BDI Design Principles and Cooperative Implementation in RoboCup

Jan Wendler[1,*], Markus Hannebauer[2], Hans-Dieter Burkhard[1], Helmut Myritz[1], Gerd Sander[1], and Thomas Meinert[1]

[1] Humboldt University Berlin, Department of Computer Science,
Artificial Intelligence Laboratory, Unter den Linden 6, D-10099 Berlin, Germany
email: `wendler/hdb/myritz/sander/meinert@informatik.hu-berlin.de`,
WWW: `http://www.ki.informatik.hu-berlin.de`

[2] GMD – German National Research Center for Information Technology,
Research Institute for Computer Architecture and Software Technology (FIRST)
Planning and Optimization Laboratory, Kekulestr. 7, D-12489, Germany
email: `hannebau@first.gmd.de`

Abstract. This report discusses two major views on BDI deliberation for autonomous agents. The first view is a rather conceptual one, presenting general BDI design principles, namely heuristic options, decomposed reasoning and layered planning, which enable BDI deliberation in real-time domains. The second view is focused on the practical application of the design principles in RoboCup Simulation League. This application not only evaluates the usefulness in deliberation but also the usefulness in rapid cooperative implementation. We compare this new approach, which has been used in the Vice World Champion team AT Humboldt 98, to the old approach of AT Humboldt 97, and we outline our ideas for further improvements, which are still under work.

Conditions faced by deliberation in multi agent contexts differ significantly from the basic assumption of classical AI search and planning. Traditional game playing methods for example assume a static well-known setting and a fixed round-based interaction of players by a finite set of actions. Additionally, players have a rather long time for deliberation. In contrary to that, many real-world domains are characterized by a continuous action space and an environment that is permanently changed not only by the agent itself, but also by parallel events and actions of other agents. Domains with the need for real-time computing demand in addition time-bounded deliberation processes. The RoboCup Simulation League [9] is an artifical soccer testbed for the international evaluation of approaches that aim at agent deliberation in such real-time dynamic domains.

The *Belief-Desire-Intention (BDI) architecture* founded on Cognitive Science (refer e. g. to [1]) has been applied to deliberation in Artificial Intelligence by several researchers (e. g. [3, 8, 14]). It claims to be highly suitable for domains

* This research has been partially supported by the German Research Society, Berlin-Brandenburg Graduate School in Distributed Information Systems (DFG grant no. GRK 316).

that are characterized by a non-deterministic environment, competing desiderata, local information and bounded rationality. We have recently proven this to be true in RoboCup [2, 11]. We have also reported that structuring agents according to BDI is advantageous in the implementation process of RoboCup teams [7].

This report discusses two major views on BDI deliberation for autonomous agents. The first view given in section 1 is a rather conceptual one, presenting design principles, that can help to structure deliberation in real-time domains. In section 2, these design principles are directly evaluated by applying them to RoboCup. The development of RoboCup agents does not only impose constraints and difficulties on the deliberation design, but also on the implementation process. Usually, RoboCup development teams consist of several people with heterogeneous skills. Hence, section 2 also gives a glimpse on time- and resource-bounded software engineering, which is supported by the BDI design principles. In the conclusions we briefly evaluate and compare the BDI approach of AT Humboldt 97 to the presented approach, which has been used for RoboCup 98. Moreover, we outline our ideas for further improvements, which are still under work.

1 BDI Design Principles

The presented principles examine the following design tasks: **What** options of acting may an agent have in a given situation and what is their heuristic utility? **Which** of these options shall a rational agent choose as desired and intended? **How** can the intended options be pursued efficiently? These tasks directly correspond to consecutive phases in the deliberation design process.

1.1 Heuristic Options

As well as classical AI search is not sufficient for real-time dynamic domains, its set of terms, including "state" for the given situation and "operand" for an atomic action, is not sufficient to describe the agent's environment and its abilities to act. We therefore have been inspired by the notions *world* and *option* as introduced by Rao and Georgeff in [13] to substitute these terms.

