Enhancing Training by Using Agents with a Theory of Mind*

Maaike Harbers
Utrecht University
P.O. Box 80.089, Utrecht
The Netherlands
maaike@cs.uu.nl

Karel van den Bosch
TNO Defence, Security & Safety
P.O. Box 23, Soesterberg
The Netherlands
karel.vandenbosch@tno.nl

John-Jules Meyer
Utrecht University
P.O. Box 80.089, Utrecht
The Netherlands
jj@cs.uu.nl

ABSTRACT

Virtual training systems with intelligent agents are often used to prepare people who have to act in incidents or crisis situations. Literature tells that typical human mistakes in incidents and crises involve situations in which people make false assumptions about other people's knowledge or intentions. To develop a virtual training system in which correctly estimating others' knowledge and intentions can be trained in particular, we propose to use agents that act on the basis of their own mental concepts, but also on mental states that they attribute to other agents. The first requirement can be realized by using a BDI-based agent programming language, resulting in agents which behavior is based on their goals and beliefs. To make these agents able to attribute mental states to others, they must be extended with a so-called Theory of Mind. In this paper we discuss the possible benefits and uses of agents with a Theory of Mind in virtual training, and discuss the development and implementation of such agents.

Keywords

Virtual training, Theory of Mind, BDI agents.

1. INTRODUCTION

Virtual training systems are often used to train people who are in command during incidents or crisis situations. In an increasing number of training systems, intelligent agents are used to generate the behavior of virtual characters in the training scenarios, saving time and costs. We consider training systems in which one trainee has to interact with one or more of these agents to accomplish a certain task or mission in the scenario. Though simple agent behavior can be accomplished quite well, the development of agents displaying more complex behavior is still to be improved.

Typical human errors in incidents or crises involve situations in which people make false assumptions about other people's knowledge or intentions. For example, a leading fire-fighter that receives information about a second fire in a building may immediately make an assessment of the new situation, come up with an alternative plan, and redirect part of the team. In the rush of the moment, the leading fire-fighter could unjustly assume that other people nearby, e.g. ambulance personnel, already know about the second fire, unnecessarily exposing them to high risks. Besides similar stories of professionals [10, 28], attributing incorrect knowledge and intentions to others in general is a well described phenomenon in cognitive sciences (e.g. [20, 15]). To provide a system in which correctly estimating others' knowledge and intentions can be trained, we propose to use agents that act on the basis of actual mental concepts, and are able to attribute mental states to other agents.

The first requirement, agents acting on the basis of mental concepts, can be realized by using a BDI-based agent programming language. The BDI-model is based on folk psychology, which encapsulates the way humans *think* that they reason [2]. Namely, humans use concepts such as beliefs, goals and intentions to to understand and explain their own and others' behavior [14]. Accordingly, a BDI agent executes actions based on its beliefs, plans and goals. Several applications have demonstrated that BDI agents are appropriate for modeling virtual characters in games or training systems (e.g. [21, 25], respectively). In virtual training with BDI agents, trainees can practice in correctly interpreting the agents' behavior and compare their estimated mental concepts to the agents' actual ones.

To satisfy the second requirement, agents attributing mental states to other agents, the agents have to be extended with a Theory of Mind (ToM). Entities with a ToM attribute mental states such as beliefs, intentions and desires to others in order to better understand, explain, predict or even manipulate others' behavior. Beliefs about the other's mental state can be different from beliefs about one's own. Agents that act on basis of a ToM give trainees the opportunity to experience how their behavior is interpreted by others, and to train on coping with people that make false assumptions about other's beliefs and goals. As far as the authors know there are currently no agent programming languages providing explicit constructs for the implementation of agents with a ToM.

In this paper, we will explore the possible benefits and uses of agents with a ToM in virtual training, and discuss the development and implementation of such agents. In section 2, we start by providing some background information on ToMs. In section 3, we sketch possible uses of agents with a ToM in virtual training, and therefrom determine implications for the implementation of agents with a ToM. Based on the implied requirements, we introduce an approach for the implementation of agents with a ToM in section 4. We end the paper with a discussion on related research in section 5, and a conclusion and suggestions for future work in section 6.

*This research has been supported by the GATE project, funded by the Netherlands Organization for Scientific Research (NWO) and the Netherlands ICT Research and Innovation Authority (ICT Regie).

2. WHAT IS A THEORY OF MIND?

The concept of a Theory of Mind is studied in different research fields. Philosophers and psychologists debate on how a ToM works in humans, developmental psychologists study how children acquire a ToM to obtain more insight into its working, and others study behavior of autistic people, who show deficits in their ToM use. The false-belief task is often used to determine whether someone has a fully developed ToM [27]. To pass the test, the participant has to attribute a false belief to someone else. Furthermore, neuroscientists study neural correlates of ToM use, and biologists discuss about whether primates have a ToM.

