BDI Agents: From Theory to Practice

April, 1995

Technical Note 56

By:

Anand S. Rao
Australian Artificial Intelligence Institute

Michael P. Georgeff
Australian Artificial Intelligence Institute

This research was supported by the Cooperative Research Centre for Intelligent Decision Systems under the Australian Government's Cooperative Research Centres Program. This paper also appears in the *Proceedings of the First International Conference on Multi-Agent Systems (ICMAS-95)*, San Francisco, USA, June, 1995

Abstract

The study of computational agents capable of rational behaviour has received a great deal of attention in recent years. Theoretical formalizations of such agents and their implementations have proceeded in parallel with little or no connection between them. This paper explores a particular type of rational agent, a Belief-Desire-Intention (BDI) agent. The primary aim of this paper is to integrate (a) the theoretical foundations of BDI agents from both a quantitative decision-theoretic perspective and a symbolic reasoning perspective; (b) the implementations of BDI agents from an ideal theoretical perspective and a more practical perspective; and (c) the building of large-scale applications based on BDI agents. In particular, an air-traffic management application will be described from both a theoretical and an implementation perspective.

1 Introduction

The design of systems that are required to perform high-level management and control tasks in complex dynamic environments is becoming of increasing commercial importance. Such systems include the management and control of air traffic systems, telecommunications networks, business processes, space vehicles, and medical services. Experience in applying conventional software techniques to develop such systems has shown that they are very difficult and very expensive to build, verify, and maintain. Agent-oriented systems, based on a radically different view of computational entities, offer prospects for a qualitative change in this position.

A number of different approaches have emerged as candidates for the study of agent-oriented systems [Bratman *et al.*, 1988; Doyle, 1992; Rao and Georgeff, 1991c; Rosenschein and Kaelbling, 1986; Shoham, 1993]. One such architecture views the system as a rational agent having certain *mental attitudes* of Belief, Desire and Intention (BDI), representing, respectively, the information, motivational, and deliberative states of the agent. These mental attitudes determine the system's behaviour and are critical for achieving adequate or optimal performance when deliberation is subject to resource bounds [Bratman, 1987; Kinny and Georgeff, 1991].

While much work has gone into the formalization [Cohen and Levesque, 1990; Jennings, 1992; Kinny *et al.*, 1994; Rao and Georgeff, 1991c; Singh and Asher, 1990] and implementation [Burmeister and Sundermeyer, 1992; Georgeff and Lansky, 1986; Muller *et al.*, 1994; Shoham, 1993] of BDI agents, two main criticisms have been levelled against these endeavours. First, the having of these three attitudes is attacked from both directions: classical decision theorists and planning researchers question the necessity of having all three attitudes and researchers from sociology and Distributed Artificial Intelligence question the adequacy of these three alone. Second, the utility of studying multi-modal BDI logics which do not have complete axiomatizations and are not efficiently computable is questioned by many system builders as having little relevance in practice.

This paper addresses these two criticisms from the perspectives of the authors' previous work in BDI logics [Rao and Georgeff, 1991a; Rao and Georgeff, 1991c; Rao and Georgeff, 1993], systems [Georgeff and Lansky, 1986], and real-world applications [Ingrand *et al.*, 1992; Rao *et al.*, 1992]. We argue the necessity (though not the adequacy) of these three attitudes in domains where real-time performance is required from both a quantitative decision-theoretic perspective and a symbolic reasoning perspective. To address the second criticism, we show how one can build practical systems by making certain simplifying assumptions and sacrificing some of the expressive power of the theoretical framework. We first describe a practical BDI interpreter and show how it relates to our theoretical framework. We then describe an implemented agent-oriented air-traffic management system, called OASIS, currently being tested at Sydney airport.

The primary purpose of this paper is to provide a unifying framework for a particular type of agent, BDI agent, by bringing together various elements of our previous work in theory, systems, and applications.

2 The System and its Environment

We first informally establish the necessity of beliefs, desires, and intentions for a system to act appropriately in a class of application domains characterized by various practical limitations and requirements. As typical of such a domain, consider the design of an air traffic management system that is to be responsible for calculating the expected time of arrival

(ETA) for arriving aircraft, sequencing them according to certain optimality criteria, reassigning the ETA for the aircraft according to the optimal sequence, issuing control directives to the pilots to achieve the assigned ETAs, and monitoring conformance.

This and a wide class of other real-time application domains exhibit a number of important characteristics:

1. At any instant of time, there are potentially many different ways in which the environment can evolve (formally, the environment is nondeterministic); e.g., the wind field can change over time in unpredictable ways, as can other parameters such as operating conditions, runway conditions, presence of other aircraft, and so on.