A world is a timed snapshot of all environmental information which may be of use for the agent. Since we assume a situated agent's view on the environment, a world represents always local and incomplete (and even partially incorrect) knowledge about the real environment. Hence, our understanding of world is equivalent to that of *believed world*. Since the agent has usually more than one option to act, there are different following worlds. Some of these worlds fulfill certain conditions to be desirable for an agent. This subset of possible worlds is called *desired worlds* in classical BDI theory. Only a subset of these desired worlds may be achievable and consistent with respect to the given circumstances, and a rational agent may choose some of them to become *intended worlds*.

The other important notion is "option". In terms of possible future worlds, each option corresponds to a set of future worlds, where some conditions (e. g. ball control by the player) are fulfilled. Some of these worlds may be reachable by a related plan of the agent. But such a plan might also fail according to the non-determinism mentioned above, e. g. it might end in a world, where the desired condition does not hold.

The development of the world depends on the actions of others players, and of the uncertainty of the environment, too. The actions of cooperating agents are predictable to some extend. But the behavior of opponents is nearly unpredictable: We might assume that they behave in their best way, but then again we need to know about best plans (from opponents' view in this case). Thus, the outcome of a plan is non-deterministic from an agent's point of view. In [11] a theoretical model using Utility Theory (e. g. [12]) is used to describe this situation. In any case, it is impossible to compare all available plans by related calculations in a reasonable time.

This forces an approach according to the principles of "bounded rationality" [1] in the spirit of BDI. Our approach is based on *heuristic options*, which are the domain for *desires* and *intentions* (corresponding to a class `option` in object-oriented programming). Heuristic options are chosen from typical short-term goals, e. g. ball interception in soccer. It is important, that such goals consider a planning horizon which is restricted, but not in a uniform manner (a ball interception might e. g. last 3 or 30 steps). It is possible to enlarge this horizon by medium-term goals.

Options are realized by *skills*, which are configurable plans — or parameterizable procedures from the programming viewpoint, respectively. They implement typical basic capabilities of the agent, e. g. *running*, *kicking*, *dribbling* of RoboCup agents. As their names imply, they are in close relation to the options.

The task of deliberation is now the choice of a promising skill with appropriate parameters. This is basically the same problem as the above choice of a best plan, but our BDI-setting allows for useful heuristics. At first, we choose desires from the set of all available options: The options are ordered by approximations of their utilities, and the best scoring are considered as desires. Then it is proven, whether there really exists a plan for the achievement of such a desire. If it does, then the desire is chosen as an intention, which is successively refined by determining useful parameters for a related skill. According to BDI theory, the intention sets a screen of admissibility for the refinement of the plan, and in some cases for the consideration of conflicting future desires, too. Our utility approximation tries to determine a useful option, for which sufficiently reliable plans hopefully exist. When choosing such an option as a desire, we can dramatically restrict our search in the space of plans resp. possible future worlds. Further heuristics (including learning approaches) can be used for the determination of the concrete plan.

In principle, we can deal with multiple concurrent intentions – and we will use it in the future. Up to now additional intentions occur only in the special form of *constraints*: Several options may have certain properties in common. This holds especially in the case of resource consumption, which could be interpreted as costs for the execution of an option. Resource control is done by constraints, which can be applied to the heuristic utility calculation of an option. Constraints have also utilities. They can increase the overall expected utility of the option, which considers to fulfill them.

Concerning software technology, we state, that up to now agent oriented programming — and especially the implementation of BDI — has been mostly considered in the tradition of logic and rule based programming (e. g. [17, 6, 18]). Our approach uses agent oriented techniques as a structuring method in an object-oriented environment. Traditional programs are related to well understood control flows: The programmer knows in advance, which *procedures with which parameters* are to be called under which *concrete conditions*. This is implemented using the control structures of procedural languages. The situation changes in the case of autonomous agents in highly dynamic and complex worlds: The programmer does not know in advance all the conditions to call a procedure with appropriate parameters. Only the *criteria* of such calls can be described to some extend (as in a chess program: the programmer does not know about all concrete moves, only some decision criteria can be implemented). Instead of a fixed procedure control flow, an agent program has to implement a reasoning process for the choice of procedures. We will discuss some more details on this flexibility of control in the following subsection.