All this research aims to improve understanding in the actual working of a ToM in humans or animals. In contrast, in order to develop agents with a ToM it has to be decided how a ToM is designed. Agents with a ToM do not always have to be as similar to humans as possible; the design guidelines for endowing agents with a ToM depend on the purpose for which the ToM will be used. Our aim is to endow agents with a ToM to enrich virtual training. When applied in virtual training agents should behave human-like, but only to a certain extent. Interaction with the agents should prepare trainees for interaction with real humans, but behavior deviating from average human behavior can be used to create interesting learning situations. In conclusion, we inspire our approach of ToM on theories about human ToM, but do not strictly follow them.

The two most prominent accounts of human ToM are the theory-theory (e.g. [6]) and simulation theory (e.g. [11]). According to theory-theorists, a ToM is developed automatically and innately, and instantiated through social interactions. The mental states attributed to others are unobservable, but knowable by intuition or insight. ToMs are not dependent on knowledge of one's own mind. In contrast, simulation theorists argue that each person simulates being in another's shoes, extrapolating from each one's own mental experience. According to them, a ToM is an ability that allows one to mimic the mental state of another person. As will become clear throughout this paper, we adopt aspects from both theory-theory and simulation theory in our approach of ToM-based agents.

3. USES OF TOM-BASED AGENTS

In this section we discuss different uses of agents with a ToM in virtual training systems: providing feedback on trainee behavior, simulating errors due to an incorrect ToM, and supporting the trainee. It should be noted that the last two uses are special cases of the first. Namely, independent of the exact agent model, a trainee should always get feedback on his behavior by interacting with an agent with a ToM. In the last subsection, the implications of the desired uses of agents with a ToM to their implementation are examined.

3.1 Providing feedback on behavior

Agents with a ToM in virtual training should be able to make assumptions about a trainee's goals, beliefs, and future actions, and based on that determine their own actions. As a result, the trainee gets feedback on its own actions in the form of behavior of other agents. For instance, in the introduction we gave an example of a leading fire-fighter trainee that while handling an incident was challenged with the information about a second fire. Based on the trainee's behavior, which was redirecting the firefighters, the firefighter agents in his team might be able to derive the trainee's goals, e.g. splitting the team and fighting both fires simultaneously. Besides, because the trainee did not give any commands about warning bystanders, the agents might assume that the trainee already took care of that. Hence, the agents would not warn people nearby, but instead they could e.g. proactively start to divide the team in two teams. From the agents' behavior, the trainee make the inference that the others correctly derived the fire attack plan from his behavior, but that they misinterpreted his behavior concerning the bystanders.

Besides feedback through agent behavior, a trainee could also receive feedback on its behavior by ToM-based agents giving explanations about their behavior. In earlier work we have introduced a methodology for developing self-explaining agents [12], which also used BDI agents. According to this approach, self-explaining agents create a log in which, for each action, they store the goals and beliefs that brought about that action. After a training session is over, the agents can explain actions by revealing the goals and beliefs that were responsible for them. To make ToM-based agents self-explaining, not only their own beliefs and goals underlying an action should be stored, but also the attributed beliefs and goals which influenced the choice for that action. The explanations derived from such logs would thus reveal how a ToM-based agent interpreted the trainee's behavior.

To summarize, agent behavior based on a ToM of the trainee and explanations about such behavior give the trainee insight into how his behavior is interpreted by others. Such insight aims to make the trainee aware of the effects of his actions on other agents, and let him prevent possible misinterpretations of his behavior. Moreover, by receiving explanations about agents' behavior, the trainee can check whether his own interpretations of their behavior were correct.

3.2 Simulating errors due to incorrect ToMs

As mentioned in the introduction, incorrect (use of) ToMs is a well described phenomenon in the cognitive sciences. A lot of research demonstrates the human tendency to impute one's own knowledge to others (Nickerson gives an extensive overview [20]). Although this serves them well in general, they often do so uncritically and assume erroneously that other people have the same knowledge as they have. Sometimes the mechanism thus yields incorrect ToMs.

Keysar et al's research suggests human limits on the effective deployment of theory of mind [15]. They describe one of their experiment as follows: "A person who played the role of director in a communication game instructed a participant to move certain objects around in a grid. Before receiving instructions, participants hid an object in a bag, such that they but not the director would know its identity. Occasionally, the descriptions that the director used to refer to a mutually-visible object more closely matched the identity of the object hidden in the bag. Although they clearly knew that the director did not know the identity of the hidden object, they often took it as the referent of the director's description, sometimes even attempting to comply with the instruction by actually moving the bag itself. In a second experiment this occurred even when the participants believed that the director had a false belief about the identity of the hidden object, i.e. that she thought that a different object was in the bag." The results show a stark dissociation be-

tween the ability to reflectively distinguish one's own beliefs from others', and the routine deployment of this ability in interpreting the actions of others. Some adult subjects could not correctly reason in a practical situation about another person's lack of knowledge.

Hedden and Zhang [13] conducted experiments in which people were challenged to use a theory of mind in a sequence of dyadic games. Players generally began with first order reasoning (my co-player knows p), and only some of the players started to use second order reasoning (my co-player does not know that I know that p). Mol et al [19] also found that only few people deploy second order ToM in a task where reasoning about others was advantageous. The skill to deploy second order ToM however can be essential in domains such as crisis management and firefighting.