2. At any instant of time, there are potentially many different actions or procedures the system can execute (formally, the system itself is nondeterministic); e.g., the system can take a number of different actions, such as requesting an aircraft change speed, stretch a flight path, shorten a flight path, hold, and so on.

3. At any instant of time, there are potentially many different objectives that the system is asked to accomplish; e.g., the system can be asked to land aircraft QF001 at time 19:00, land QF003 at 19:01, and maximize runway throughput, not all of which may be simultaneously achievable.

4. The actions or procedures that (best) achieve the various objectives are dependent on the state of the environment (context) and are independent of the internal state of the system; e.g., the actions by which the aircraft achieve their prescribed landing times depend on wind field, operating conditions, other aircraft, and so on, but not on the state of the computational system.

5. The environment can only be sensed locally (i.e., one sensing action is not sufficient for fully determining the state of the entire environment); e.g., the system receives only spot wind data from some aircraft at some times at some locations and thus cannot determine in one sensing operation the current wind field.

6. The rate at which computations and actions can be carried out is within reasonable bounds to the rate at which the environment evolves; e.g., changes in wind field, operational conditions, runway conditions, presence of other aircraft, and so on, can occur during the calculation of an efficient landing sequence and during the period that the aircraft is flying to meet its assigned landing time.

One way of modelling the behaviour of such a system, given Characteristics (1) and (2), is as a branching tree structure [Emerson, 1990], where each branch in the tree represents an alternative execution path. Each node in the structure represents a certain state of the world, and each transition a primitive action made by the system, a primitive event occurring in the environment, or both.

If we differentiate the actions taken by the system and the events taking place in the environment, the two different types of nondeterminism manifest themselves in two different node types. We call these *choice (decision) nodes* and *chance* nodes, representing the options available to the system itself and the uncertainty of the environment, respectively.

In this formal model, we can identify the objectives of the system with particular paths through the tree structure, each labelled with the objective it realizes and, if necessary, the benefit or payoff obtained by traversing this path.

As the system has to *act*, it needs to select appropriate actions or procedures to execute from the various options available to it. The design of such a selection function should enable the system to achieve effectively its primary objectives, given the computational resources available to the system and the characteristics of the environment in which the system is situated.

Under the above-mentioned domain characteristics, there are at least two types of input data required by such a selection function. First, given Characteristic (4), it is essential that the system have information on the state of the environment. But as this cannot necessarily be determined in one sensing action (Characteristics 1 and 5), it is necessary that there be some component of system state that represents this information and which is updated appropriately after each sensing action. We call such a component the system's *beliefs*. This component may be implemented as a variable, a database, a set of logical expressions, or some other data structure. Thus, beliefs can be viewed as the *informative* component of system state.[1]

Second, it is necessary that the system also have information about the objectives to be accomplished or, more generally, what priorities or payoffs are associated with the various current objectives (Characteristics 3 and 4). It is possible to think of these objectives, or their priorities, as being generated instantaneously or functionally, and thus not requiring any state representation (unlike the system beliefs, which cannot be represented functionally). We call this component the system's *desires*, which can be thought of as representing the *motivational* state of the system.[2]

Given this picture, the most developed approach relevant to the design of the selection function is decision theory. However, the decision-theoretic approach does not take into account Characteristic (6); namely, that the environment may change in possibly significant and unanticipated ways either (1) during execution of the selection function itself or (2) during the execution of the course of action determined by the selection function.

The possibility of the first situation arising can be reduced by using a faster (and thus perhaps less optimal) selection function, as there is then less risk of a significant event occurring during computation.

Interestingly, to the second possibility, classical decision theory and classical computer science provide quite different answers: decision theory demands that one re-apply the selection function in the changed environment; standard computer programs, once initiated, expect to execute to completion without any reassessment of their utility.

Given Characteristic (6), neither approach is satisfactory. Re-application of the selection function increases substantially the risk that significant changes will occur during this calculation and also consumes time that may be better spent in action towards achieving the given objectives. On the other hand, execution of any course of action to completion increases the risk that a significant change will occur during this execution, the system thus failing to achieve the intended objective or realizing the expected utility.

