Goal Delegation without Goals
BDI Agents in Harmony with OCMAS Principles

Alexander Pokahr and Lars Braubach

Distributed Systems and Information Systems
Computer Science Department, University of Hamburg
Vogt-Kölln-Str. 30, 22527 Hamburg, Germany
{pokahr, braubach}@informatik.uni-hamburg.de

Abstract. The BDI model is concerned with the rational action of an individual agent. At the multi-agent layer especially coordination among agents is an important factor that determines how overall system goals can be accomplished. Thus, from a software engineering perspective it is desirable to extend the BDI programming model to the multi-agent layer and make BDI concepts also useable for coordination among agents. A severe problem with existing approaches that tried to follow this path e.g. by proposing a BDI teamwork model is that they violate the OCMAS principles stating that no assumptions on agent architectures should be made on the multi-agent level. If OCMAS principles are violated the kinds of agents that can participate in coordination is limited ab initio to a specific sort such as BDI. In this paper we propose a new goal delegation mechanism that allows for both. On the one hand, BDI agents can delegate their normal goals to other agents and on the other hand these goals are not represented explicitly on the multi-agent level so that also non-BDI agents can act as receivers and help accomplishing a goal.

1 Introduction

The belief-desire-intention model (BDI) of Bratman [3] was inspiration source for one of the most successful agent architectures called PRS (procedural reasoning system) initially proposed in [9]. Main appeal of the BDI agent architecture is its folk psychological grounding that allows developers to naturally describe agent behavior in terms of beliefs (what is known of the world), goals (what is to be achieved), and plans (how goals can be achieved) in a similar way humans think that they think [12]. The traditional BDI architecture from [15] (cf. Fig. 1 left hand side) reduces the core process of BDI - called practical reasoning - to the means-end reasoning phase. Means-end reasoning refers to the process of deciding how to accomplish a goal or how to treat an event. Software-technically this is interpreted as plan selection and execution process, i.e. for a given goal the agent searches relevant plans (fitting to the goal type) and further inspects their applicability in the current context (referring to its beliefs). From this knowledge an applicable plan list is created and plans are executed one by one until the goal is achieved or no more plans are available causing the goal to be failed.

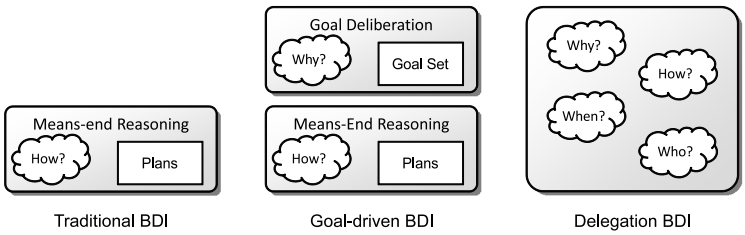

Fig. 1. BDI deliberation

In previous work the traditional BDI architecture has been extended towards supporting the full practical reasoning cycle consisting of the phases goal deliberation and means-end reasoning (cf. Fig. 1 middle). In the upstream goal deliberation phase an agent deals with the question which of its goals it should pursue at the current point in time. The main aspect of goal deliberation consists in generating a conflict-free goal set for the agent as some of the existing goals interfere with each other. Prerequisites for supporting the full practical reasoning cycle consist in introducing explicitly represented goals [6,4], a goal deliberation strategy [14] and an extended BDI architecture with a new deliberation cycle [13]. These extensions allow for programming goal-driven agents that are able to autonomously deliberate about their goals and pursue them in flexible context dependent ways but do not touch the area of inter-agent dependencies.

Thus, in the envisioned BDI architecture (cf. Fig. 1 right hand side), besides the aforementioned intrinsic aspects of goal deliberation and means-end reasoning, also goal delegation should be covered. This transforms a multi-agent system into a distributed goal-oriented reasoning engine and allows for natural cooperation among cognitive agents. In contrast to previous extensions, goal delegation renders the deliberation process much more difficult. Basically, it has to be decided in which cases a goal should be delegated and to which agent. Furthermore, also timing aspects become more important because goal deadlines might influence the goal delegation decisions. In contrast to the aforementioned two-phased process of practical reasoning, goal delegation requires a new BDI deliberation approach.

