Representing Long-Term and Interest BDI Goals

Lars Braubach and Alexander Pokahr

Distributed Systems and Information Systems,
Computer Science Department, University of Hamburg, Germany
`{braubach|pokahr}@informatik.uni-hamburg.de`

Abstract. In BDI systems, agents are described using mentalistic notions such as beliefs and goals. According to the intentional stance this helps specifying and understanding complex behavior, because the system is made up of folk psychological concepts that humans naturally tend to use for explaining reasoning and behavior and therefore can easily grasp. To close the gap between the natural usage of the term goal and its operationalization within agent systems, BDI goals should reflect the typical characteristics of goals in the folk psychological sense, which is not completely the case for existing BDI goal representations. Hence, in this paper desirable features of BDI goals are presented and important aspects that are currently not covered in existing specifications are further elaborated. Concretely, the representation and processing of BDI goals is extended supporting also *long-term* and *interest* goals. The usefulness of the newly gained expressivity will be illustrated by an example application, implemented in the Jadex BDI agent system.

1 Introduction

A concise definition of the term goal is extraordinarily hard to find, so that it is used in many psychological and artificial intelligence articles without a strict definition and with partially different meanings [2]. The main difficulty of the definition problem arises from the fact that in order to be useful for a variety of application areas, a definition has to reveal the term's essence without being too strict. Typically, definitions in the context of planning [10] and also w.r.t. specific theories for intentional agents (cf. [7, 20]) tend to be to be too narrow and often reduce the meaning of a goal to a desirable world state that needs to be achieved. A recent attempt in the area of multi-agent systems proposes the following definition: "a goal is a mental attitude representing preferred progressions of a particular multi-agent system that the agent has chosen to put effort into bringing about" [29]. Even though the definition is broader than the ones mentioned before and allows capturing different kinds of goals such as achievement or maintenance, it is already quite restrictive. Firstly, it uses the term "preferred progressions", which is not suitable for all kinds of goals. In case of e.g. avoidance goals [28] doing nothing could be better than doing any of the available actions. Secondly, especially the last part of the definition limits its usability by requiring an agent to put effort into the goal achievement. If an agent has no means to pursue a goal at some moment in time, that doesn't mean that it cannot possess the goal. It could just sit and wait until it gets the possibility to act towards the goal or just wait passively for its achievement [2].

This paper proposes using a property-based view of goals. The rationale behind this view is to abandon the objective to introduce a strict separation of what is a goal and what is not a goal, but to see goals as a tool for analyzing and specifying systems. Considering a goal by its characteristics may further help understanding how it should be represented and processed and avoids definitions with limited applicability. Note, that a similar procedure led to the agreed upon characterizations of the term agent [23, p. 33], especially the weak/strong notion of agency [31], which is based on the characterizing properties autonomy, reactivity, proactivity, social abilities, and mentalistic notions.

In the next Section 2, the characteristics of BDI goals will be presented. Thereafter, in Section 3 existing goal representations will be extended under consideration of the uncovered aspects of the previous section. In this respect, special attention will be paid to long-term and interest goals, which have in common that they both might not lead to actions immediately respectively at all. In Section 4 the usefulness of the extended goal semantics will be illustrated by a booktrading example application. Finally, in Section 5 a conclusion is given and some aspects of future work are presented.

2 Characteristics of BDI Goals

After having shown the difficulties in defining the term goal precisely, in this section characteristics of goals will be discussed especially in the context of the belief-desire-intention (BDI) model, because it is one of the predominant agent architectures today [25]. In addition to identifying those properties we will examine the degree to which the existing PRS architecture addresses these issues and which deficiencies still exist.

Before concrete characteristics will be presented, the three different BDI perspectives – philosophical, logical and software technical – will be sketched, because they use slightly different terms. In the original work of Bratman [3] the most general meaning of the goal concept in the form of desires is introduced. A desire represents the motivational reasons for an agent's acting. Bratman allows desires to be quite vague and also conflicting so that an agent has to decide to commit to some of its desires making them concrete intentions, which are considered as conflict-free. In contrast to this perspective, in the logical interpretation of [21] no desires but only goals are considered. They are represented as logical formulae of a branching time logic. Here goals are seen as a subset of belief-accessible worlds and are therefore per se declarative and of type achieve. Hence, in this perspective the only difference between goals and beliefs is the optative vs. indicative interpretation of the logical expression. Considering the software engineering perspective the goal concept has been further simplified and reduced to be some kind of volatile event. In the procedural reasoning system (PRS) architecture [9] and AgentSpeak(L) [19] those kinds of goal events are only used for triggering suitable plans and do not posses an explicit representation.

In the following sections an initial attempt is made to identify the most important goal properties from the existing literature. The first five properties have already been identified in a seminal paper of Rao and Georgeff [22]. In addition to these basic properties several further desirable characteristics can be found in the agent as well as social science literature. Note that the further characteristics mainly aim to isolate goal properties that are useful for a software engineering perspective. The discussion in all sections will first explain the meaning of the property and will then discuss its support in the context of the original PRS architecture as well as recent advancements.