We seem caught on the horns of a dilemma: reconsidering the choice of action at each step is potentially too expensive and the chosen action possibly invalid, whereas unconditional commitment to the chosen course of action can result in the system failing to achieve its objectives. However, *assuming that potentially significant changes can be determined instantaneously*,[3] it is possible to limit the frequency of reconsideration and thus achieve an appropriate balance between too much reconsideration and not enough [Kinny and Georgeff,

[1] We distinguish beliefs from the notion of knowledge, as defined for example in the literature on distributed computing, as the system beliefs are only required to provide information on the likely state of the environment; e.g., certain assumptions may be implicit in the implementation but sometimes violated in practice, such as assumptions about accuracy of sensors, or rate of change of certain environmental conditions.

[2] We distinguish desires from goals as they are defined, for example, in the AI literature in that they may be many at any instant of time and may be mutually incompatible.

[3] That is, at the level of granularity defined by the primitive actions and events of the domain.

1991]. For this to work, it is necessary to include a component of system state to represent the currently chosen course of action; that is, the output of the most recent call to the selection function. We call this additional state component the system's *intentions*. In essence, the intentions of the system capture the *deliberative* component of the system.

3 Decision Trees to Possible Worlds

While in the previous section we talked abstractly about the belief, desire, and intention components of the system state, in this section we develop a theory for describing those components in a propositional form. We begin with classical decision theory and show how we can view such a theory within a framework that is closer to traditional epistemic models of belief and agency. In later sections, we will show how this model can then be used to specify and implement systems with the characteristics described above.

Informally, a decision tree consists of decision nodes, chance nodes, and terminal nodes, and includes a probability function that maps chance nodes to real-valued probabilities (including conditional probabilities) and a payoff function that maps terminal nodes to real numbers. A deliberation function, such as *maximin* or *maximizing expected utility* is then defined for choosing one or more best sequences of actions to perform at a given node.

We transform such a decision tree, and appropriate deliberation functions, to an equivalent model that represents beliefs, desires, and intentions as separate accessibility relations over sets of possible-worlds. This transformation provides an alternative basis for cases in which we have insufficient information on probabilities and payoffs and, perhaps more importantly, for handling the dynamic aspects of the problem domain.

We begin by considering a *full* decision tree, in which every possible path is represented (including those with zero payoffs). Given such a decision tree, we start from the root node and traverse each arc. For each unique state labeled on an arc emanating from a chance node, we create a new decision tree that is identical to the original tree except that (a) the chance node is removed and (b) the arc incident on the chance node is connected to the successor of the chance node. This process is carried out recursively until there are no chance nodes left. This yields a set of decision trees, each consisting of only decision nodes and terminal nodes, and each corresponding to a different possible state of the environment. That is, from a traditional possible-worlds perspective, each of these decision trees represents a different possible world with different probability of occurrence. Finally, the payoff function is assigned to paths in a straightforward way. The algorithm for this transformation can be found elsewhere [Rao and Georgeff, 1991b].

The resulting possible-worlds model contains two types of information, represented by the probabilities across worlds and the payoffs assigned to paths. We now split these out into two accessibility relations, the probabilities being represented in the belief-accessibility relation and the payoffs in the desire-accessibility relation. At this point in the story, the sets of tree structures defined by these relations are identical, although without loss of generality we could delete from the desire-accessible worlds all paths with zero payoffs.

Given a decision tree and the above transformation, an agent can now make use of the chosen deliberation function to decide the best course(s) of action. We can formally represent these selected path(s) in the decision tree using a third accessibility relation on possible worlds, corresponding to the intentions of the agent. In essence, for each desire-accessible world, there exists a corresponding intention-accessible world which contains only the best course(s) of action as determined by the appropriate deliberation function.

Thus, our possible worlds model consists of a set of possible worlds where each possible world is a tree structure. A particular index within a possible world is called a *situation*.

With each situation we associate a set of *belief-accessible* worlds, *desire-accessible worlds*, and *intention-accessible worlds*; intuitively, those worlds that the agent *believes* to be possible, *desires* to bring about; and *intends* to bring about, respectively.

4 BDI Logics

The above transformation provides the basis for developing a logical theory for deliberation by agents that is compatible with quantitative decision theory in those cases where we have good estimates for probabilities and payoffs. However, it does not address the case in which we do not have such estimates, nor does it address the dynamic aspects of deliberation, particularly those concerning commitment to previous decisions.

We begin by abstracting the model given above to reduce probabilities and payoffs to dichotomous (0-1) values. That is, we consider propositions to be either believed or not believed, desired or not desired, and intended or not intended, rather than ascribing continuous measures to them. Within such a framework, we first look at the static properties we would want of BDI systems and next their dynamic properties.