As a first step towards such general architecture in this paper the foundations of goal delegation will be laid out. In the next Section related work is discussed. In this context, the concept of goal delegation is explained and contrasted with similar but nonetheless different approaches from the teamwork area. In Section 3 the principles and architecture of the new goal delegation approach are presented. Finally, in Section 4 some concluding remarks and aspects of planned future work are given.

2 Related Work

For the treatment of related work, first goal delegation is placed it into a larger context regarding the general coordination of BDI agents. Afterwards existing works with regard to goal delegation are presented.

Example Approach \ Shared Attitude(s)	Beliefs	Goals	Plans
Hive BDI	X	-	-
Goal Delegation	-	X	-
Coordinated SaPa	-	-	X
Joint Intentions	X	X	-
Joint Responsibility	X	X	X
OCMAS	-	-	-

Table 1. Comparison of coordination approaches

2.1 BDI Agents and Coordination

Coordination among BDI agents in a multi-agent system can take a number of forms. A fundamental property of BDI agents is their definition in terms of mental attitudes such as belief, goals and plans. Therefore an interesting question concerns the role of such mental attitudes in any coordination approach. A large family of BDI coordination approaches can be derived by following the general assumption that when some mental attitudes are shared, the individual reasoning processes will lead to coordinated behavior. This idea is appealing due to a number of reasons. On the one hand, using mental attitudes also for coordination purposes is conceptually in line with the intuitive, folk psychological interpretation of BDI. On the other hand, the BDI reasoning process is already quite sophisticated and it is expected that introducing extensions for sharing mental attitudes is technically less challenging than to develop a new reasoning process specifically for coordination.

Table 1 differentiates some existing coordination approaches with regard to the question, which mental attitudes are shared among agents. The approaches have been selected as examples for their respective category. The selection of approaches should not be considered exhaustive and an in-depth treatment of the individual representatives is out of the scope of this paper. Instead, the following descriptions try to elaborate the general implications of sharing some mental attitude(s). In particular, the criteria of coupling and communication are considered. Coupling refers to how directly one agent is able to influence the behavior of another agent. In addition, the communication overhead of coordination results from both the communication load (i.e. the amount of transferred data) and the communication frequency (i.e. how often agents need to engage in communication). The result of this analysis is illustrated in Figure 2.

In *Hive BDI* [1] agents can share some of their beliefs and thus induce indirect coordination similar to the way ant-like agents influence each other using digital pheromones. The approach aims at supporting large-scale agent systems composed of huge numbers of agents and features a high robustness and scalability. The sharing of beliefs represents a low coupling, because agents only influence each other indirectly. The communication overhead is relatively high, due to the communication frequency, which e.g. follows from the dynamics of the environment. *Goal delegation* has been defined in [2] as the *delegation of commitment*, i.e. an agent adopting a commitment of another agent, and the *delegation of*

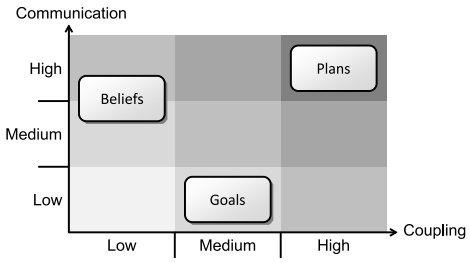

Fig. 2. Communication and coupling implications of shared mental attitudes

strategy, i.e. the agent on the receiving and has full control over how it pursues the delegated goal. In consequence, goal delegation incurs a rather low communication overhead, because at the start of the goal delegation, only the commitment needs to be communicated and after the delegated goal is achieved, some result might be passed back. Goal delegation allows medium influence on the behavior of other agents, because goal delegation states what the agent should achieve (commitment), but not how (strategy). Tight coupling can be produced by plan sharing approaches. E.g. *Coordinated Sapa* [10] allows resolving conflicts between plans, such that two cooperative agents can pursue their goals without accidentally executing actions detrimental to the other agent. This is facilitated by one agent sending its plan to the other and the other agent choosing its own actions in a way that avoids any conflict. Therefore tight coupling is established that requires communicating complete plan structures. There are also combined approaches that incorporate sharing of more than one attitude. E.g. in joint intentions [7], the notion of a shared goal is extended to require also shared knowledge about the state of the goal. Joint responsibility [11] takes these ideas further by also incorporating the plan level. These approaches lead to increased coupling, that may be advantageous, e.g. to improve the performance of teamwork, but comes with the cost of increased communication overhead.