Persistent Persistent goals are entities that have a persistent character, which means that they exist over a period of time. In volatile environments it is important for an agent to commit to its goals and give them up only for good reasons. Hence, the persistence of goals serves for stability in an agent's behavior [21].

The persistency of goals intentions has not been defined exactly for the PRS architecture. Instead it has often been discussed in the context of commitment strategies [21, 30, 26], whereby such strategies determine to what extent an agent should keep pursuing its current goals. In the literature a distinction between blind (fanatical), single-minded and open-minded commitment strategies have been proposed. These strategies implement different strengths of commitments. The most committed agent is blindly committed and sticks to its goals until they finally succeed. A single-minded agent also abandons goals on failure and an open-minded agent can get rid off goals by also dropping them intentionally. Experiments have shown that the efficiency of those strategies is heavily dependent on the existing environmental dynamics, i.e. the faster an environment changes the more flexibility an agent has to adapt its goals [13]. Compared to human decision making, especially an open-minded strategy seems to be promising for truly goal-directed agents, because BDI agents should be enabled to reason about their goals (support goal deliberation) and drop them any time, if an important reason occurs.

Consistent In order to describe the consistency property of goals it is necessary to distinguish between the actively pursued goals called adopted goals and the currently inactive goals called candidate goals resp. options [17, 25]. The adopted goals of an agent should be consistent with each other at any point in time in the sense that all goals should be achievable concurrently. An agent should therefore refrain from pursuing a goal, which it thinks stays in conflict with some adopted goals. In case the agent wants to urgently adopt this new goal, a goal deliberation process has to decide if a conflict-free goal set can be found and possibly then has to drop some of the already adopted goals.

The original PRS architecture assumes goals to be always consistent and does not take into account the first phase of practical reasoning, i.e. goal deliberation [16]. This shifts the task for ensuring conflict-freeness to the application layer so that the developer has to cope with these tedious issues directly. This deficiency of the original architecture has been subject of intensive research yielding proposals for supporting also the goal deliberation phase [27, 17].

Possible An adopted goal should be possible to pursue, i.e. an agent should be convinced that it can achieve a goal and it does not contradict its current beliefs.

This property ensures that an agent does not adopt goals it cannot achieve, but it does not guarantee that a goal can always be successfully pursued.

In PRS the conformance of goals and beliefs cannot be directly ensured due to the event-based character of goals. In an indirect way, the pre- and context conditions of plans help guaranteeing that a goal can only be (successfully) processed when those conditions are valid. Nonetheless, as plans could be applicable for different goal types this support is not fully adequate and should be complemented with checks on the goal level.

Known/Explicit A rational agent should be aware of all its goals (candidate and adopted), because this is a necessary prerequisite for any kind of reasoning on its objectives [17].

In PRS an agent knows its goals as long as they are part of the means-end reasoning. The initialization of a goal normally results from a subgoal call within a plan and leads to the generation of a goal event. This event is saved within the corresponding intention of the calling plan, often called the intention stack [19]. During the processing of the goal via different plans the goal is kept in the stack and is e.g. used to save information about its execution state (e.g. which plans have already been tried) [12]. This event-based representation is not expressive enough for supporting goal deliberation and hence explicit goal representations have been proposed [5, 29].

Unachieved An agent should only pursue goals, which it assumes to be unachieved. This kind of behavior will ensure that no unnecessary actions will be initiated and resources will not be wasted.

This property is realized by the PRS architecture via testing the achievement condition of a goal before plan processing is started. In case the condition is immediately true, the goal is considered as succeeded and no plans will be executed. Even though this mechanism ensures correct achieve goal processing, it cannot directly be applied to other goal kinds such as maintain. In order to support this property generically its meaning needs to be adapted to different goal kinds guided by the idea to avoid means-end reasoning if the goal does not require it. E.g. a query goal, that is responsible for information retrieval, should only initiate plan processing when the requested data cannot be directly extracted from the beliefbase [5].

Producible/Terminable In order to be useful for agents, goals should be producible and terminable [8]. For the creation as well as the termination of goals, procedural as well as declarative means should be supported, i.e. an agent should be enabled to create/terminate a goal from a plan as well as due to situational reasons.

In PRS goals can typically be created only in a procedural way by issuing subgoal calls from within a plan. The ex post termination of goals is not possible at all due to their implicit representation. A plan is only allowed to issue one subgoal at a time (intention stack) and the subgoal call itself passivates the original plan and lets it wait until the subgoal processing has finished. Declarative means for creating and terminating goals have been introduced e.g. in [5] and rely on a generic representation for the various goal kinds.

Suspendable In addition to the termination of goals it can be advantageous in certain situations to suspend the pursuit of a goal [8, 5, 24, 29], e.g. if the agent has devoted considerable effort into bringing about the goal and cannot continue to pursue it due to a conflict with another possibly more important goal. Further use cases for goal suspension are detailed in [24]. The suspension of a goal should allow saving the current processing state of that goal and continue from there when it gets activated again [24].

The original PRS architecture does not support the suspension of goals. In [5] and [29] the suspension of goals has been addressed by introducing goal lifecycle states, which allow differentiating between active and suspended goals in an agent. The underlying concept is extended in [24] by also addressing the suspension of plans that are executed whilst their goal is suspended.