The axiomatization for beliefs that we adopt is the standard weak-S5 (or KD45) modal system[Hughes and Cresswell, 1984]. We adopt the D and K axioms for desires and intentions; i.e., desires and intentions have to be closed under implication and have to be consistent. We also have the inference rule of necessitation[Hughes and Cresswell, 1984] for beliefs, desires, and intentions.

A number of researchers have proposed their preferred axiomatizations capturing the relationships between beliefs, desires, and intentions. However, in other work [Rao and Georgeff, 1991c] we depart from this approach and give a comprehensive family of BDI logics similar in tradition to that of modal logic systems (i.e., KD45 system, S4 system, etc.). The reason for this departure is that we do not believe that there need be a unique and correct axiomatization that covers all interesting BDI agents—one may want to model different types of agents for different purposes.

Static Constraints: The static relationships among the belief-, desire-, and intention-accessible worlds can be examined along two different dimensions, one with respect to the *sets* of possible worlds and the other with respect to the *structure* of the possible worlds. Given two relations $\mathcal{R}_1$ and $\mathcal{R}_2$, four possible relationships are possible between them: one being a subset of the other and vice versa, and their intersections being null or non-null. Similarly, as each possible world is a time tree, there are four possible structural relationships that can hold between any pair of worlds: one could be a sub-world of the other or vice versa, or the worlds could be identical or incomparable.

Now we can combine the set and structural relationships of the belief, desire, and intention worlds to obtain twelve different BDI systems. Some of these relationships and axiomatizations can be derived from the others. Three of the above relationships and axiomatizations have been considered before under the terms *realism* [Cohen and Levesque, 1990] (if an agent believes a proposition, it will desire it), *strong realism* [Rao and Georgeff, 1991c] (if an agent desires to achieve a proposition, it will believe the proposition to be an option) and *weak realism* [Rao and Georgeff, 1991a] (if an agent desires to achieve a proposition, it will not believe the negation of the proposition to be inevitable).

The choice of BDI system depends also on which other properties are desired of the agent. For example, a number of researchers have proposed requirements concerning the asymmetry between beliefs and other attitudes [Bratman, 1987; Rao and Georgeff, 1991a] and consequential closure principles [Cohen and Levesque, 1990]. The first requires that rational agents maintain consistency between their beliefs, desires, and intentions, but not completeness. The second requires that the beliefs, desires, and intentions of an agent must not be closed under the implications of the other attitudes. All the above properties are satisfied by a BDI system in which the pair-wise intersections of the belief-, desire-, and intention-accessible worlds is non-null. Other BDI systems in which intention-accessible worlds are sub-worlds of desire-accessible worlds, which are sub-worlds of belief-accessible worlds satisfies some, but not all of these properties.

Dynamic Constraints: As discussed earlier, an important aspect of a BDI architecture is the notion of *commitment* to previous decisions. A commitment embodies the balance between the reactivity and goal-directedness of an agent-oriented system. In a continuously changing environment, commitment lends a certain sense of stability to the reasoning process of an agent. This results in savings in computational effort and hence better overall performance [Bratman, 1987; Kinny and Georgeff, 1991; Rao and Georgeff, 1991c].

A commitment usually has two parts to it: one is the condition that the agent is committed to maintain, called the *commitment condition*, and the second is the condition under which the agent gives up the commitment, called the *termination condition*. As the agent has no direct control over its beliefs and desires, there is no way that it can adopt or effectively realize a commitment strategy over these attitudes. However, an agent can choose what to do with its intentions. Thus, we restrict the commitment condition to intentions. An agent can commit to an intention based on the object of the intention being fulfilled in one future path or all future paths leading to different commitment conditions and hence different dynamic behaviours.

Different termination conditions result in further variations in behaviour[Rao and Georgeff, 1991c; Rao and Georgeff, 1993; Georgeff and Rao, August 1995]. For example, we can define a *blindly-committed* agent which denies any changes to its beliefs or desires that would conflict with its commitments; a *single-minded* agent which entertains changes to beliefs and will drop its commitments accordingly; and an *open-minded* agent which allows changes in both its beliefs and desires that will force its commitments to be dropped.

The various forms of termination and commitment can be expressed as axioms of our logic and semantic constraints can be placed on the dynamic evolution of the accessiblity relations. As before, rather than claiming that one particular commitment strategy is *the* right strategy, we allow the user to tailor them according to the application.

The purpose of the above formalization is to build formally verifiable and practical systems. If for a given application domain, we know how the environment changes and the behaviours expected of the system, we can use such a formalization to specify, design, and verify agents that, when placed in such an environment, will exhibit all and only the desired behaviours. Elsewhere [Rao and Georgeff, 1993] we have described how to verify certain behaviours of agents based on their static constraints and their commitment strategies using a model-checking approach. In the next section, we turn to the task of building a practical system based on the above theory.