1.2 Decomposed Reasoning

BDI reasoning means to us the rational choice of promising options to become desires and intentions. This process should show the following properties, which are related to bounded rationality inside the agent as well as to software technological requirements:

Time-boundedness — In a real-time environment the reasoning process is bounded by time restrictions. Either the environment may enforce a timely decision (e. g. in applications with security demands) or late decisions may lead to suboptimal behavior (i.e. missing an opportunity to act).

Distinction of Control and Knowledge — The control of the reasoning process itself should be generic, such that it is independent from the domain-specific options and remains flexible.

Independence of Options — Alternating, deleting or adding an option should influence the reasoning process and other options only marginally or even not at all. This property could be called *scalability*.

The main idea that guarantees the fulfillment of the above mentioned demands is the decomposition of the reasoning process into modular heuristic options. In our model, every option implements a standardized interface, which defines an efficient utility estimation, an attainability predicate, a layered planner and a continuation enforcement predicate. This interface is used by the detached reasoning process as shown by figure 1. The concepts of *decision points*, *continuation enforcement* and *layered planning* will be described in the next subsection.

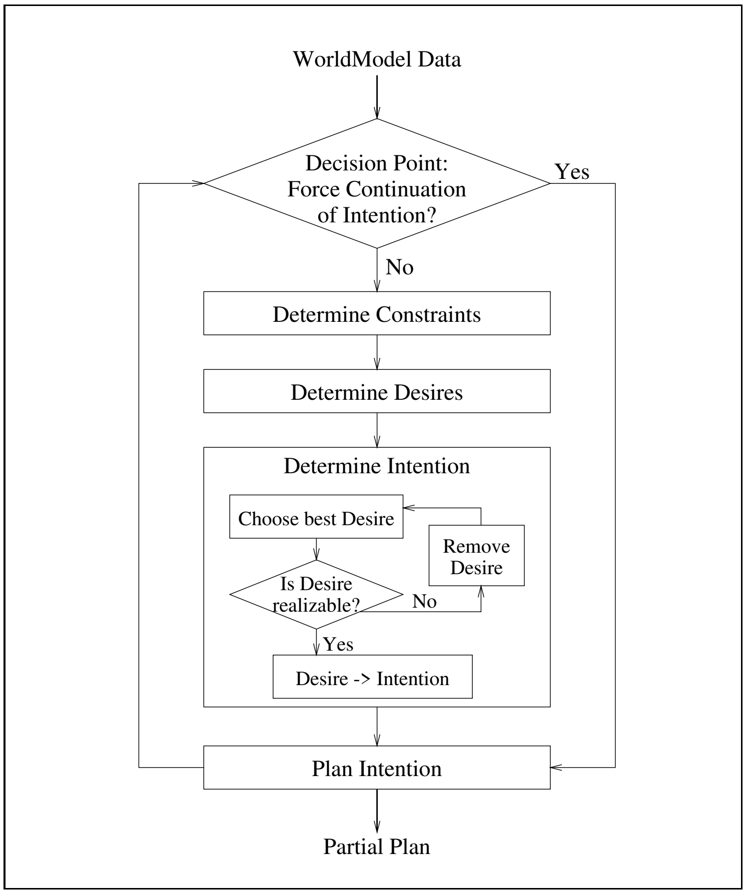

Fig. 1. BDI Reasoning Process

An initial reasoning process starts with the determination of applicable constraints. Given a domain-specific set of possible constraints PC and a utility threshold `MIN_CONSTRAINT_UTILITY`, the set of currently applicable constraints C can be determined by

$$C = \{c \in PC \mid c.\text{utility} > \texttt{MIN\_CONSTRAINT\_UTILITY}\}.$$

Given a domain-specific set of possible options PO and a utility threshold `MIN_DESIRE_UTILITY`, the set of current desires D can be determined by

$$D = \{o \in PO \mid o.\text{utility}\,(C) > \texttt{MIN\_DESIRE\_UTILITY}\}.$$

When choosing only one intention from the desires, the current intention i can be determined under consideration of the recent intention ri by

$$i = \underbrace{(\ \arg\max_{d \in D} \{d.\text{utility}\,(C) \mid d.\text{isPossible}\})}_{\text{possible desire with maximal utility}} .\text{adapt}\,(ri)\,.$$

Usually there is only one desire with a maximal utility, which is then chosen as intention. If there are two or more desires with the same maximal utility no further destinction between the desires can be done, so the intention is chosen arbitrary among these desires.