In a training situation, agents could unjustly assume that the trainee has certain knowledge about the situation. For instance, a fire-fighter agent, extinguishing a fire in a building with its colleagues, observes that there is a victim. The agent communicates its observation to its team members, but not to the commander, which is played by the trainee. The agents might impute their own knowledge to the trainee, i.e. they think the trainee also knows about the victim. Consequently, the agents think that it is not necessary to communicate the information to the trainee, and independently start to take actions to take care of the victim. Though the trainee does not know about the victim, he could derive from the agents' actions that something has happened. For example, he notices that there is not going any water through the fire hoses. In such a case, the trainee could contact the fire-fighters to ask why they are not yet extinguishing the fire. In reaction, the fire-fighters can explain their behavior by their ToM of the trainee, e.g. I thought you knew there was a victim, therefore I started taking care of the victim.

Another example in which the trainee can practice with agents with a limited ToM is that agents have wrong expectations about each other's tasks. If an agent expects that something is not a goal of the trainee because it thinks the goal does not belong to the trainee's tasks, it might adopt that goal itself. Then, unnecessarily, two players would try to achieve the same goal. The other way around can even be worse, if an agent unjustly thinks that something is the trainee's responsibility, the task might not be performed at all. The trainee is challenged to detect these incorrect expectations on the bases of other agents' actions, and gets the opportunity to deal with such situations.

In conclusion, agents with a limited ToM can be used to create challenging training situations by making realistic errors. Namely, agents with an incorrect ToM about the trainee will not perform optimal behavior, and the trainee is challenged to detect such errors as early as possible and overcome possible problems.

3.3 Supporting the trainee

The goal of a training scenario is to engage a trainee in intended learning situations. Sometimes, this causes a tension between freedom of the trainee and control over the events in the scenario. The trainee's freedom increases if it can perform a wide range of actions to which the virtual environment reacts in a believable way, from which he might learn a lot. However, too much freedom endangers the continuation of the storyline of the training scenario in a desired way. For example, if a trainee makes bad decisions at an early stage of a session, the situation might quickly walk out of hand and the session be over soon. If the trainee would have reacted more adequately in the beginning, he would have encountered much more learning opportunities.

Agents with a ToM can be used to exert some control over the storyline by supporting the trainee if he makes errors or fails to take actions that are crucial to the continuation of the storyline. Because the agents' support actions are based on their ToMs and not just artificial interventions, ToM-based behavior is a natural way to balance user freedom and story control. For instance, an agent with beliefs about the current situation and a ToM containing information about the trainee's tasks, i.e. his goals, can simulate a reasoning process with its own beliefs and the attributed goals. The outcome of the process shows what the agent would do itself in the trainee's place. If that differs from the trainee's actual actions, the agent can redirect the trainee by supporting behavior such as correcting the him or taking over his tasks.

Most research to finding a balanced combination of user freedom and control over a storyline has been done in the interactive storytelling community, where the problem is called the *narrative paradox* [17]. A common solution to the narrative paradox is the use of a director or manager agent acting 'behind the scenes' (e.g. [18, 23]). Its task is to make sure that the scenario is carried out according to predefined constraints, while preserving realism. Generally, story guidance in these approaches happens at certain points of the scenario and leaves the player free space to interact in the remaining time.

The use of agents with a ToM to exert control over a scenario does not contradict approaches with a director agent, instead, both could complement each other. Namely, in some occasions you might want to support the trainee, e.g. if he is performing bad, but in other cases you do not, e.g. to let him experience the consequences of his actions. The agents acting in the training scenario have knowledge about the domain, and know how to support the trainee in such a way that the incident will be solved. A director agent has didactical knowledge and knows on which occasions you do or do not want to support the trainee. The particular design of ToM-based agents facilitates directing them; the director agent can command them to either use or not use their ToMs. The director agent could also control the simulation of mistakes due to incorrect ToMs as described in the previous section by prescribing when agents have to purposely make mistakes and when not.

The advantage a combined ToM-based agents and director approach is that it facilitates direction by a director agent. Moreover, not only the observable behavior of the agents remains realistic, but also the reasoning steps that generated it. The agents' reasoning processes are not interrupted, but changed in a plausible manner. Consequently, agents' self-explanation would still deliver useful explanations, even if their behavior was redirected. We have discussed the ideas in this paragraph more extensively in [24].

3.4 Implications for implementing a ToM

By interacting with agents with a ToM and studying their explanations, a trainee obtains indirect and direct feedback on his behavior, respectively. In our approach to self-explaining agents, we argue that the explanation of behavior should be connected to its generation [12]. Thus, intentional self-explaining agents should not only behave *as if they were*

intentional, but act on the basis of actual goals and beliefs. As a result, the self-explaining agents can be best implemented in a BDI-based agent programming language. Similarly, self-explaining agents with a ToM should not only behave *as if they had* a ToM, but actual attributed mental concepts should play a role in the generation of behavior. This requirement implies that an implemented agent with a ToM should be able to explicitly represent the beliefs, goals and other mental concepts that it attributes to others.