5 Abstract Architecture

While it is not necessary that a system that is specified in terms of beliefs, desires and intentions be designed with identifiable data structures corresponding to each of these components, the architecture we propose below is based on such a correspondence. The rationale for such a design is that the identification of beliefs, desires, and intentions is useful when the system must communicate with humans or other software agents and can be expected to simplify the building, maintenance, and verification of application systems.

On the other hand, the architecture cannot be simply based on a traditional theoremproving system, even if extended to handle the temporal, epistemic, and non-deterministic elements of the logic described above. The reason for this is that the time taken to reason in this way, and thus the time taken to act, is potentially unbounded, thereby destroying the reactivity that is essential to an agent's survival. Thus, although we could use a theorem prover to reason "off-line" about the behaviour of an agent-based system, we cannot directly use such a theorem prover to implement the system itself.

The abstract architecture we propose comprises three dynamic data structures representing the agent's beliefs, desires, and intentions, together with an input queue of events. We allow update and query operations on the three data structures. The update operations on beliefs, desires, and intentions are subject to respective compatibility requirements. These functions are critical in enforcing the formalized constraints upon the agent's mental attitudes as described before. The events the system can recognize include both external events and internal events. We assume that the events are atomic and are recognized after they have occurred. Similarly, the outputs of the agent–actions–are also assumed to be atomic. The main interpreter loop is given below. We assume that the event queue, belief, desire, and intention structures are global.

BDI-interpreter
initialize-state();
repeat
    options := option-generator(event-queue);
    selected-options := deliberate(options);
    update-intentions(selected-options);
    execute();
    get-new-external-events();
    drop-successful-attitudes();
    drop-impossible-attitudes();
end repeat

At the beginning of every cycle, the option generator reads the event queue and returns a list of options. Next, the deliberator selects a subset of options to be adopted and adds these to the intention structure. If there is an intention to perform an atomic action at this point in time, the agent then executes it. Any external events that have occurred during the interpreter cycle are then added to the event queue. Internal events are added as they occur. Next, the agent modifies the intention and desire structures by dropping all successful desires and satisfied intentions, as well as impossible desires and unrealisable intentions.

This abstract architecture is an idealization that faithfully captures the theory, including the various components of practical reasoning [Bratman, 1987]; namely, option generation, deliberation, execution, and intention handling. However, it is not a practical system for rational reasoning. The architecture is based on a (logically) closed set of beliefs, desires, and intentions and the provability procedures required are not computable. Moreover, we have given no indication of how the option generator and deliberation procedures can be made sufficiently fast to satisfy the real-time demands placed upon the system.

We therefore make a number of important choices of representation which, while constraining expressive power, provide a more practical system for rational reasoning. The system presented is a simplified version of the Procedural Reasoning System (PRS) [Georgeff and Lansky, 1986; Ingrand *et al.*, 1992], one of the first implemented agent-oriented systems based on the BDI architecture, and a successor system, dMARS (distributed MultiAgent Reasoning System).

First, we explicitly represent only beliefs about the *current* state of the world and consider only ground sets of literals with no disjunctions or implications. Intuitively, these represent beliefs that are currently held, but which can be expected to change over time.

Second, we represent the information about the means of achieving certain future world states and the options available to the agent as *plans*, which can be viewed as a special form of beliefs [Rao and Georgeff, 1992]. Intuitively, plans are abstract specifications of both the means for achieving certain desires and the options available to the agent. Each plan has a *body* describing the primitive actions or subgoals that have to be achieved for plan execution to be successful. The conditions under which a plan can be chosen as an option are specified by an *invocation condition* and a *precondition*; the invocation condition specifies the "triggering" event that is necessary for invocation of the plan, and the precondition specifies the situation that must hold for the plan to be executable.

Third, each intention that the system forms by adopting certain plans of action is represented implicitly using a conventional run-time stack of hierarchically related plans (similar to how a Prolog interpreter handle clauses).[4] Multiple intention stacks can coexist, either running in parallel, suspended until some condition occurs, or ordered for execution in some way.