Let's look at the demanded properties of a BDI reasoning process in dynamic environments. The time-boundedness of the presented reasoning process depends on the efficiency of the constraints' and options' utility estimations. The best solution for utility estimations are approximation algorithms, which deliver the better estimations, the more computing time is available. The control process itself is highly efficient, since it follows a simple greedy approach and calculates the costly `isPossible` method only in case of promising desires. Control and knowledge is fully detached, because all utility and planning heuristics are encapsulated in options. This also supports the demanded scalability of the reasoning process, since options estimate their utility independently from other options.

1.3 Layered Planning

In making a rational choice of a single intention from the given desires, the agent has decided, what to do. The planning task is then to determine, how to do it. In dynamic environments there is always a trade-off between short-term reactive control and long-term deliberative planning [4]. Reactive control has the advantage of being always well-informed about the environment and the disadvantage of a highly restricted horizon. Just the opposite holds for long-term planning. To adjust the agent's planning horizon properly, we are experimenting with *layered planning*, which tries to incorporate the advantages of both reactivity and planning (related to abstractions [10, 16] and Hierarchical Task Network (HTN) planning [5, 15]). Figure 2 shows the different layers of planning, which include coarse-grained planning on the intention layer, fine-grained planning on the skill layer and execution on the atomic actions layer. Following the principle of decomposed reasoning, all the functionality described here lies within the intention, chosen by the reasoning process.

The topmost layer shows exactly one (abstract) intention that describes the intended transition from the current world to a new world satisfying the intended conditions. The coarse-grained planning horizon directly corresponds to the estimated length of the intention. Thus, an intention corresponds to a single compound task in HTN planning. There is always a special problem in choosing the time points for monitoring the progress of intention execution and for reconsidering the intention. Too few monitoring and reconsideration might lead to a behavior, which is not appropriate to the current situation, too much of it could overload the deliberation process. Our concept of monitoring and reconsideration involves the use of so-called *decision points*. They are time points, at which the agent monitors the environment and reconsiders its choices.

Decision points usually enclose several steps for atomic actions. At this point, fine-grained planning is needed. Fine-grained planning is done by the agent's skill that is associated to the chosen intention. Compared to HTN planning, a skill is similar to a method leading to primitive tasks. Since the distance between