In all of the described uses of agents with a ToM, the agents reason with others' mental concepts, i.e. they predict actions on the basis of attributed goals and beliefs. For example, an agent that believes that agent B has belief X and goal Y, and that agents such as B with X and Y intend action *a*, should be able to derive that agent B probably intends action *a*. Based on its expectation of action *a*, the agent can select its own actions and thereby show the trainee how it interpreted his behavior (section 2.1), challenge the trainee to detect the flaws in its reasoning (section 2.2), or give the trainee support (section 2.3). It should be noted that a 'normal' reasoning process results in an actual action that is executed in the environment, but ToM-based reasoning should only result into an expected action which is not executed. In other words, the simulation of someone's reasoning process should not have consequences for the environment. Thus, the implementation must allow agents to reason with attributed mental concepts, without affecting the environment.

To support the trainee, an agent uses its ToM to derive the cause of a trainee's wrong or failing behavior, e.g. an incorrect belief or the lack of a goal. Therefore, the agent should not just reason with attributed beliefs and goals, but also with ones that are possibly attributable. For instance, the agent should be able to predict what agent B would do if it would believe X and have goal Y, without actually believing that B has that belief and goal. By comparing predicted actions to the trainee's actual ones, the agent can diagnose the probable causes for his behavior and react on that. For the implementation this means that agents with a ToM must be able to reason with different combinations of not (yet) attributed goals and beliefs.

Finally, agents should use their ToM for the generation of behavior in order to add value to a training system. Only then, trainees can be given feedback, challenged or supported. Thus, the agent program should contain actions that are involved with updating, querying or reasoning with the agent's ToM. To give support for instance, the agent program should include a rule that if the trainee is not acting for more than X seconds, the agent determines with its ToM what the trainee should do and how it can bring about the desired result.

4. IMPLEMENTATION OF A TOM

In this section we show how the requirements on agents with a ToM discussed in section 2.4 can be met by implementing them in a BDI-based agent programming language which allows for modularity. There are several BDI-based agent programming languages that allow for modularity, e.g. Jack [4], Jadex [3] and extended 2APL [9]. In these proposals, modularization is considered as a mechanism to structure an individual agent's program in separate modules. Each module contains mental elements such as beliefs, goals and plans, and might be used to generate behavior in a specific situation.

Of these approaches, the approach in extended 2APL provides agent programmers with most control over how and when modules are used [9], and therefore we use extended 2APL to illustrate our approach of implementation agents with a ToM. Currently, the extension of 2APL by modules has not been fully realized yet, but we follow the definitions of extended 2APL as given in [9] (for an overview of 'normal' 2APL see [8]). Besides, we suggest some additions to the proposed modular approach of extended 2APL, to make it appropriate for developing agents with a ToM.

In this section we first introduce a modular approach to implement agents with a ToM and show its advantages over non-modular approaches. In the second subsection, we discuss how agents can develop their ToM. For instance, when an agent observes actions of other agents it should be able to update its ToMs about those agents. Subsequently, we discuss how an agent should use its ToM in order to determine its own behavior, i.e. its assumptions about other agents' beliefs, goals or expected actions influence its own actions.

4.1 A module-based approach

A 2APL agent has a belief, a goal, and a plan base containing the agent's beliefs, goals and plans, respectively. An agent's beliefs, goals and plans are related to each other by a set of practical reasoning rules. A typical 2APL reasoning rule has the form: Head <- Guard | Body, in which the Head and Guard are tests on the agent's goal and belief base, respectively. The Body of the rule contains one or more actions, which can be abstract, decomposable actions (plans), or atomic, directly executable actions. In 2APL, beliefs in an agent's belief base are treated as Prolog facts and rules, such that the belief base of a 2APL agent becomes a Prolog program.

For the implementation of a ToM, we use extended 2APL, i.e. 2APL extended by modules, as introduced in [9], and a ToM of another agent is represented in a module. If *no* modules were used, an agent could only represent mental states attributed to other agents in its belief base. Only there, arguments can be added to the attributed mental states to distinguish them from the agent's own beliefs, and to distinguish different attributed elements from each other, e.g. beliefs and goals. For instance, without modules, the beliefs belief(B,X) and goal(B,Y) could be used to represent that an agent believes that agent B believes X and has goal Y.

The disadvantage of implementing a ToM in an agent's belief base, however, is that the interpreter of the agent program does not recognize attributed plans and goals, but only beliefs. Then, in order to reason with attributed mental concepts, epistemic reasoning rules have to be added to the agent;s belief base. A rule that makes combinations between beliefs about another agent's beliefs and goals is for example: plan(B,P) :- belief(B,X), goal(B,Y). With this rule, the agent can derive that if it believes that agent B has belief X, goal Y and reasons according to Y <- X | P, agent B's plan is P. However, this requires a translation from a supposed BDI program with practical reasoning rules to a Prolog program with epistemic reasoning rules. In contrast, when attributed mental states are represented in a module, attributed goals and plans can be represented and thus recognized as such, and the agent's own deliberation power can be used to reason with attributed mental states. Besides that the module-based solution is more practical, it saves

work for the programmer.