The main interpreter loop for this system is identical to the one discussed previously. However, as the system is embedded in a dynamic environment, the procedures appearing in the interpreter must be fast enough to satisfy the real-time demands placed upon the system. One way of tailoring and thus improving the process of option generation is to insert an additional procedure, `post-intention-status`, at the end of the interpreter loop. The purpose of this procedure is to delay posting events on the event queue regarding any changes to the intention structure until the end of the interpreter loop. By posting appropriate events to the event queue the procedure can determine, among other things, which changes to the intention structure will be noticed by the option generator. In this way, one can model various notions of commitment and results in different behaviors of the agent. Details of this and other efficiency techniques are described elsewhere[Rao and Georgeff, 1992].

6 Applications

In this section, we consider an air-traffic management system, OASIS, and relate it to the theoretical formalism and the abstract architecture of the previous sections. The system architecture for OASIS is made up of one `aircraft agent` for each arriving aircraft and a number of global agents, including a `sequencer`, `wind modeller`, `coordinator`, and `trajectory checker`. At any particular time, the system will comprise up to seventy or eighty agents running concurrently, sequencing and giving control directives to flow controllers on a real-time basis. The aircraft agents are responsible for flying the aircraft and the global agents are responsible for the overall sequencing and coordination of the aircraft agents. A detailed description of the system can be found elsewhere [Ljungberg and Lucas, 1992]. The system is currently undergoing parallel evaluation trials at Sydney airport, receiving live data from the radar.

Modelling: An `aircraft agent` is responsible for flying along a certain flight path given by the coordinates of a sequence of waypoints. An example of the chance or uncertainty in the domain is the wind field. If this were the only environmental variable, for each value of the wind velocity at a particular waypoint we would have a corresponding belief-accessible world. The choices available to an aircraft agent include flying along various trajectories

[4]This is an efficient way of capturing all the paths of intention-accessible worlds. In other words, the interpreter does a lazy generation of all possible sequences of actions that it can intend from the plan library.

between its minimum speed and maximum speed and at an altitude between its minimum and maximum altitude. This can be represented by multiple branches in each of the belief-accessible worlds mentioned above. As the final waypoint is the destination airport, the paths desired by the aircraft agent are those paths where the calculated ETA of the end node is equal to the desired ETA. The desire-accessible worlds can be obtained from the belief-accessible worlds by pruning those paths that do not satisfy the above condition. The intention-accessible worlds can be obtained from the desire-accessible paths by retaining only those that are the "best" with respect to fuel consumption, aircraft performance, and so on.

Decision Theory and Commitment: The primary objective of the `sequencer agent` agent is to land all aircraft safely and in an optimal sequence. Given the performance characteristics of aircraft, desired separation between aircraft, wind field, runway assignment, and a cost function, the sequencing agent uses a number of different deliberation strategies to compute the "best" arrival sequence for aircraft and their respective ETA's. On determining a particular schedule, the scheduling agent then single-mindedly commits to the intention; in other words, the scheduling agent will stay committed until (a) it believes that all aircraft have landed in the given sequence; or (b) it does not believe that there is a possibility that the next aircraft will meet its assigned ETA. Note that this is not the classical decision-theoretic viewpoint—any change in wind field, for example, should, in that view, cause a recalculation of the entire sequence, even if all aircraft could still meet their assigned ETAs.

Abstract Interpreter: In the implemented version of OASIS, each agent in the system deals only with current beliefs and desires and the options available to the agent to achieve its desires are written as plans. For example, although there may be many different ways of achieving the desired ETA (e.g., flying low at full speed), the plans of the aircraft agents only include as options those trajectories that are maximally fuel efficient.

In addition to the above application, PRS and dMARS have been used in a number of other large-scale applications, including a system for space shuttle diagnosis [Ingrand *et al.*, 1992], telecommunications network management [Ingrand *et al.*, 1992], air-combat modelling [Rao *et al.*, 1992], and business process management. This experience leads us to the firm conviction that the agent-oriented approach is particularly useful for building complex distributed systems involving resource-bounded decision-making.

Essential Features: The essential characteristics which have contributed to the success of our approach can be summarized as follows:

- The ability to construct plans that can *react to specific situations*, can be invoked based on their *purpose*, and are *sensitive to the context* of their invocation facilitates *modular* and *incremental development*. It allows users to concentrate on writing plans for a subset of essential situations and construct plans for more specific situations as they debug the system. As plans are invoked either in response to particular situations or based on their purpose, the incremental addition of plans does not require modification to other existing plans.

- The *balance* between *reactive* and *goal-directed behaviour* is achieved by committing to plans and periodically reconsidering such committed plans. The management of such real-time and concurrent activities is done by the system, while still giving the user control in terms of specifying to the system how the balance is to be achieved. As a result, end-users need not be involved in complex low-level programming (a difficult and error-prone activity, even for systems programmers), leading to a *reliable* system.