Variable Duration Intelligent behavior is based on a combination of strategic and tactical action. Strategic behavior is based on long-term goals, which persist over longer time periods and are typically challenging to achieve, e.g. they need several milestones being reached before the goal as a whole can be tackled. Tactical behavior is in many cases based on short-term goals or even reflexes. Hence, short-term goals often only live for the short moment in which the reason for their creation, e.g. an environmental change, was detected. These kinds of goals are closely linked to (physical) actions and exhibit event-based character.

The PRS architecture focuses exclusively on means-end reasoning and therefore goals only exist during their execution phase, i.e. a goal is held as long as plans are executed for this goal. If no (more) plans are available for a goal the means-end reasoning phase has finished and the goal is considered as finished. This does not necessarily mean that goals are always short-term in PRS, because the corresponding plans could be long-lasting. Nevertheless, using long-term goals requires that goal processing can be immediately started, which is not always desired. Furthermore, goals in PRS are typically not of strategic nature, because conflicts between them cannot be detected and long-lasting goals would considerably increase the probability of goal clashes. Hence, the traditional PRS idea is more centered on realizing short-term goal-driven behavior.

Action Decoupled Goals express and incarnate motivations with respect to a specific situation. This motivation can exist even if an agent cannot contribute actively to the goal achievement. These so called interest or passive goals do not directly lead to action execution, but should nonetheless be allowed to persist within an agent [2]. On the one hand, an agent might eventually gain new procedural knowledge for pursuing the goal [1] or on the other hand the goal might be fulfilled by a third party, e.g. other agents.

Due to the action-centeredness of PRS, interest goals cannot be represented. Instead goals are always part of an intention stack and cannot exist without that structure. The means-end reasoning of PRS ends as soon as no further plans can be executed. This also terminates the corresponding goal by popping the event from the intention stack.

2.1 Challenges

The aforementioned goal properties shape the complex nature of goals and also indicate in which directions the PRS architecture should be further extended

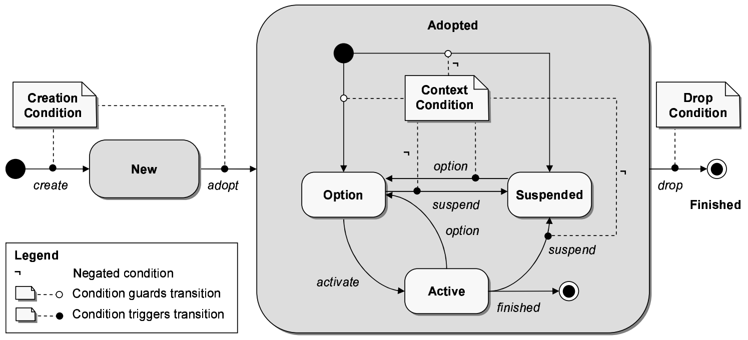

Fig. 1. Basic goal lifecycle

to support the full gamut of goals. Previous works mainly tackled aspects of explicit goal representation [26, 5, 29] and goal deliberation aspects [17, 16, 27]. Goal processing was examined with respect to PRS means-end reasoning [21, 6] and recently also concerning additional features such as goal suspension [24].

To our knowledge none of the existing works has tackled questions of long-term and interest goals for BDI agents, even though these kinds of goals represent a helpful extension for the conceptual canon of BDI agent programmers. The importance of interest goals is emphasized in the literature especially by the cognitive structure of emotions model (OCC - Ortony, Clore and Collins) [14], which assumes that three different goal types exist: “active goals” an agent can directly bring about by performing some actions (e.g. open a bottle), “interest goals” representing states of the world an agent would like to become reality but cannot do anything about pursuing them (e.g. make my soccer team win) and finally “replenishment goals”, which repeatedly spawn activities only on demand (e.g. keep healthy). The first and third OCC goal types are already covered by BDI achievement and maintenance goals [5], whereas no support for long-term and interest goals exists. Application cases for interest goals are all scenarios in which external factors (be it actors, processes or sth. else) are responsible for the agent’s goal achievement. Examples include conversation handling, where one participant depends on its communication partners [15] and acting in competitive multi-agent settings, where the actions of one agent can contribute to or thwart the goals of another one [11]. Hence, this paper investigates how long-term and interest goals can be represented and how the PRS architecture needs to be modified in order to allow their processing.

3 Long-Term and Interest Goals

To meet the set out requirements and enable a representation of long-term and interest goals in BDI agent systems, two properties should be ensured. First, goals need to be represented explicitly and separately from other BDI constructs such as events or procedural plans. Only with such an explicit representation, the long-term and action-decoupled goals can exist independently of short-lived events and concrete plans resp. actions. Second, an agent should stay committed to these kinds of goals, even if a goal cannot (immediately) be achieved. Otherwise, the agent will not know when to start acting towards the goal when the time comes (long-term goal) or to refrain from counterproductive actions until the goal is achieved (interest goal).

