agre, P., and Chapman, D. 1987. PENGI: An Implementation of a Theory of Activity. In Proceedings of the Sixth National Conference on Artificial Intelligence, 268–272. Menlo Park, Calif.: American Association for Artificial Intelligence.

Allen, J. F. 1983. Recognizing Intentions from Natural Language Utterances. In *Computational Models of Discourse*, eds. M. Brady and R. C. Berwick, 107–166. Cambridge, Mass.: MIT Press.

Austin, J. L. 1962. *How to Do Things with Words*. Cambridge, Mass.: Harvard University Press.

Brafman, R., and Tennenholtz, M. 1994. Belief Ascription and Mental-Level Modeling. In *Proceedings of the Fourth International Conference on Principles of Knowledge Representation and Reasoning (KR'94)*. San Francisco, Calif.: Morgan Kaufmann.

Brafman, R. I.; Latombe, J.-C.; and Shoham, Y. 1993. Toward Knowledge-Level Analysis of Motion Planning. In Proceedings of the Eleventh National Conference on Artificial Intelligence, 670–675. Menlo Park, Calif.: American Association for Artificial Intelligence.

Brafman, R. I.; Latombe, J.-C.; Moses, Y.; and Shoham, Y. 1993. Knowledge as a Tool in Motion Planning under Uncertainty. In *Theoretical Aspects of Reasoning about Knowledge: Proceedings of the Fifth Conference (TARK 1994)*. San Francisco, Calif.: Morgan Kaufmann.

Bratman, M. E. 1987. *Intention, Plans, and Practical Reason*. Cambridge, Mass.: Harvard University Press.

Brooks, R. A. 1986. A Robust Layered Control System for a Mobile Robot. *IEEE Journal of Robot Automation* 2(1): 14–23.

Cohen, P. R., and Levesque, H. J. 1990. Intention Is Choice with Commitment. *Artificial Intelligence* 42(3): 213–261.

Cohen, P. R., and Levesque, H. J. 1997. Rational Interactions as the Basis for Communication. In *Intentions in Communication*, eds. P. R. Cohen, J. Morgan, and M. E. Pollack. Cambridge, Mass.: MIT Press. Forthcoming.

Cohen, P. R., and Perrault, C. R. 1979. Elements of a Plan-Based Theory of Speech Acts. *Cognitive Science* 3:177–212.

Del Val, A., and Shoham, Y. 1994. Deriving Properties of Belief Update from Theories of Action. *Journal of Logic, Language, and Information* 3(2): 81–119.

Del Val, A., and Shoham, Y. 1996. A Unified View of Belief Revision and Update. *Journal of Logic and Computation* 4(5): 797–810.

Dennett, D. C. 1987. *The Intentional Stance.* Cambridge, Mass.: MIT Press.

Drummond, M. 1989. Situated Control Rules. In *Proceedings of the First International Conference on Knowledge Representation and Reasoning,* eds. R. J. Brachman and H. J. Levesque, 103–113. San Francisco, Calif.: Morgan Kaufmann.

Ferber, J., and Carle, P. 1990. Actors and Agents as Reflective Concurrent Objects: A Mering IV Perspective. In Proceedings of the Tenth International Workshop on Distributed Artificial Intelligence, Technical Report ACT-AI-355-90, MCC, Austin, Texas.

Genesereth, M. R. 1989. A Proposal for Research on Informable Agents, Technical Report, Logic-89-4, Computer Science Department, Stanford University.

Grice, P. 1989. *Studies in the Ways of Words.* Cambridge, Mass.: Harvard University Press.

Halpern, J. Y., and Zuck, L. D. 1987. A Little Knowledge Goes a Long Way: Simple Knowledge-Based Derivations and Correctness Proofs for a Family of Protocols. In Proceedings of the Sixth ACM Symposium on Principles of Distributed Computing, 269–280. New York: Association of Computing Machinery.

Hayes-Roth, B.; Washington, R.; Hewett, R.; Hewett, M.; and Seiver, A. 1989. Intelligent Monitoring and Control. In Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, 243–249. Menlo Park, Calif.: International Joint Conferences on Artificial Intelligence.

Hewitt, C. 1990. Toward Open Information Systems Semantics. In Proceedings of the Tenth International Workshop on Distributed Artificial Intelligence, Technical Report ACT-AI-355-90, MCC, Austin, Texas.

Hewitt, C. 1977. Viewing Control Structures as Patterns of Passing Messages. *Artificial Intelligence* 8:323–364.

Isozaki, H., and Shoham, Y. 1992. A Mechanism for Reasoning about Time and Belief. In Proceedings of the Conference on Fifth-Generation Computer Systems, 694–701. Amsterdam: IOS Press.

Kaelbling, L. P. 1988. Goals as Parallel Program Specifications. In Proceedings of the Seventh National Conference on Artificial Intelligence, 60–65. Menlo Park, Calif.: American Association for Artificial Intelligence.

Kautz, H. A. 1990. A Circumscriptive Theory of Plan Recognition. In *Intentions in Communication,* eds. P. R. Cohen, J. Morgan, and M. E. Pollack, 60–65. Cambridge, Mass.: MIT Press.

Kittock, J. E. 1994. The Impact of Locality and Authority on the Emergence of Conventions. In Proceedings of the Twelfth National Conference on Artificial Intelligence, 420–425. Menlo Park, Calif.: American Association for Artificial Intelligence.