- The *high-level* representational and programming language has meant that end-users can encode their knowledge directly in terms of basic mental attitudes without needing to master the programming constructs of a low-level language. This has led to *greater* *flexibility* and *shorter development cycles*. For example, when FORTRAN rules that modelled pilot reasoning were replaced with plans, the turn-around time for changes to tactics in an air-combat simulation system [Rao *et al.*, 1992] improved from two months to less than a day.

7 Comparison and Conclusion

The BDI architecture draws its inspiration from the philosophical theories of Bratman [Bratman, 1987], who argues that intentions play a significant and distinct role in practical reasoning and cannot be reduced to beliefs and desires. Cohen and Levesque [Cohen and Levesque, 1990] provided one of the first logical formalizations of intentions and the notion of commitment. Later formalizations include the representationalist theory by Konolige and Pollack [Konolige and Pollack, 1993] and the work by Singh [Singh and Asher, 1990].

While the earlier formalisms present a particular set of semantic constraints or axioms as being *the* formalization of a BDI agent, we adopt the view that one should be able to choose an appropriate BDI system for an application based on the rational behaviours required for that application. As a result, following the modal logic tradition, we have discussed how one can categorize different combinations of interactions between beliefs, desires, and intentions.

A number of agent-oriented systems have been built in the past few years [Burmeister and Sundermeyer, 1992; Georgeff and Lansky, 1986; Muller *et al.*, 1994; Shoham, 1993]. However, while many of these appear interesting and have different strengths and weaknesses, none have yet been applied to as wide a class of complex applications as the ones discussed in this paper.

Currently, there is very little work on bridging the gap between theory, systems, and applications. The work by Bratman *et. al.* [Bratman *et al.*, 1988] describes the different modules of a BDI architecture and discusses the philosophical foundations for each of these modules. However, compared to our abstract interpreter, this model is at a higher level of abstraction and is not useful as a practical system. More recent work by Fisher [Fisher, 1994] on Concurrent Metatem specifies agent behaviours as temporal logic specifications that are directly executed by the system. However, for applications in which the environment changes at rates comparable with the calculation cycle of the system, such theorem provers are unsuited as system implementations.

The primary contribution of this paper is in integrating the various aspects of BDI agent research—theoretical foundations from both a quantitative decision-theoretic perspective and a symbolic rational agency perspective, system implementation from an ideal theoretical persepective to a more practical perspective, and the applications that rely on the theoretical foundations and are implemented using a practical BDI architecture.

Acknowledgements

This research was supported by the Cooperative Research Centre for Intelligent Decision Systems under the Australian Government's Cooperative Research Centres Program. We also thank Liz Sonenberg for her valuable comments on the paper.

References

[Bratman *et al.*, 1988] M. E. Bratman, D. Israel, and M. E. Pollack. Plans and resource-bounded practical reasoning. *Computational Intelligence*, 4:349–355, 1988.

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

[Burmeister and Sundermeyer, 1992] B. Burmeister and K. Sundermeyer. Cooperative problem-solving guided by intentions and perception. In E. Werner and Y. Demazeau, editors, *Decentralized A.I. 3*, Amsterdam, The Netherlands, 1992. North Holland.

[Cohen and Levesque, 1990] P. R. Cohen and H. J. Levesque. Intention is choice with commitment. *Artificial Intelligence*, 42(3), 1990.

[Doyle, 1992] J. Doyle. Rationality and its roles in reasoning. *Computational Intelligence*, 8(2):376–409, 1992.

[Emerson, 1990] E. A. Emerson. Temporal and modal logic. In J. van Leeuwen, editor, *Handbook of Theoretical Computer Science: Volume B, Formal Models and Semantics*, pages 995–1072. Elsevier Science Publishers and MIT Press, Amsterdam and Cambridge, MA, 1990.

[Fisher, 1994] Michael Fisher. Representing and executing agent-based systems. In *Intelligent Agents: Theories, Architectures, and Languages. Lecture Notes in Artificial Intelligence LNAI 890*, Amsterdam, Netherlands, 1994. Springer Verlag.

[Georgeff and Lansky, 1986] M. P. Georgeff and A. L. Lansky. Procedural knowledge. In *Proceedings of the IEEE Special Issue on Knowledge Representation*, volume 74, pages 1383–1398, 1986.

[Georgeff and Rao, August 1995] M. P. Georgeff and A. S. Rao. The semantics of intention maintenance for rational agents. In *Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence (IJCAI-95)*, Montreal, Canada, August, 1995.