In the following, an extended goal representation will be introduced that is based on existing work on explicit goal representation and adds the necessary property of *long-term, action-independent commitment.* First, a well-accepted goal lifecycle model from the literature will be described as it forms the basis for the new goal representation. It will be shown, how long-term and interest goals fit into this general model. Second, the detailed processing of long-term and interest goals will be discussed. Moreover, it will be shown how the long-term and interest state of goals can be embedded in different goal types, such as achieve, perform, maintain. The section closes with considerations about the usage of long-term and interest goals.

3.1 Goal Lifecycle Model

Figure 1 shows a basic goal lifecycle introduced in [5]. This lifecycle divides the set of adopted goals of an agent according to the three possible substates *option*, *active* and *suspended* in order to support the goal deliberation and means-end reasoning phases [30]. Thereby, the means-end reasoning of the agent operates on active goals only, i.e. only active goals can lead to the execution of plans and actions. Moreover, suspended goals are goals, which currently cannot be pursued due to an invalid context condition, while options are those goals, which the agent's deliberation mechanism has decided not to pursue (e.g. in favor of other more important goals). Although each goal can only be in exactly one of the substates at any point in time, the state of a goal can change, e.g. when changes happen to beliefs or other goals of an agent.

Based on this generic goal lifecycle, the characteristics of long-term and interest goals can be defined. Whenever a goal enters the active state, the agent will start the means-end reasoning process in order to find suitable means for pursuing the goal. Unlike usual short-term goals, which immediately lead to actions, for a long-term or interest goal it might be an appropriate means to actually do nothing at all. *Therefore, a long-term or interest goal is a goal, which can be active even without executing any plans for it.* The distinction between long-term in contrast to interest goals is merely one of future expectations. For a long-term goal, the agent expects to find suitable means somewhere in the future, while for an interest goal, the agent does not expect to be able to contribute to goal achievement, but still expects that the goal might be (automatically) achieved in the future, if the agent refrains from doing counterproductive actions.

Allowing for goals to be active even without suitable actions leads to some important advantages with respect to goal deliberation processes. Because the goal is among the active goals of an agent, it will be considered as such during the agent's goal deliberation. Therefore, other conflicting goals will not be activated, unless they are considered more important than a currently active long-term or interest goal. This example also shows that sticking to a long-term or interest goal even without a suitable plan does not contradict an open-minded commitment (cf. section 2), as the agent still can decide to abandon the goal at any time. Another advantage with respect to long-term goals is that for an active goal, the agent can adopt any suitable plan as soon as it becomes available/viable, and does not have to enter complex goal deliberation processes.

3.2 Processing Long-Term and Interest Goals

The previous section showed that the expressibility of the goal lifecycle model can be extended to allow for long-term and interest goals by supporting active goals even when there are (currently) no suitable plans. In general, not all goals are of the long-term/interest type, but usually should be abandoned, when no suitable plans can be found. Therefore, an agent needs to know which goals are of long-term/interest type and how to treat these goals differently from the "normal" short-term goals. The requirements for this special treatment can be condensed to the following questions:

When to stop, but not drop? The usual behavior is to fail a goal, when no suitable plans can be found once the goal has become active. For long-term/interest goals, the agent programmer needs a mechanism for overriding this behavior in such a way that the agent will stop the means-end reasoning, but not drop the goal.

When to continue processing? Some time after a long-term goal has stopped processing, the agent should re-check the availability of plans. For efficiency and effectiveness the programmer should have fine-grained, yet simple control over the re-check activity for ensuring that the agent stays reactive in a changing environment, but avoiding the overhead of unnecessarily checking too frequently.

When to succeed/finally fail? Long-term as well as interest goals should not stay in the agent forever. For one, similar to short-term goals the agent should be able to detect when a goal has been achieved, e.g. because the environment has changed or some successful plan could finally be found. Moreover, because dropping unachievable long-term/interest goals is not done automatically, an agent programmer may want to explicitly state reasons for dropping a long-term or interest goal.

To capture the required functionality, a generic goal processing component is introduced (cf. Figure 2). This component is generic in the sense that it is independent of the concrete goal type (perform, achieve, maintain, etc.), but forms the conceptual and technical basis for processing of all goal types. The component is activated through the *in* edge, which is triggered depending on the goal type, e.g. when a goal becomes active (achieve) or a condition becomes violated (maintain). Regardless of how the component is activated, it will enter two nested loops, which are responsible for basic means-end reasoning and long-term/interest goal handling respectively. In the following sections it will be discussed, how this generic component gives answers to the questions raised above.