2.2 Goal Delegation Approaches

The work presented in this paper aims at exploiting the intuitive BDI concepts for the development of open multi-agent systems. The above analysis shows that goal delegation represents a good trade-off between coupling and communication. With the initial delegation of the goal and the returning of some final result, it has clear interaction points with an intuitive semantics. Yet, for open and extensible systems the principles of organization-centered multi-agent systems (OCMAS) as laid down in [8] are considered very important as well. OCMAS advocates to treat agents as black boxes and therefore prohibits the use of mental attitudes at the MAS level. For a combination of both, two aspects are important. First, on the MAS-level, goal delegation needs to be represented in a standardized interaction that is open to non-BDI agents, too. Second, on the BDI-level, the standardized interaction scheme needs to be mapped to goals, which are delegated to and from other agents. Finally, the BDI reasoning cycle needs to be extended to

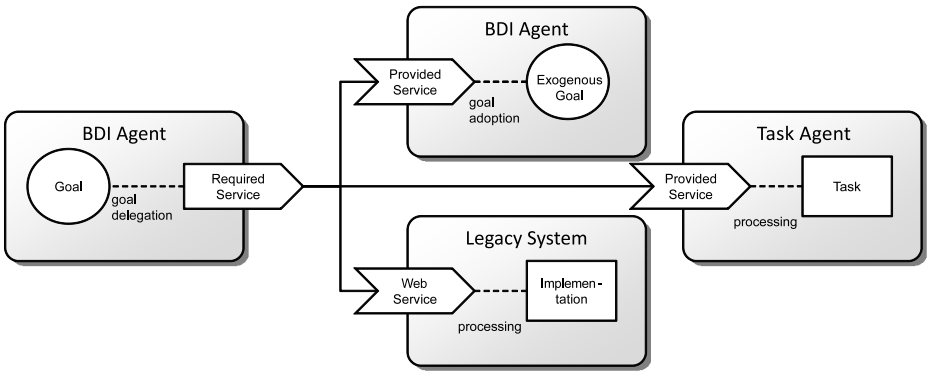

Fig. 3. Goal delegation concept

incorporate decisions regarding the delegation and acceptance of goals. In the following, some existing goal delegation approaches are discussed with respect to their support for either of those aspects.

In [2] the authors propose a FIPA-compliant protocol for goal delegation. While being FIPA-compliant means that syntactically, any kinds of agents may engage into conversations according to this protocol, the motivation of the protocol is explicitly capturing the semantics of goal delegation in the protocol itself. It is therefore a heavy-weight approach that violates OCMAS principles in favor of a clearly defined semantics for each message. Furthermore, no suggestions regarding the implementation of BDI agents according to the protocol are made.

An explicit distinction between the MAS and the BDI level is considered by [16]. In addition to the goal inside an agent, the notions of task and service are introduced. A task represents a concrete goal instance to be delegated to another agent (i.e. on the delegator side), whereas a service describes the ability of an agent to receive task requests (delegatee side). To perform the actual delegation of tasks, the usage of a contract-net negotiation is proposed. Furthermore, a reasoning cycle has been conceived that triggers delegation of goals, when no local plans are found. The approach is interesting due to the explicit separation of the three aspects of goals (goal, task, service). Yet, the approach is not fully generic, because of the contract-net-oriented interaction and goal delegation being hard-coded into the reasoning cycle.

3 OCMAS Goal Delegation

There are two main requirements for our goal delegation approach. The first requirement is that BDI agents should be enabled to use the notion of goal delegation to outsource some of their goals to other agents. This means that the specification of BDI agents needs to be extended to incorporate details about which goals can be delegated at which moments to what agents. The second requirement mandates that the goal delegation approach should be in line with the OCMAS principles, i.e. the approach should not make any assumptions about

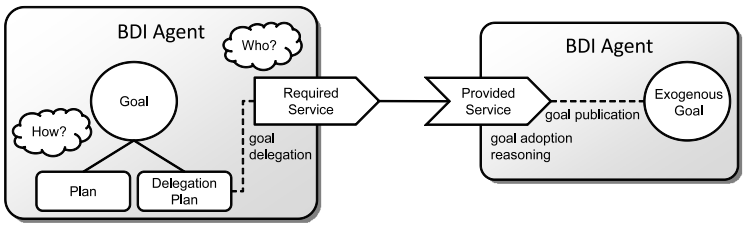

Fig. 4. Goal delegation with means-end reasoning

the internal architectures of the participating agents. These two requirements seem to be naturally contradicting as using notions such as shared goals or plans requires all possible goal adopters to understand that notion and in this way restricts the heterogeneity of the participating agents.