In order to reason with the attributed mental concepts of an agent, the attributing agent can execute the module of that particular agent. However, the outcomes of the reasoning process should have no direct consequences for the agent's environment, i.e. actions resulting from the reasoning should not be executed. The difference between executing an actual program and a module is that in the former derived actions are always executed and in the latter this is not necessarily the case. With a "dryrun" execution action, a module is executed without consequences for the environment and other agents. The state of a module after execution can be queried to estimate what the agent will do and what its active goals are. In extended 2APL for example, the command agentB.dryrun(ψ) executes agent B's till stopping condition ψ is reached. The dryrun function has been proposed in extended 2APL, but not completely specified yet.

Each module can own its own modules, which allows for modeling agents that believe that other agents also have a ToM about other agents, which in turn might have a ToM, etc. The different levels of depth in reasoning about other agents are called orders of ToM. A first order ToM defines someone's beliefs, thoughts and desires that influence one's behavior, e.g. 'she does not know that her book is on the table'. In a second order ToM it is also recognized that to predict others' behavior, the desires and beliefs that they have of one's self and the predictions of oneself by others must be taken into account, e.g. 'she does not know that I know her book is on the table'. To have a third order ToM is to acknowledge that others have a second order ToM, etc. Though it is possible to implement higher order ToMs following our approach, they quickly become complex and unpractical. However, as already mentioned, there is evidence that humans only use first-order ToM most of the time [19], and it is thus not necessary to have agents with a high order ToM for creating realistic human-like behavior.

4.2 Using a ToM

The beliefs and goals in a ToM can be present from the beginning of a scenario or obtained during interaction. In general, agents already have some knowledge about other agents before they interact. The context in which interaction takes place gives information, e.g. someone at the department hall of an airport probably has the goal to go to some place. In virtual training for incident and crisis management, the roles of the participants in the scenario give a lot of information about their probable mental states. For instance, a firefighter most probably has the goal to extinguish fires. The tasks connected to these roles in the intended domain are highly procedural in nature; for each possible situation is described how one should act. Such preknown information in a ToM is expected to be stable over the course of interaction, e.g. during one training session, and thus does not need to be updated on line.

Agents attribute new mental states to other agents during interaction based on their observations of the environment. In a shared environment one can expect that the other has similar beliefs about the environment as oneself. For instance, the ringing of a telephone is most probably heard by everybody in the same room. In extended 2APL, a module of agent B is updated with a belief X by agentB.updateBB(X), and the module of agent C is updated with a goal Y by agentC.adoptgoal(Y). Information obtained during interaction can remain the same for the rest of the session, e.g. the belief 'there has been a fire alarm'. And other attributed beliefs such as 'I am at location X' will most likely change within the course of the session. A trade-off has to be made between conservatively developing ToMs, i.e. only adding new beliefs and goals in case one can be quite sure that they are correct, or not. The former implies for the greater part correct ToMs with little information and the latter more extensive ToMs with the risk of being incorrect. Dependent on the context, the extend to which ToMs may be incorrect can be chosen.

Given that a ToM module contains some reasoning rules, goals, beliefs or plans, an agent can use a ToM by letting its decisions also depend on checks on its ToM. In other words, the results of a check on one or more of its modules are input for the agent's own deliberation process. For instance, agent B only adopts goal G if it believes that agent A has a specific belief or goal, or is going to perform a particular action. In the remainder of this section we explain the general method by which an agent can use its ToM.

As discussed in section 4.1, a typical 2APL reasoning rule has the form: Head `<-` Guard | Body. Normally, to implement that a goal G has a subgoal G', which adoption depends on conditions C, the following code is used [12].

```
G and not G' <- C | adopt(G')
```

The rule ensures that if an agent has goal G, it can only adopt G' if G' has not already been adopted and conditions C are derivable from the agent's belief base. The above rule can be used when condition C only contains possible beliefs about e.g. the agent's environment. However, if the adoption of G' also depends on what the agent believes that another agent believes, an extra step is needed. Namely, in order to obtain up-to-date beliefs about the other agent, an agent must check its ToM of that agent, possibly involving the updating or execution of the ToM module. Checking the ToM before determining whether G' should be adopted can be accomplished by the following two rules.

```
G and not G' <- not checkedToM(A,G') |
   checkToM(A,G')

G and not G' <- checkedToM(A,G') and C |
   adopt(G')
```

These two rules ensure that if an agents has goal G, before it possibly can adopt subgoal G', a check on the agent's ToM is performed. Initially, the agent does not have a belief *checkedToM(A,G')*, meaning that the ToM of agent A concerning goal G' has been checked, and only the first rule can apply. The body of the first rule contains a plan *checkToM(A,G')* for checking the agent's ToM about agent A, concerning goal G'. The execution of this plan adds two beliefs to the agent's belief base: the result of the ToM check, and the belief *checkedToM(A,G')*. Thus, after the execution of the plan *checkToM(A,G')* the first rule no longer applies, and the second rule may apply. Application of the second rule and thus the adoption of goal G' depends on whether the conditions specified in C are derivable from the agent's belief base, in this case involving beliefs about agent A's mental state.