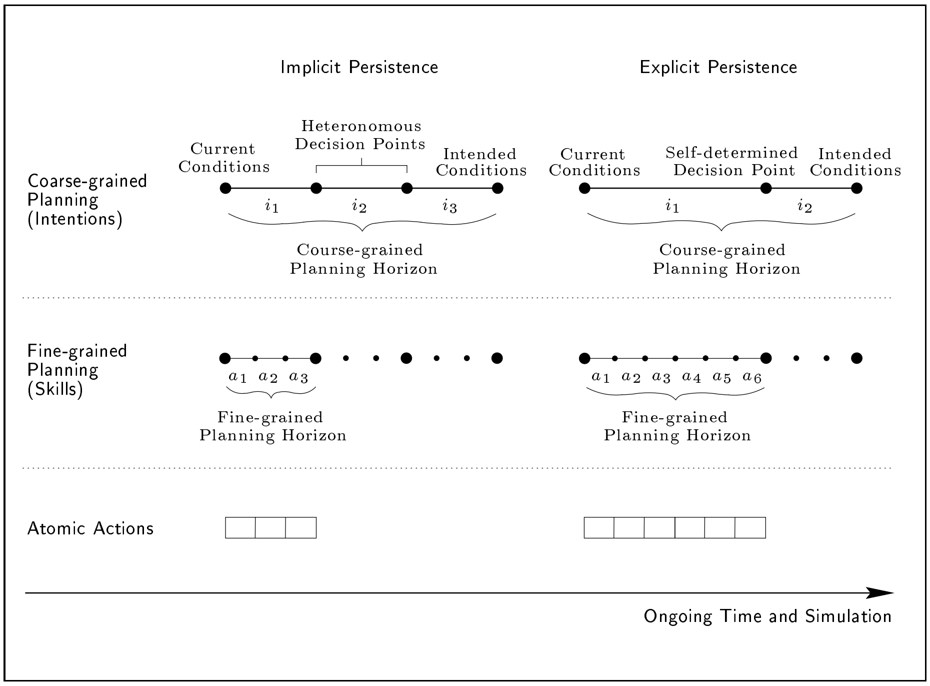

Fig. 2. Layered Planning

decision points might be dynamic and non-deterministic, fine-grained planning must have an anytime-property. That means, that these short-term plans should need no or only short initialization sequences. Otherwise, there would be much initialization overhead in case of near decision points. The approach of fine-grained planning is domain-dependent and can vary from intention to intention. For example it could use classical planning approaches, reactive planning or pre-compiled plan skeletons. Additionally, it should regard the constraints put onto the given intention. The action sequences planned by the fine-grained planner can directly be executed by atomic actions.

To guarantee stability of committed intentions we propose two different strategies, which influence the whole layered planning approach. The first is *implicit persistence*. Following this strategy, the agent considers all time points, at which the environment's behavior (e. g. an input of sensory data) implies reconsideration, as (heteronomous) decision points. In case of an environment, that has changed as expected, the intention chosen at one of these decision points will be most likely the same as the recent unfinished intention. In contrary to that, *explicit persistence* means, that the agent chooses on its own, on what time point to reconsider its intention. This is implemented by the intention's continuation enforcement predicate, which is illustrated in figure 1. In general, all planning layers could be subject to protections against unwanted changes: We might forbid a change of the intention while allowing a modification of the related plan, and we might even forbid changes of the plan itself (which we call "fanatism"), respectively.

2 Cooperative Implementation in RoboCup

Giving a brief example of a typical option in artificial soccer, in this section we will apply the presented generic BDI principles to the development of RoboCup agents. We will show, that these principles can not only be successfully used for agent deliberation, but also for rapid cooperative implementation. A more detailed report on this issue can be found in [7].

2.1 Constraints on Time and Staff

Artificial soccer is very useful for practical exercises in the area of Distributed Artificial Intelligence. Soccer is well-known and interesting for students. Results of work can directly be seen. In a practical exercise attached to the course 'modern methods of AI' in summer semester 1998 several students had to be included in the development process of AT Humboldt 98. In opposite to the year before, there was only few time between the start of the practical exercise (April) and the World Championship 98 (in the beginning of July).

Because of these hard time constraints, the project management was a critical task in this project. The development team of AT Humboldt 98 consisted of one core team including four persons and four feature teams with altogether eleven developers. Additionally, one of the major aims was to establish a code base, which is understandable, reusable and which can be extended and improved by additional work of students.

Due to the short time amount, much work had to be done in parallel: The students of the practical exercise had to be introduced into the new domain. Then the students had to pick up one component of the agent to specify, implement and test. After that, these components had to be integrated in the final agent and to be tested again to stabilize the behavior of the agents. The design principles of modular heuristic options and decomposed reasoning helped a lot in structuring the implementation work and breaking up the deliberation into small and to some degree independent pieces.

With AT Humboldt 99 we started to extend AT Humboldt 98 by improved skills, new options and partly by a larger planning horizon. These extensions are also subject to further improvements in the next versions of AT Humboldt.

2.2 Application of the BDI Design Principles: An Example

Our RoboCup agents know several *active options*, which may be only desired, if the agent has possession of the ball. They include such options like `GoalKick`, `DirectPass`, `ForwardPass`, `Dribbling` and so on. If the agent does not control the ball, *passive options* will be desirable. They include `CollectInformation`, `InterceptBall`, `GoToHomePosition`, `DefendGoal` and others. All these options have their own utility estimations, continuation enforcement predicates and planners. There are also two constraints, that have impact on the reasoning on option as described above. They are `ConserveStamina` and `AvoidOffside`.