The proposed solution path builds on the idea of having a common black-box view for all kinds of agents (BDI, task-based, etc.) that adds provided and required service specifications known from component based software engineering. These required and provided service descriptions are based on interface definitions with method signatures so that the interoperability of different components is clearly defined. The idea now is to treat goal delegation as a process of three perspectives that can be considered separately. From the perspective of the delegating agent, which is assumed to be a BDI agent, a mechanism is added to route a goal to another agent, After having delegated the goal it is guaranteed that a result of this delegation process is eventually received and the goal state is updated. From the perspective of the receiving agent goal delegation may lead to the exogenous creation of a goal from another agent. The receiving agent can reason about goal acceptance and afterwards can process the goal in the same way as it does with its own goals. In between the goal is not explicitly represented - hence "goal delegation without goals" - but is abstracted to a simple service invocation, i.e. the delegating agent invokes one of its required services. This required service is potentially dynamically bound to the provided service of another agent which if being BDI automatically maps the call to a goal.

This general invocation scheme is graphically depicted in Fig. 3. It further illustrates that on the receiving side the service must not recreate a goal but can also be perceived as normal service invocation. In this way also other types of agents, like task-based ones or even other types of systems such as web services can be naturally used as goal executors.

3.1 Reasoning Integration

As already presented in the introduction, goal delegation makes the BDI reasoning process of an agent more complicated, because 'who' and 'when' questions need to be answered and it is not clear at what points in the reasoning process this should be done. Rather, it seems that a very flexible approach is needed that allows for answering these questions at different stages in the decision process, i.e. during goal deliberation as well as as part of the means-end reasoning. In

this paper we will not present a complete solution to this intricate aspect but will approach it with a conservative extension to the means-end reasoning part only. To install goal delegation at the mean-end reasoning phase a new generic delegation plan is introduced (cf. Fig. 4), which actually performs the delegation by automatically searching and invoking one or more services according to the user specification. In this way the means-end reasoning process determines if and when goal delegation in used. Plan priorities can be employed to determine the relation between own plans and goal delegation, i.e. if it should be tried to outsource a goal as soon as possible or rather as last available option. In addition pre- and context conditions can be used to define in which situations goal delegation is applicable. On the other side the incoming service call is handled by a generic goal creation service, which first checks if goal adoption is wanted by the agent and afterwards creates the exogenous goal for further processing.

In case the agent has decided to delegate a goal to another agent the question arises to which agent is should be sent if more than one option is available. It should be possible to base this decision upon functional as well as non-functional criteria. In our proposed solution the decision is not part of the goal delegation mechanism but belongs to the underlying service matching between required and provided services of agents. A required service specification is functionally defined via a service interface and additionally declares a search scope (e.g. platform or global scope) in which the search is performed. Depending on the required service specification, the search for suitable service providers (and hence goal delegation targets) can be done dynamically on each request or only once in case of a static scenario with fixed sets of service providers. Currently, non-functional criteria cannot be explicitly specified as part of the required service, but only by using custom classes that extend the search mechanism. Declarative specifications of non-functional criteria are an important part of future work.

3.2 Implementation

The goal delegation concept has been implemented using the Jadex agent platform [5]. The specification details will be presented in the following and they will be further illustrated by a small example application. The artificial scenario is called the money painter application. It consists of two agent types. The first, called rich agent, has the objective to become rich by getting a certain amount of money. It decomposes this goal to get the specified amount of money by creating a number of subgoals each responsible for getting one euro. To get one euro two different means are available. Either the agent can decide to paint one euro by itself or it can delegate the goal to paint one euro to another agent type called painter. It has to be noted that each agent can only paint one euro at the same time and it also needs some time to accomplish this task. In this scenario painter agents refuse to work on a new task as long as they are busy with an old one.