To actually check a ToM, the agent requires a plan for each goal of which the adoption of that goal depends on

the agent's beliefs about another agent, i.e. a plan checkToM(A,G'). Such plans must per goal and agent specify in which way that agent's ToM has to be checked. In general, a plan for checking a ToM involves the following steps: updating, executing and querying the ToM module, and finishing the ToM check, of which the first two are possible steps. An example of a plan for checking a ToM in extended 2APL is as follows.

```
checkToM(A,G') <- true |
   {
   agentA.updateBB(b);
   agentA.dryrun(psi);
   if agentA.P(a) then
      UpdateBelief(agentA(a));
   UpdateBelief(checkedToM(A,G'))
   }
```

The first step of the plan checkToM(A,G') is to add belief b to the ToM module of agent A. In the next step, the module of agent A is executed till stopping condition *psi* is reached, which is e.g. that the first external action has been determined. Subsequently, by agentA.P(a) the plan base in the ToM module of agent A is queried about whether it contains a plan a, and if this is the case, the belief *agentA(a)* is added to the agent's belief base. Finally, the belief *checkedToM(A,G')* is added to the agent's belief base. In the agent's main program, one of the conditions that must be true to adopt goal G' is agentA(a).

All plans for checking a ToM involve a step in which the ToM module is queried, and all end by the addition of a belief denoting that the ToM of a particular agent for a particular goal has been checked. However, there are different ways to check a ToM, delivering different kinds of information. In the next section we discuss five ways in which a ToM can be checked.

4.3 Different ways to check a ToM

The *first* way in which an agent can check its ToM is to query the attributee agent's believed belief base, possibly after updating and executing it. For example, in extended 2APL the action agentD.B(X) queries whether belief X can be derived from agent D's believed belief base. The result should be added to the agent's own belief base, and dependent on the result the agent can adopts a goal or not.

The *second* possible check is to query the attributee's believed goal base, possible after updating it. In extended 2APL, agentD.G(Y) queries whether it is believed that agent D has goal Y. Here again, the result of the query should be added to the agent's belief base.

Third, a ToM can be used to predict what another agent is going to do. To answer that question, the ToM module of that agent has to be executed, after possible updates, till the first action that the agents is expected to perform is determined. As mentioned before, the execution is a dryrun execution so that the resulting action will not actually be performed. Subsequently, the plan base in the module can be queried, and beliefs about plans and upcoming actions can be added to the agent's own belief base. The code in the previous section with a plan for checking a ToM is an example of this third possible way of checking a ToM.

Note that is only possible to execute a module till the first external action is determined, because execution of the action would change the environment. Feedback from the environment about its new state is necessary to determine the next action, however, simulating someone's reasoning process should not have an effect on the environment. Currently, its is not possible to query the plan base of a module in extended 2APL, this capability should be added to the language.

The *fourth* use of a ToM involves the interpretation of someone's actions, that is, observed agent actions are explained by a ToM. For example, when an agent suddenly stops extinguishing a fire, it might have seen a victim or something that causes explosion danger. In order to use a ToM according to this fourth way an agent needs: one or more observed actions, a ToM with already some information about the other's mental state, and a set of possible explanations.

Depending on the situation, agents have more or less knowledge on specific elements of another's mental state. For example, in a game like chess, the opponent's main goal and its beliefs are completely clear, that is, winning the game and the positions of the pieces on the board. However, the plans with which it wants to achieve this goal, and the reasoning rules it uses are harder to estimate. In contrast, for the guesser in a game like mastermind, the other's beliefs are exactly what it does not know but needs to find out, i.e. the colors and positions of the four pieces. With information about part of someone's reasoning elements and observations of his actions, missing information can be derived and added to a ToM. Without any information it is impossible to interpret someone's actions and extend the ToM.

Besides a ToM with a number of attributed mental concepts, an agent needs to have knowledge about possible explanations for someone's actions. In each context, particular events are likely to take place and of influence on agents' actions; these events are the possible explanations for someone's behavior. Following from the previous paragraph, in some domains possible explanations consist of beliefs (e.g. a mastermind game), and in others they consist of plans (e.g. a chess game). Per application, a set of possible explanations has to be defined, i.e. a set of beliefs or goals or plans. In the domain of crisis management, agents usually have information about others' possible goals and plans because these are connected to their roles. The agents' beliefs which determine their goals and plans, in contrast, are expected to be constantly changing during an incident scenario: an incidents develops, the agents obtain more and more information about the incident, and the agents are actively trying to change the situation.

Once an agent has the disposal of one or more observed actions, a ToM (a module) and a set of possible explanations (set of beliefs or goals), it can interpret an attributee's actions. To do so, the agent has to 'try out' different possible explanations by, for each possible explanation, temporarily adding or removing a belief or goal to the ToM module and perform a dryrun execution with the new module. The actions that are outcomes of the dryrun executions are compared to the attributee's actual actions. If there is a match, the particular alternation to the ToM probably explains the attributee's actions, and the temporary change can be incorporated in the ToM permanently. The best explanation of someone's actions can be added to the agent's own belief base, and used to determine its own behavior. For this capability, extended 2APL has to be extended by the possibility to temporarily update a module, execute and query it, and

then make the temporal update undone.