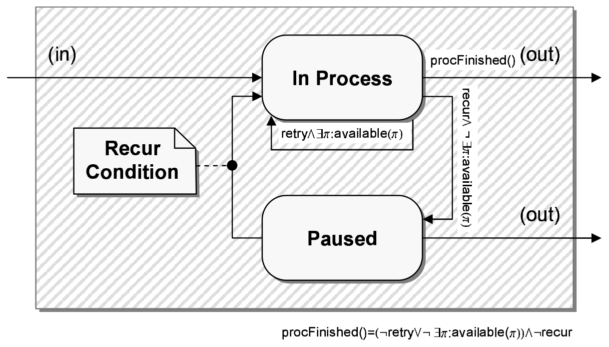

Fig. 2. Generic goal processing component

When to stop, but not drop? The inner loop is the "retry loop", which represents traditional PRS-style means-end reasoning as known from currently implemented BDI systems. This part is captured by the *In Process* state in the figure, in which usually the agent selects and executes a single plan for a goal.[1] If the goal is not finished after the plan has been executed ($procFinished()$), the retry loop continues, leading to the next plan being selected and executed. The retry loop iterates – unless retry behavior is disabled with the *retry* flag – until no more applicable plans are available for the goal ($retry \wedge \exists\pi : available(\pi)$).

To extend this basic means-end reasoning with the required functionality for handling long-term and interest goals, a new *Paused* state is introduced. This state effectively represents the means of "doing nothing" to achieve a goal. Extended means-end reasoning for long-term/interest goals therefore happens according to the outer "recur loop", which alternates between the *In Process* and *Paused* states. The first question posed above then becomes the question of when to move from the *In Process* to the *Paused* state. For this decision, the *recur* flag is introduced that an agent developer can set to true for a goal to be handled as a long-term/interest goal. Hence, the *Paused* state is entered, when the *recur* flag is set and no (more) plans are available ($recur \wedge \neg\exists\pi : available(\pi)$). Note that *Paused* and *In Process* are substates of the lifecycle state *Active* (cf. Figure 1), which means that paused goals suppress the execution of other conflicting, but less important goals e.g. according to the "easy deliberation strategy" [17].

When to continue processing? The continuation of processing forms the second part of the "recur loop", i.e. moving from the *Paused* state back to *In Process*. To allow fine-grained control over when an agent should reconsider the processing of long-term goals, three different specification means are supported – recur delay, recur condition, and recur action – that a developer can choose from or combine, depending on the application at hand. The recur delay of a goal is a simple mechanism for continuously looking for newly applicable plans becoming available. This allows the developer to specify a time interval, after which

[1] For brevity, we do not cover detailed fine-tuning of the means-end reasoning process, such as caching vs. recalculation of the applicable plan list (APL) or parallel execution of plans for the same goal (post-to-all), even though these variations are supported by the proposed model as well.

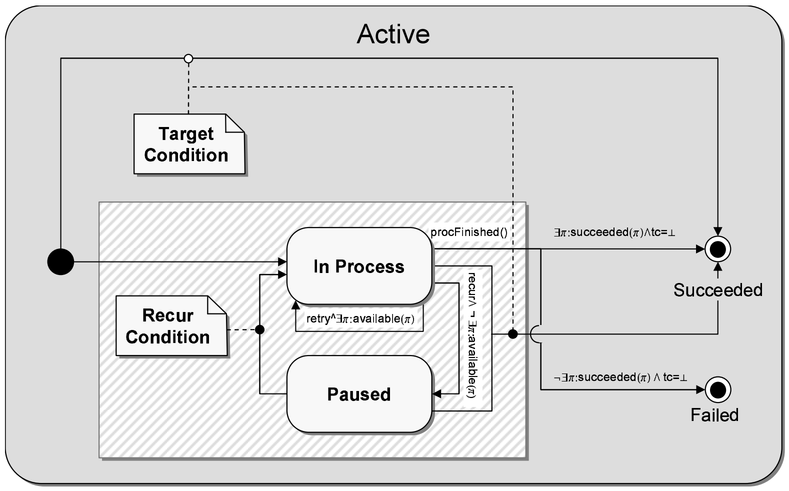

Fig. 3. Extended processing for achieve goals

processing of the goal should be restarted. A more advanced way is the *Recur Condition*, illustrated in the figure. The recur condition allows the specification of a world state (based on the agent's beliefs) that should cause reprocessing of the goal. Therefore it provides a declarative way for specifying the conditions, under which a reconsideration of the goal becomes worthwhile. The most flexible, but least automated way is an explicit recur action that can be manually invoked on a goal. Therefore, the developer can provide arbitrary code (e.g. inside a procedural plan), to determine when certain goals should be reconsidered and explicitly invoke the recur action on these as needed.

When to succeed/finally fail? In Figure 2, the two *out* edges from the *In Process* as well as *Paused* state determine possible ways of exiting the goal processing. First, it should be noted that as illustrated in Figure 2, goal processing would never finish when the *recur* flag is set to true. The lower *out* edge has no guard and therefore will never actively trigger, while the guard on the upper *out* edge is $\mathit{procFinished}\,() = (\neg \mathit{retry} \vee \neg\exists\pi : \mathit{available}\,(\pi)) \wedge \neg \mathit{recur}$, i.e. processing is only finished, when *retry* and *recur* are both false or when *recur* is false and no more plans are available. An answer to the question above, therefore cannot be given in the context of the generic goal processing component alone, but also needs to consider the generic goal lifecycle (cf. Figure 1) and the specifics of the different goal types, such as achieve and maintain.