At the initiator agent side (i.e. an agent that wants to delegate a goal) merely a small extension to the plan element is necessary that allows for specifying a goal delegation plan. The delegation plan description differs from usual plan

```
01: <goals>
02:   <achievegoal name="get_one_euro" recur="true" recurdelay="1000">
03: </goals>
04:
05: <plans>
06:   <plan name="let_another_paint_one_euro">
07:     <body service="paintservices" method="paintOneEuro"/>
08:     <trigger>
09:       <goal ref="get_one_euro"/>
10:     </trigger>
11:   </plan>
12: </plans>
13:
14: <services>
15:   <requiredservice name="paintservices" class="IPaintService" multiple="true">
16:     <binding dynamic="true", scope="platform">
17:   </requiredservice>
18: </services>
```

Fig. 5. Cutout of the rich agent file

descriptions in the way the plan body is defined.[1] For normal plans, a class name is used to refer to the class that should be instantiated and executed. In case of a delegation plan, a service name and a method name have to be declared. The service name is a local name that corresponds to a required service definition from the services section of the agent and the method name is used to identify the method of the required service that should be called when a goal delegation is started.

In Fig. 5 a cutout of the rich agent is depicted. It can be seen that a goal to get one euro (lines 1-3), a delegation plan (lines 5-12) and a required service definition (lines 14-18) need to be specified. For each get one euro goal instance that agent possesses (creation not shown) it tries to accomplish it by executing the delegation plan (cf. the plan trigger in lines 8-10). This plan uses the new body description introduced above and links the delegation to the paintOneEuro method of the required service called paintservices. The required service specification defines the type of service should be bound (here of type IPaintService, line 15), that all instances of that services should be returned (multiple, line 15) and dynamic search should be applied (line 16) and that all components from the agent platform should be included into the search (scope platform, line 16). In case the delegation plan fails due to the fact that all painters are busy, the goal will pause and retry after one second (recur settings, line 2).

At the participant side (i.e. the agent that handles a goal delegation service call) an extension has been introduced to publish a goal as specific service method. Furthermore, a parameter mapping has been introduced that can be used to describe how a service argument is mapped to a parameter of a goal. In this way the method arguments can be passed to the goal also selectively or in a different order.

In Fig. 6 a cutout of the painter agent is shown. It owns a belief with name painting (line 1-3) of type boolean that indicates whether the agent is currently

[1] The plan body refers to the class that implements the domain logic of a plan. If the BDI interpreter executes a plan as first step it creates the plan body and afterwards executes it stepwise.

```
01: <beliefs>
02:   <belief name="painting" class="boolean">
03: </beliefs>
04:
05: <goals>
06:   <achievegoal name="get_one_euro">
07:     <publish class="IPaintService" method="paintOneEuro">
08:   </achievegoal">
09: </goals>
10:
11: <plans>
12:   <plan name="paint_one_euro">
13:     <body class="PaintOneEuroPlan"/>
14:     <trigger>
15:       <goal ref="get_one_euro"/>
16:     </trigger>
17:     <precondition>!beliefbase.painting</precondition>
18:   </plan>
19: </plans>
```

Fig. 6. Cutout of the painter agent file

busy with painting a euro. Furthermore, it has one goal (lines 5-9) for painting a euro and a corresponding plan (lines 11-19). The goal is equipped with the new publish declaration, which defines it as an exogenous goal that is instantiated when a method call ("paintOneEuro") to the IPaintService is received (line 7). Always when a new goal is created, goal processing will start and look for a suitable plan. In this example the agent only has one fitting plan that reacts to the goal (cf. plan trigger in line 15). Furthermore, a precondition is used to check whether the agent is already painting (line 17). The painting belief is modified by the painting plan itself when it starts and ends working. If a goal is created and the agent is busy the precondition evaluates to false and the plan cannot be used. As the agent has no other plan the goal is considered as failed and the service caller is automatically notified that goal delegation did not succeed.