One of the passive options used for our agent team is the option `InterceptBall`. If useful, it will represent the desire of intercepting the ball to gain ball control. The utility of intercepting the ball directly depends on the expected success in gaining ball control. The ball may be intercepted in different ways. Directly running into the way of the ball and waiting for it is secure but suboptimal, since an opponent may reach the ball first. Otherwise, if the agent tries to intercept the ball as fast as possible, it may miss the ball because of unforseen delays in its run. For a complete analysis of such a problem all possible plans have to be taken into account. As mentioned before, this seems impossible or at least not manageable. Therefore, we use a heuristic to calculate the utility of this option.

To estimate the utility of `InterceptBall` the agent calculates for itself and every team mate, how many time steps it will need to gain ball control if it moves optimally. For this calculation the agent uses the position of the team mate and the position and the speed of the ball. Furthermore this utility estimation can be influenced by the constraint `ConserveStamina`, which may be put onto `InterceptBall`. After having estimated the utility of this option, it may be chosen as a desire or even as an intention by the decomposed reasoning process.

In the latter case, the rough area for intercepting the ball is already known by the utility estimation. Hence, when fixing `InterceptBall` as intention, we determine the precise destination region at which the ball can be intercepted early but also relatively secure. The horizon of this coarse-grained plan corresponds to the time needed to gain ball control. After that, the coarse-grained plan has to be refined by a fine-grained plan reaching to the next decision point, which is done by a corresponding skill. While planning the first atomic actions towards the destination region according to the fine-grained planning horizon, the skill also considers the `ConserveStamina` constraint and avoids obstacles in the agent's path.

If an agent has chosen the option `InterceptBall` as intention and has determined a corresponding region, it may loose sight of the ball, while running to the destination region[1]. If the player reconsidered its intention every time a new sensor information arrives, it might try to look for the ball again and as a consequence loose time. To avoid this, the agent uses the continuation enforcement predicate of `InterceptBall`. During the calculation of the destination region, the agent also determines a "don't care" interval, in which no reconsideration is allowed. This interval is given by the minimal time, the fastest player needs to intercept the ball. Nevertheless, if the agent gets new information about the ball without additional actions, it is able to successfully reconsider its intention. In this case, the intention is not enforced to be continued. This behavior im-

[1] Players can use necks in Soccer Server Version 5, thus they can now turn his neck to observe the ball while running. Nevertheless, the example explains the idea of intention stability.

plements the mentioned explicit persistence and guarantees the stability of the `InterceptBall` intention.

3 Conclusion

To give a brief conclusion and overall evaluation of the presented BDI principles and their implementation, we compare them to the deliberation approach of the former soccer team AT Humboldt 97.

The development of the deliberation process in AT Humboldt 97 was rather intuitive. The team's knowledge of BDI design and implementation was just evolving and did not directly guide the development. Due to this, the resulting code did not precisely reflect the use of mental categories like belief, desires or intentions. For example, the choice of desires and intentions was done by a fixed hard-coded decision tree, which mixed up control and domain-specific knowledge. Hence, its development lay within the hands of only one person and it was hard to maintain. Writing the papers published later, helped the team to theoretically reconsider the techniques, which had been used informally. Ideas, which were already present in 1997, like layered planning, implicit and explicit persistence and others, underwent a strict review by a widened group of developers. This reconsideration lead to a full re-implementation of AT Humboldt.

The presented BDI principles have supported this re-implementation. The decomposition of control and knowledge has shown to be highly valuable for transparency and for rapid cooperative implementation. The notion of a modular heuristic option has played a central role in this context. By encapsulating the domain-specific knowledge, an option could be designed and realized almost independently from other options. Though, this modularity has certain drawbacks. The heuristic utility of a given option can not be found trivially. Another serious problem is the global normation, such that the utilities of different options remain comparable. We consider this to be a great challenge for learning techniques in our future work.

Since the reasoner always chooses only one intention to be pursued, the team had to introduce constraints to allow parallel influences. This concept has not fully paid off, since only a few of them could be identified in the real application. A better approach for future work would be the introduction of parallel intentions. In contrary to that, the principle of layered planning and persistence showed encouraging results. It allowed to balance the trade-off between adaption and stability very well, especially in case of passive options.