The generic goal lifecycle of Figure 1 allows for several ways of exiting the *Active* state, which would cause the substates like *In Process* and *Paused* to be exited as well. First, the *Active* state can be exited due to goal deliberation issues, e.g. moving to the *Option* state when a more important but conflicting goal occurs or to the *Suspended* state, when the context of the goal becomes invalid. In both cases, processing of the goal would be stopped regardless of the goal being a short-term, long-term, or interest goal. Moreover, goals can be abandoned at any time, e.g. when the *Drop Condition* triggers or the goal is dropped manually. In the latter cases, the goal will finish before being achieved

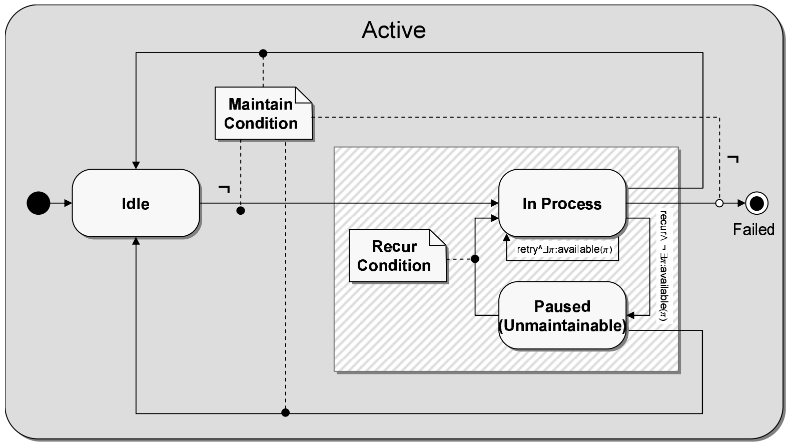

Fig. 4. Extended processing for maintain goals

and therefore can be considered as finally failed. How to determine goal success depends on the specific goal type and will be discussed in the next section.

3.3 Realization of Specific Goal Types

The extended reasoning process has been embedded into the four goal types perform, achieve, query, and maintain, as introduced in [5]. For space reasons, only the achieve and maintain goal types will be discussed in the following. Figure 3 shows the extended model for processing achieve goals. For achieve goals, the generic goal processing component is used as a basis and has been augmented by the *Target Condition*, which can trigger a transition from the *In Process* or *Paused* state to the *Succeeded* state. Moreover, achieve goals without target condition ($tc = \perp$) are also supported by two guards on the upper *out* edge and are considered succeeded, when at least one plan finishes successfully ($\exists \pi : \mathit{succeeded}(\pi) \wedge tc = \perp$) or failed otherwise ($\neg \exists \pi : \mathit{succeeded}(\pi) \wedge tc = \perp$).

For maintain goals (cf. Figure 4), the newly introduced *Paused* state has been mapped to the *Unmaintainable* state of the original processing model from [5]. Unlike the other goal types, a maintain goal does not start with processing immediately. Instead, the goal initially enters the *Idle* state and only moves to *In Process*, when the *Maintain Condition* is violated. While processing is active, a fulfilled maintain condition will move the goal back to the *Idle* state from both *In Process* and *Paused*. This semantics exactly resembles the original semantics from [5] (with *Unmaintainable* renamed to *Paused*), except when the *recur* flag is set to false the maintain goal will stop processing and fail, when none of the available plans is able to re-establish the maintain condition.

3.4 Usage of Long-Term and Interest Goals

The model presented above has been implemented in the Jadex agent framework [18]. One primary advantage of the model is that long-term and interest goals are represented just like other (short-term) goals. Therefore, their usage does not differ from these and the available mechanisms for creating and handling goals can be reused. In this respect, long-term and interest goals can be given to an agent, when it is born (initial goals) and can also be dynamically created at runtime based on creation conditions or inside plans. Plans can choose to dispatch goals as independent top-level goals that exist outside the scope of the plan or as subgoals, which will be automatically dropped when the plan finishes or is aborted. This control over the goal scope is especially important for long-term and interest goals, which potentially can reside inside an agent for a long period of time. Due to the unified representation of long-term and short-term goals, the plans that get executed in response to a long-term goal do not need to know about the nature of the goal and can be defined independently of the goal's nature. The next section will show how long-term and interest goals can be employed in the context of an illustrative example scenario.

4 Example Application

To illustrate how long-term and interest goals can be used in practice the book-trading scenario from [4] is used, where personal buyer and seller agents are responsible for trading books according to orders given by their principals. The participants use a market-based coordination strategy following the contract-net protocol for reaching agreements acceptable for both sides. It is assumed that buyers take the initiator role of the protocol, whereas sellers play the participant role. An order of a principal includes all relevant data needed for an agent to be able to buy resp. sell a book. Concretely, it contains the name of the book, the start and limit prices as well as a deadline at which the transaction has to be done at latest. The start price represents the acceptable price for an agent at the beginning of the negotiation. While time passes and the deadline approaches a linear price adaptation strategy is used to calculate the currently acceptable price between start and limit price.[2]