4 Conclusion

This paper has tackled the question how goal delegation can be introduced without violating the OCMAS principles and forcing all agents of a multi-agent system to be BDI agents. The proposed solution distinguishes between the perspectives of the delegating agent, the multi-agent layer, and the receiving agent. Only at the delegating and possibly at the receiving agent goals are explicitly represented, whereas in between no goals occur and service calls are used as interaction means. As services are a common property of all kinds of agents (and especially of active components) the receiving side can interpret them in different ways, e.g. a BDI agent can recreate the goal, while a task based agent can execute a behavior. The contract between both sides is kept minimal and only requires the receiver to answer the service call with a result or an exception. This provides interoperability between BDI agents and other agent types as well as legacy systems, e.g. using web services. Taken together, the contributions of this paper therefore achieve *goal delegation transparency* on the intra-agent and inter-agent level. On the intra-agent level, transparency is achieved by treating

endogenous and exogenous goals the same way. On the inter-agent level, goals are made transparent using service calls. In future work, especially the development of a new BDI architecture will be targeted, which is able to use goal delegation at different stages and not only as part of means-end reasoning.

References

1. M. Barbieri and V. Mascardi. Hive-bdi: Extending jason with shared beliefs and stigmergy. In *Proc. Int. Conf. on Agents and Artificial Intelligence (ICAART)*. SciTePress, 2011.
2. F. Bergenti, L. Botelho, G. Rimassa, and M. Somacher. A FIPA compliant Goal Delegation Protocol. In *Proc. Workshop on Agent Communication Languages and Conversation Policies (AAMAS 2002)*, Bologna, Italy, 2002.
3. M. Bratman. *Intention, Plans, and Practical Reason*. Harvard Univ. Press, 1987.
4. L. Braubach and A. Pokahr. Representing long-term and interest bdi goals. In *Proc. of (ProMAS-7)*, pages 29–43. IFAAMAS Foundation, 5 2009.
5. L. Braubach and A. Pokahr. Addressing challenges of distributed systems using active components. In *Proc. of the Int. Symp. on Intelligent Distributed Computing (IDC 2011)*, pages 141–151. Springer, 2011.
6. L. Braubach, A. Pokahr, D. Moldt, and W. Lamersdorf. Goal Representation for BDI Agent Systems. In *Proc. of (ProMAS 2004)*, pages 44–65. Springer, 2005.
7. P. R. Cohen and H. J. Levesque. Teamwork. Technical Report Technote 504, SRI International, Menlo Park, CA, March 1991.
8. J. Ferber, O. Gutknecht, and F. Michel. From Agents to Organizations: an Organizational View of Multi-Agent Systems. In *Proc. of Int. Workshop on Agent-Oriented Software Engineering IV (AOSE 2003)*, pages 214–230. Springer, 2003.
9. M. Georgeff and A. Lansky. Reactive Reasoning and Planning: An Experiment With a Mobile Robot. In *Proceedings of the 6th National Conference on Artificial Intelligence (AAAI 1987)*, pages 677–682. AAAI, 1987.
10. M. Hashmi and A. El Fallah Seghrouchni. Coordination of temporal plans for the reactive and proactive goals. *Web Intelligence and Intelligent Agent Technology*, 2:213–220, 2010.
11. N. Jennings and E. Mamdani. Using Joint Responsibility to Coordinate Collaborative Problem Solving in Dynamic Environments. In *AAAI*, pages 269–275, 1992.
12. E. Norling. Folk Psychology for Human Modelling: Extending the BDI Paradigm. In *Proc. Int. Conf. Autonomous Agents and Multiagent Systems (AAMAS)*, 2004.
13. A. Pokahr, L. Braubach, and W. Lamersdorf. A Flexible BDI Architecture Supporting Extensibility. In *Proc. of the Int. Conference on Intelligent Agent Technology (IAT 2005)*, pages 379–385. IEEE Computer Society, 2005.
14. A. Pokahr, L. Braubach, and W. Lamersdorf. A goal deliberation strategy for bdi agent systems. In *Proc. of Conf. on Multi-Agent System TEchnologieS (MATES-2005)*. Springer, 2005.
15. A. Rao and M. Georgeff. BDI Agents: from theory to practice. In *Proc. of the Int. Conf. on Multi-Agent Systems (ICMAS)*, pages 312–319. MIT Press, 1995.
16. M. Scafes and C. Badica. Distributed goal-oriented reasoning engine for multi-agent systems: Initial implementation. In *Intelligent Distributed Computing III*, pages 305–311. Springer Berlin / Heidelberg, 2009.