The *fifth* use, like the fourth, involves the addition or removal of different combinations of beliefs or goals to a ToM. The resulting actions after executing the modules are not compared to actual actions of the agent, but to desired actions. This is useful when the agent wants to make an agent act in a certain way, i.e. socially manipulate the agent. By trying different combinations of mental states, one can find out what an agent must believe or which goal it must have to display the desired behavior. This information can help the agent to bring about this behavior. For the last two uses of a ToM, an agent's ToM must be updated temporarily with certain beliefs or goals, but after execution and one or a few queries, the updates must be made undone.

5. RELATED WORK

Bosse et al have introduced a formal BDI-based agent model for Theory of Mind [1]. As in our approach, the attributing agent is BDI-based, and its ToM about other agents is also in terms of beliefs, desires and intentions. Bosse et al especially focus on how the agent can use its ToM in its own BDI-based reasoning processes with the purpose of social manipulation. In this paper, we have focused more on the implentation of a ToM and only shown the possibilities of using it, which also involves social manipulation. A difference is that in their approach agents only reason *about* attributed mental concepts, and in our approach agents reason *with* attributed mental concepts as it were their own. Thus, as argued in section 4.1, in our approach the deliberation power for normal reasoning is also used for reasoning with attributed mental concepts.

PsychSim is a simulation tool for modeling interactions and influence involving agents with a ToM [22]. PsychSim agents have a decision-theoretic world model, including beliefs about their environment and recursive models of other agents. Instead of using qualitative BDI concepts, they use quantitative models of uncertainty and preferences. According to them, a quantitative model is required to resolve the ambiguity among equally possible, but unequally plausible or preferred options. We take a BDI approach because it gives insight into the agents reasoning processes, which is desired in training applications. In our model, information about preferences is included in the agents' BDI models, and between equally preferred options is chosen randomly.

In section 2.4 we have discussed several ways to develop a ToM, among which the ascription of mental states to an agent based on its actions. This phenomenon is also called intention or plan recognition and has been investigated by several researchers. Models of plan inference all start with a set of goals that an agent might be expected to pursue in the domain and an observed action by the agent. Most plan recognition systems infer an agent's goal from the observed action by constructing a sequence of goals and actions that connect the observed action to one of the possible domain goals [5]. The difference between these systems and our approach is that they start with the observed action and infer possible sub-goals and goals, and we start reasoning with possible goals and determine which actions could result from them. The disadvantage of our approach is that it requires more steps because all possible goals have to be considered, whereas most other systems can exclude possible goals. However, the advantage of our approach over others is that it makes use of an agent's existing reasoning strategies, and no extra plan inference system has to be built.

One of the uses of agents with a ToM is to provide sophisticated explanations about behavior. There are several accounts of self-explaining agents in virtual training systems, e.g. Debrief [16], XAI version I [26] and II [7], and our own [12]. In these systems, trainees get the opportunity to query agents about their behavior in the played training session. The explanations aim to improve trainees' insight in the training situation. The kind of questions that the agents are able to answer vary from systems to system. Debrief reveals what must have been the underlying beliefs of an agent, XAI I informs about the state of an agent at a particular time, e.g. its position or ammunition, XAI II answers questions about agents underlying motivations that are made available by the simulation, and our methodology delivers agents that give explanations in terms of their underlying beliefs and goals. None of the current accounts of self-explaining agents provides explanations involving attributed mental states.

6. CONCLUSION

In the present paper we have examined the possible benefits and uses of agents with a ToM in virtual training. We have given different examples of how agents with a ToM can be useful in such applications, and illustrated that training with ToM-based agents is valuable in particular to train on correctly interpreting others' behavior and becoming aware of how one's behavior is interpreted by others. We have introduced a modular approach for the implementation of ToM-based agents. The proposed approach to implement a ToM allows an agent to reason *with* attributed mental concepts instead of *about* them which implies that an agent can only attribute mental concepts to other agents that it can have itself. As a result, ToM-based agents can fully make use of their deliberation power when reasoning about other agents.

In section 2 we have introduced the two most prominent current accounts on how ToMs work in humans: theory-theory and simulation theory. In our approach, after checking their ToMs, agents do obtain beliefs about mental states of other agents in their own belief base, which corresponds to the vision adhered to by theory-theorists. However, reasoning about nested beliefs and goals is done by simulation, which has similarities with the simulation theoretical view. In conclusion, our approach involves aspects of both theory-theory and simulation theory. In our opinion, an agent needs to have some preprogrammed constructs in order to be able to develop a ToM. However, it may use its own reasoning power to reason with mental concepts attributed to another agent, i.e. simulate the reasoning process of another agent.

We have illustrated the implementation approach with examples in extended 2APL, a version of 2APL extended with modularity as proposed in [9]. The examination of uses of agents in virtual training systems delivered a number of new requirements to extended 2APL which were not foreseen in the original proposal. Namely, to implement agents with a ToM a dryrun function, the possibility to query a module's plan base, and the possibility to make temporal updates to a module are needed. We are currently working on the development of 2APL extended with modules, including these extra functionalities. After extended 2APL has been developed, we will implement actual agents with a ToM and apply them in a virtual training system. Then, we will be

able to conduct user experiments, which should show the use of agents with a ToM in virtual training systems.