Buy or sell orders are entered by the principals through the user interface for each agent. For each of these orders the agents form purchase resp. sell goals, which express the motivations of the principals. For a buyer agent a purchase book goal represents the long-term goal for buying a book, according to the definitions in the order. It is of long-term nature, because initially there might be no seller available that offers the book at the desired price. Nevertheless, the agent should not drop the goal in this case, but instead wait for new sellers to appear or the book gets cheaper at the available sellers. The purchase book goal is therefore modeled as an active achieve goal, which has the purpose of initiating negotiations with potential sellers in fixed time intervals until the book could be bought or the deadline has passed. At the top of Figure 5 the concrete implementation of the *purchase_ book* goal is illustrated. The *purchase_ book* goal (lines 1-5) contains the principal's order in a corresponding parameter (line 2). It

[2] In case of a buyer the start price is lower than the limit price and is continuously increased. The opposite behavior is used by sellers.

```
 1  <achievegoal name="purchase_book" recur="true" recurdelay="10000">
 2    <parameter name="order" class="Order"/>
 3    <targetcondition>Order.DONE.equals($goal.order.state)</targetcondition>
 4    <dropcondition>$beliefbase.time > $goal.order. deadline </dropcondition>
 5  </achievegoal>
 6
 7  <achievegoal name="sell_book" recur="true">
 8    <parameter name="order" class="Order"/>
 9    <targetcondition>Order.DONE.equals($goal.order.state)</targetcondition>
10    <dropcondition>$beliefbase.time > $goal.order. deadline </dropcondition>
11  </achievegoal>
```

Fig. 5. Purchase and sell book goals

is defined as a long-term goal via the recur flag (line 1), which enables the long-term processing loop. In addition, it is made active by specifying the recurdelay, stating that each 10 seconds (10000 ms.) a new negotiation round is started. If the negotiation is successful, the state of the buy order will change to *DONE*, which is tracked by the goal's targetcondition (line 3). On the other hand, the dropcondition (line 4) monitors the deadline and lets the goal automatically fail, when no negotiation result could be achieved before the deadline ends.

The seller's *sell_book* goal (lines 7-11) is realized in a very similar way. The main difference here is that it is a pure interest goal, i.e. it is assumed that sellers passively wait for buy requests to come in and match them with their existing sell goals. An interest goal is specified by activating the recur flag without specifying means for recur initiation, i.e. no recur delay and no recur condition (line 7). The seller agent has no plans for achieving its *sell_book* goal. But the agent does have a plan for reacting to buy book requests from buyer agents and engaging in a corresponding negotiation. When such a negotiation comes to a result, the target condition (line 9) is fulfilled and the *sell_book* goal is achieved.

This example shows how long-term and interest goals can facilitate the high-level and natural modeling of BDI scenarios. The availability of these conceptual abstractions allows for a direct mapping of buy and sell orders to goals, which are present as long as the corresponding orders are relevant. Using only standard BDI goals would require additional error prone code: The buyer's long-term goal could be emulated using a long-term plan, which captures the recur semantics and initiates negotiations in certain intervals. The seller's interest goal would have to be mapped to other structures such as beliefs leading to a rather artificial design, which would also differ a lot from the buyer side.

5 Conclusion

This paper has tackled the representation and processing of long-term and interest goals. In order to understand what make-up goals in BDI agent systems definitions of the term goal have been reviewed. This review mainly revealed that the essence of the term is very hard to capture, because on the one hand different perspectives on BDI exist – ranging from philosophical to implementational – and on the other hand many definitions tend to be too restrictive by overlooking important goal aspects. As a result, this paper has proposes characterizing goals according to their typical properties, similar to property-based definitions of the term agent. Such a specification is more practically useful as it has the aim of supporting the goal-oriented software specification and is not targeted towards a clear-cut separation of what is a goal and what isn't.

Based on the characterization of goals it has been shown that long-term and interest goals are not currently considered in the modeling and implementation of BDI systems. These goals are typically long-lasting, whereby they can be active without having plans executed for them. In the case of interest goals, an agent does not possess plans for their fulfillment, whereas in the case of long-term goals, plans could exist but not fit to the situation at activation time. Their main relevance can be seen in the more strategic nature of these goals allowing also long-lasting objectives to be expressed. Especially, in combination with goal deliberation mechanisms such strategic goals represent useful extensions to traditional BDI goals, which are of rather tactical, short-term nature.

The main contribution of this paper consists of providing a representation and processing mechanism for long-term and interest goals. The new mechanism builds on the explicit goal representation from [5], and introduces a generic goal processing component for short- and long-term means-end reasoning. This component introduces two control loops, one responsible for traditional plan selection and execution and one responsible for pausing the execution of long-term goals in the case where no processing is currently possible. This generic component can be used to support different goal kinds such as achieve and maintain. The concepts of long-term and interest goals have been implemented within the Jadex BDI agent system and already been used for building different example applications, such as the presented booktrading scenario.

In future work we plan to further extend the expressiveness of goal specifications available for describing BDI agent systems. In this respect one important goal type are soft-goals, which represent non-functional properties and have therefore been excluded from the implementation layer so far.