Lamarre, P., and Shoham, Y. 1994. Knowledge, Certainty, Belief, and Conditionalization. In *Proceedings of the Fourth International Conference on Principles of Knowledge Representation and Reasoning (KR'94),* eds. J. Doyle, E. Sandewall, and P. Torasso, 415–424. San Francisco, Calif.: Morgan Kaufmann.

Litman, D. J., and Allen, J. F. 1990. Discourse Processing and Commonsense Plans. In *Intentions in Communication*, eds. P. R. Cohen, J. Morgan, and M. E. Pollack. Cambridge, Mass.: MIT Press.

McCarthy, J. 1979. Ascribing Mental Qualities to Machines, Technical Report Memo, 326, AI Lab, Stanford University.

Malone, T. W. 1991. Toward an Interdisciplinary Theory of Coordination, Technical Report, CCS 120, Sloan School of Management, Massachusetts Institute of Technology.

MCC. 1990. Proceedings of the Tenth International Workshop on Distributed Artificial Intelligence, Technical Report ACT-AI-355-90, MCC, Austin, Texas.

Mitchell, T. M. 1990. Becoming Increasingly Reactive. In Proceedings of the Eighth National Conference on Artificial Intelligence, 1050–1058. Menlo Park, Calif.: American Association for Artificial Intelligence.

Moses, Y., and Shoham, Y. 1993. Belief as Defeasible Knowledge. *Journal of Artificial Intelligence* 64(2): 299–321.

Pollack, M. E. 1997. Plan as Complex Mental Attitudes. In Intentions in Communication, eds. P. R. Cohen, J. Morgan, and M. E. Pollack. Cambridge, Mass.: MIT Press. Forthcoming.

Rosenschein, S. J. 1985. Formal Theories of Knowledge in AI and Robotics, Technical Report 362, SRI International, Menlo Park, California.

Rosenschein, S. J., and Kaelbling, L. P. 1986. The Synthesis of Digital Machines with Provable Epistemic Properties. Paper presented at Conference on Theoretical Aspects of Reasoning about Knowledge, 19–22 March, Monterey, California.

Schneider, F. 1987. Understanding Protocols for Byzantine Clock Synchronization, Technical Report, Computer Science Department, Cornell University.

Searle, J. R. 1969. *Speech Acts: An Essay in the Philosophy of Language.* Cambridge, U.K.: Cambridge University Press.

Shoham, Y. 1993. Agent-Oriented Programming. *Journal of Artificial Intelligence* 60(1): 51–92.

Shoham, Y. 1991. Varieties of Context. In *Artificial Intelligence and Mathematical Theory of Computation*, 393–408. San Diego, Calif.: Academic.

Shoham, Y. 1989. Time for Action. In Proceedings of the Eleventh International Joint Conference on Artificial Intelligence, 954–959. Menlo Park, Calif.: International Joint Conferences on Artificial Intelligence.

Shoham, Y., and Cousins, S. 1994. Logics of Mental Attitudes in AI: A Very Preliminary Survey. In *Advances in Knowledge Representation and Reasoning,* eds. G. Lakemeyer and B. Mebel, 296–309. New York: Springer Verlag.

Shoham, Y., and Tennenholtz, M. 1992. Emergent Conventions in Multi-Agent Systems: Initial Experimental Results and Observations. In *Proceedings of the Conference on Principles of Knowledge Representation and Reasoning,* 225–231. San Francisco, Calif.: Morgan Kaufmann.

Shoham, Y., and Tennenholtz, M. 1995. Computational Social Systems: Offline Design. *Journal of Artificial Intelligence* 73(1–2): 231–252.

Smith, R. G. 1980. The Contract Net Protocol: High-Level Communication and Control in a Distributed Problem Solver. *IEEE Transactions of Computer Science* 29(12): 1104–1113.

Thomas, B. 1993. PLACA, An Agent-Oriented Language. Ph.D. diss., Department of Computer Science, Stanford University.

Torrance, M. 1991. The AGENT0 Programming Manual, Technical Report, Department of Computer Science, Stanford University.

Winograd, T., and Flores, F. 1986. *Understanding Computers and Cognition.* Norwood, N.J.: Ablex.

CHAPTER 14

KQML as an Agent Communication Language[1]

Tim Finin, Yannis Labrou, & James Mayfield

It is doubtful that any conversation about agents will result in a consensus on the definition of an agent or of agency. From personal assistants and "smart" interfaces to powerful applications, and from autonomous, intelligent entities to information retrieval systems, anything might qualify as an agent these days. But, despite these different viewpoints, most would agree that the capacity for interaction and interoperation is desirable. The building block for intelligent interaction is knowledge sharing that includes both mutual understanding of knowledge and the communication of that knowledge. The importance of such communication is emphasized by Genesereth, who goes so far as to suggest that an entity is a software agent if and only if it communicates correctly in an agent communication language (Genesereth and Ketchpel 1994). After all, it is hard to picture cyberspace with entities that exist only in isolation; it would go against our perception of a decentralized, interconnected electronic universe.

How might meaningful, constructive, and intelligent interaction among software agents be provided? The same problem for humans requires more than the knowledge of a common language such as English; it also requires a common understanding of the terms used in a given context. A physicist's understanding of velocity is not the same as that of a car enthusiast's,[2] and if the two want to converse about "fast" cars, they have to speak a "common language." Also, humans must resort to a shared etiquette of communication that is a result of societal development and that is partially encoded in the language. Although we are not always conscious of doing so, we follow certain patterns when we ask questions or make requests. Such patterns have common elements across human languages. Likewise, for software agents to interact and interoperate effectively requires three fundamental and distinct components: (i) a common language, (ii) a common understanding of the knowledge exchanged, and (iii) the ability to ex-