The BDI deliberation process of AT Humboldt 98 has proven to be flexible, scalable and maintainable. It provides a useful base for further improvements, which are still under work. The main goal is the introduction of longer planning intervals. A "cascade of intentions" will be used to establish raw sequences of subsequent future steps, which can be refined according to the development of the environment. "Emergent cooperation" in AT Humboldt 98 and 99 resulted from the programmer's knowledge about the implementation. Now we want to extend this behavior by explicitly plannable cooperation. On the lower level, additional and improved skills are to be developed (e. g. using learning techniques).

References

1. Bratman, M. E.: *Intentions, Plans, and Practical Reason.* Harvard University Press, 1987.
2. Burkhard, H.-D., Hannebauer, M., and Wendler, J.: *Belief-Desire-Intention Deliberation in Artificial Soccer.* AI Magazine 19(3): 87–93. 1998.
3. Cohen, P. R. and Levesque, H. J.: *Intention is choice with commitment.* Artificial Intelligence 42: 213–261. 1990.
4. Dean, T. L. and Wellman, M. P.: *Planning and Control.* Morgan Kaufmann Publishers, 1991.
5. Erol, K., Hendler, J. and Nam, D. S.: *HTN Planning: Complexity and Expressivity.* In Proc. of the 11th Nat. Conf. on AI (AAAI-94). AAAI Press, 1994.
6. Fisher, M.: *A survey of Concurrent* METATEM *— the language and its applications.* In Gabbay, D. M. and Ohlbach, H. J. (eds.): Temporal Logic — Proceedings of the First International Conference: 480–505. LNAI 827. Springer, 1994.
7. Hannebauer, M., Wendler, J., and Müller-Gugenberger, P.: *Rapid Concurrent Software Engineering in Competitive Situations.* In Chawdhry, P. K., Ghodous, P. and Vandorpe, D. (eds.): Advances in Concurrent Engineering (CE-99): 225–232. Technomic Publishing, 1999.
8. Jennings, N. R.: *Specification and Implementation of a Belief-Desire-Joint-Intention Architecture for Collaborative Problem Solving.* Int. Journal of Intelligent and Cooperative Information Systems 2(3): 289–318. 1993.
9. Kitano, H., Kuniyoshi, Y., Noda, I., Asada, M., Matsubara, H., and Osawa, H.: *RoboCup: A Challenge Problem for AI.* AI Magazine 18(1): 73–85. 1997.
10. Knoblock, C. A.: *Automatically generating Abstractions for Planning.* Artificial Intelligence 68(2). 1994.
11. Müller-Gugenberger, P. and Wendler, J.: *AT Humboldt 98 — Design, Implementierung und Evaluierung eines Multiagentensystems für den RoboCup-98 mittels einer BDI-Architektur.* Diploma Thesis. Humboldt University Berlin, 1998.
12. Neumann, J. V. and Morgenstern, O.: *Theory of Games and Economic Behavior.* Princeton University Press, 1944.
13. Rao, A. S. and Georgeff, M. P.: *An Abstract Architecture for Rational Agents.* In Nebel, B., Rich, C. and Swartout, W. (eds.): Proc. of the Third International Conference on Principles of Knowledge Representation and Reasoning: 439–449. Morgan Kaufmann Publishers, 1992.
14. Rao, A. S. and Georgeff, M. P.: *BDI agents: From theory to practice.* In Lesser, V. (eds.): Proc. of the First Int. Conf. on Multi-Agent Systems (ICMAS-95): 312–319. MIT-Press, 1995.
15. Tate, A.: *Generating Project Networks.* In Allen, J., Hendler, J. and Tate, A. (eds.) Readings in Planning: 291–296. Morgan Kaufmann, 1990.
16. Tenenberg, J.: *Abstraction in Planning.* In Allen, J., Kantz, H., Pelavin, R. and Tenenberg, J. (eds.): Reasoning about Plans. Morgan Kaufmann, 1990.
17. Thomas, S. R.: *PLACA, an Agent Oriented Programming Language.* Technical Report STAN-CS-93-1487. Standford University, 1993.
18. Wooldrige, M. J.: *This is* MYWORLD*: The Logic of an Agent-Oriented DAI Testbed.* In Wooldrige, M. J. and Jennings, N. R. (eds.): Intelligent Agents: 160–178. LNAI 890. Springer, 1995.