References

1. Ancona, D., Mascardi, V., Hübner, J., and Bordini, R. Coo-AgentSpeak: Cooperation in AgentSpeak through Plan Exchange. In *Proceedings of AAMAS'04* (2004), ACM press, pp. 698–705.
2. Beaudoin, L. *Goal Processing in Autonomous Agents.* PhD thesis, Mar. 1995.
3. Bratman, M. *Intention, Plans, and Practical Reason.* Harvard Univ. Press, 1987.
4. Braubach, L., and Pokahr, A. Goal-oriented interaction protocols. In *Proceedings of MATES'07* (2007), Springer, pp. 85–97.
5. Braubach, L., Pokahr, A., Moldt, D., and Lamersdorf, W. Goal Representation for BDI Agent Systems. In *Pr. of ProMAS04* (2005), Springer, pp. 44–65.
6. Busetta, P., Howden, N., Rönnquist, R., and Hodgson, A. Structuring BDI Agents in Functional Clusters. In *Proc. of ATAL'99* (2000), Springer, pp. 277–289.
7. Cohen, P. R., and Levesque, H. J. Intention is choice with commitment. *Artificial Intelligence 42* (1990), 213–261.

8. Dignum, F., and Conte, R. Intentional Agents and Goal Formation. In *Proceedings of ATAL'97* (1997), pp. 231–243.
9. Georgeff, M., and Lansky, A. Reactive Reasoning and Planning: An Experiment With a Mobile Robot. In *Proceedings of AAAI'87* (1987), AAAI, pp. 677–682.
10. Ghallab, M., Nau, D., and P.Traverso. *Automated Planning: Theory and Practice*. Morgan Kaufmann Publishers, May 2004.
11. Johns, M., and Silverman, B. G. How Emotions and Personality Effect the Utility of Alternative Decisions: A Terrorist Target Selection Case Study. In *Proceedings of SISO'01* (2001), pp. 55–64.
12. Kinny, D. Algebraic specification of agent computation. *Journal Applicable Algebra in Engineering, Communication and Computing 16*, 2-3 (July 2005), 77–111.
13. Kinny, D., and Georgeff, M. Commitment and effectiveness of situated agents. In *Proceedings of IJCAI'91* (Feb. 1991), pp. 82–88.
14. Ortony, A., Clore, G. L., and Collins, A. *The Cognitive Structure of Emotions*. Cambridge University Press, 1988.
15. Pasquier, P., Dignum, F., Rahwan, I., and Sonenberg, L. Interest-based negotiation as an extension of monotonic bargaining in 3apl. In *PRIMA'06* (2006), Springer, pp. 327–338.
16. Pokahr, A., Braubach, L., and Lamersdorf, W. A Flexible BDI Architecture Supporting Extensibility. In *Proc. of IAT'05* (2005), IEEE, pp. 379–385.
17. Pokahr, A., Braubach, L., and Lamersdorf, W. A goal deliberation strategy for bdi agent systems. In *Proceedings of MATES'05* (2005), Springer, pp. 82–94.
18. Pokahr, A., Braubach, L., and Lamersdorf, W. Jadex: A BDI Reasoning Engine. In *Multi-Agent Programming: Languages, Platforms and Applications* (2005), Springer, pp. 149–174.
19. Rao, A. AgentSpeak(L): BDI Agents Speak Out in a Logical Computable Language. In *Proceedings of MAAMAW 1996* (1996), Springer, pp. 42–55.
20. Rao, A., and Georgeff, M. Asymmetry thesis and side-effect problems in linear-time and branching-time intention logics. In *Proc. of IJCAI'91* (1991).
21. Rao, A., and Georgeff, M. BDI Agents: from theory to practice. In *Proceedings of ICMAS 1995* (1995), MIT Press, pp. 312–319.
22. Rao, A. S., and Georgeff, M. P. An abstract architecture for rational agents. In *Proceedings of KR'92* (1992), pp. 439–449.
23. Russell, S., and Norvig, P. *Artifical Intelligence: A Modern Approach*. Prentice-Hall, 2003.
24. Thangarajah, J., Harland, J., Morley, D., and Yorke-Smith, N. Suspending and resuming tasks in bdi agents. In *Proc. of AAMAS'08* (2008).
25. Thangarajah, J., Harland, J., and Yorke-Smith, N. A soft cop model for goal deliberation in a bdi agent. In *Proceedings of CP'07* (sep 2007).
26. Thangarajah, J., Padgham, L., and Harland, J. Representation and Reasoning for Goals in BDI Agents. In *Proc. of ACSC'02*.
27. Thangarajah, J., Padgham, L., and Winikoff, M. Detecting and Avoiding Interference Between Goals in Intelligent Agents. In *Proc. of IJCAI'03* (2003).
28. van Lamsweerde, A. Goal-Oriented Requirements Engineering: A Guided Tour. In *Proceedings of RE'01* (2001), IEEE Press, pp. 249–263.
29. van Riemsdijk, B., Dastani, M., and Winikoff, M. Goals in agent systems: a unifying framework. In *Proceedings of AAMAS'08* (2008), pp. 713–720.
30. Wooldridge, M. *Reasoning about Rational Agents*. MIT Press, 2000.
31. Wooldridge, M. *An Introduction to MultiAgent Systems*. Wiley & Sons, 2001.