[Hughes and Cresswell, 1984] G. E. Hughes and M. J. Cresswell. *A Companion to Modal Logic*. Methuen & Co. Ltd., London, England, 1984.

[Ingrand *et al.*, 1992] F. F. Ingrand, M. P. Georgeff, and A. S. Rao. An architecture for real-time reasoning and system control. *IEEE Expert*, 7(6), 1992.

[Jennings, 1992] N. R. Jennings. On Being Responsible. In Y. Demazeau and E. Werner, editors, *Decentralized A.I. 3.* North Holland, Amsterdam, The Netherlands, 1992.

[Kinny and Georgeff, 1991] David Kinny and Michael Georgeff. 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.

[Kinny *et al.*, 1994] D. Kinny, M. Ljungberg, A. S. Rao, E. A. Sonenberg, G. Tidhar, and E. Werner. Planned team activity. In *Artificial Social Systems, Lecture Notes in Artificial Intelligence (LNAI-830)*, Amsterdam, Netherlands, 1994. Springer Verlag.

[Konolige and Pollack, 1993] K. Konolige and M. Pollack. A representationalist theory of intention. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence (IJCAI-93)*, Chamberey, France, 1993.

[Ljungberg and Lucas, 1992] Magnus Ljungberg and Andrew Lucas. The oasis air-traffic management system. In *Proceedings of the Second Pacific Rim International Conference on Artificial Intelligence, PRICAI '92*, Seoul, Korea, 1992.

[Muller *et al.*, 1994] J. Muller, M. Pischel, and M. Thiel. A pragmatic approach to modelling autonomous interacting systems: Preliminary report. In *Intelligent Agents: Theories, Architectures, and Languages. Lecture Notes in Artificial Intelligence LNAI 890*, Amsterdam, Netherlands, 1994. Springer Verlag.

[Rao and Georgeff, 1991a] A. S. Rao and M. P. Georgeff. Asymmetry thesis and side-effect problems in linear time and branching time intention logics. In *Proceedings of the Twelfth International Joint Conference on Artificial Intelligence (IJCAI-91)*, Sydney, Australia, 1991.

[Rao and Georgeff, 1991b] A. S. Rao and M. P. Georgeff. Deliberation and its role in the formation of intentions. In *Proceedings of the Seventh Conference on Uncertainty in Artificial Intelligence (UAI-91).* Morgan Kaufmann Publishers, San Mateo, CA, 1991.

[Rao and Georgeff, 1991c] A. S. Rao and M. P. Georgeff. Modeling rational agents within a BDI-architecture. In J. Allen, R. Fikes, and E. Sandewall, editors, *Proceedings of the Second International Conference on Principles of Knowledge Representation and Reasoning.* Morgan Kaufmann Publishers, San Mateo, CA, 1991.

[Rao and Georgeff, 1992] A. S. Rao and M. P. Georgeff. An abstract architecture for rational agents. In C. Rich, W. Swartout, and B. Nebel, editors, *Proceedings of the Third International Conference on Principles of Knowledge Representation and Reasoning.* Morgan Kaufmann Publishers, San Mateo, CA, 1992.

[Rao and Georgeff, 1993] A. S. Rao and M. P. Georgeff. A model-theoretic approach to the verification of situated reasoning systems. In *Proceedings of the Thirteenth International Joint Conference on Artificial Intelligence (IJCAI-93)*, Chamberey, France, 1993.

[Rao *et al.*, 1992] A. Rao, D. Morley, M. Selvestrel, and G. Murray. Representation, selection, and execution of team tactics in air combat modelling. In Anthony Adams and Leon Sterling, editors, *Proceedings of the 5th Australian Joint Conference on Artificial Intelligence*, pages 185–190. World Scientific, November 1992.

[Rosenschein and Kaelbling, 1986] S. J. Rosenschein and L. P. Kaelbling. The synthesis of digital machines with provable epistemic properties. In J. Y. Halpern, editor, *Proceedings of the First Conference on Theoretical Aspects of Reasoning about Knowledge*, San Mateo, CA, 1986. Morgan Kaufmann Publishers.

[Shoham, 1993] Y. Shoham. Agent-oriented programming. *Artificial Intelligence*, 60(1):51–92, 1993.

[Singh and Asher, 1990] M. Singh and N. Asher. Towards a formal theory of intentions. In J. van Eijck, editor, *Logics in AI*, volume LNAI:478, pages 472–486. Springer Verlag, Amsterdam, Netherlands, 1990.