7. REFERENCES

[1] T. Bosse, Z. Memon, and J. Treur. A two-level bdi-agent model for theory of mind and its use in social manipulation. In *Proceedings of the AISB 2007 Workshop on Mindful Environments*, pages 335–342, 2007.

[2] M. Bratman. *Intention, Plans and Practical Reason.* Harvard University Press, Cambridge, Massachusets, 1987.

[3] L. Braubach, A. Pokahr, and W. Lamersdorf. Extending the capability concept for flexible bdi agent modularization. In *Proc. of ProMAS 2005*, pages 139–155, 2005.

[4] P. Busetta, N. Howden, R. Ronnquist, and A. Hodgson. Structuring bdi agents in functional clusters. In N. Jennings and Y. Lesperance, editors, *Intelligent Agents VI: Theories, Architectures and Languages*, pages 277–289, 2000.

[5] S. Carberry. Techniques for plan recognition. *User Modeling and User-Adapted Interaction*, 11(1-2):31–48, 2001.

[6] P. Carruthers. *Theories of theories of mind*, chapter Simulation and self-knowledge: a defence of the theory-theory. Cambridge University Press, Cambridge, 1996.

[7] M. Core, T. Traum, H. Lane, W. Swartout, J. Gratch, and M. van Lent. Teaching negotiation skills through practice and reflection with virtual humans. *Simulation*, 82(11):685–701, 2006.

[8] M. Dastani. 2APL: a practical agent programming language. *Autonomous Agents and Multi-agent Systems*, 16(3):214–248, 2008.

[9] M. Dastani, C. Mol, and B. Steunebrink. Modularity in agent programming languages: An illustration in extended 2apl, 2008.

[10] R. Flin and K. Arbuthnot, editors. *Incident command: Tales from the hot seat.* Ashgate Publising, 2002.

[11] R. Gordon. *Theories of theories of mind*, chapter 'Radical' simulationism. Cambridge University Press, Cambridge, 1996.

[12] M. Harbers, K. Van den Bosch, and J. Meyer. A methodology for developing self-explaining agents for virtual training. to appear, 2009.

[13] T. Hedden and J. Zhang. What do you think i think you think? theory of mind and strategic reasoning in matrix games. *Cognition*, 85:1–36, 2002.

[14] F. Keil. Explanation and understanding. *Annual Reviews Psychology*, 57:227–254, 2006.

[15] B. Keysar, S. Lin, and D. Barr. Limits on theory of mind use in adults. *Cognition*, 89:25–41, 2003.

[16] W. Lewis Johnson. Agents that learn to explain themselves. In *Proc. of the 12th Nat. Conf. on Artificial Intelligence*, pages 1257–1263, 1994.

[17] M. Lockelt, M. Pecourt, and N. Pfleger. Balancing narrative control and autonomy for virtual characters in a game scenario. *INTETAIN*, pages 251–255, 2005.

[18] M. Mateas and A. Stern. Integrating plot, character and natural language processing in the interactive drama façade. In *Technologies for Interactive Digital Storytelling and Entertainment*, 2003.

[19] L. Mol, L. Verbrugge, and P. Hendriks. Learning to reason about other people's minds. In *Proceedings of the Joint Symposium on Virtual Social Agents*, Hatfield, United Kingdom, 2005. University of Hertfordshire.

[20] S. Nickerson. How we know -and sometimes misjudge- what others know: Imputing one's own knowledge to others. *Psychological Bulletin*, 125(6):737–759, 1999.

[21] E. Norling. Capturing the quake player: using a bdi agent to model human behaviour. In J. Rosenschein, T. Sandholm, M. Wooldridge, and M. Yokoo, editors, *Proceedings of AAMAS 2003*, pages 1080–1081, 2003.

[22] D. Pynadath and S. Marsella. Psychsim: Modeling theory of mind with decision-theoretic agents. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1181–1186, 2005.

[23] Riedl and Stern. Believable agents and intelligent scenario direction for social and cultural leadership training. In *Proc. of the 15th Conference on Behavior Representation in Modeling and Simulation*, 2006.

[24] K. Van den Bosch, M. Harbers, W. Van Doesburg, and A. Heuvelink. Intelligent agents for training on-board fire fighting. under review.

[25] W. A. Van Doesburg, A. Heuvelink, and E. L. Van den Broek. Tacop: A cognitive agent for a naval training simulation environment. In S. T. M. Pechoucek, D. Steiner, editor, *Proceedings of the Industry Track of AAMAS 2005*, pages 34–41, 2005.

[26] M. Van Lent, W. Fisher, and M. Mancuso. An explainable artificial intelligence system for small-unit tactical behavior. In *Proc. of IAAA 2004*, Menlo Park, CA, 2004. AAAI Press.

[27] H. Wimmer and J. Perner. Beliefs about beliefs: Representation and constraining function of wrong beliefs in young children's understanding of deception. *Cognition*, 13:103–128, 1983.

[28] C. Zsambok and G. Klein, editors. *Naturalistic Decision Making.* Lawrence Erlbaum Associates, 1997.