# Evaluation of Virtual Agents utilizing Theory of Mind in a Real Time Action Game

Mark Hoogendoorn
VU University Amsterdam
Department of Artificial Intelligence
De Boelelaan 1081a
1081 HV Amsterdam, the Netherlands
mhoogen@cs.vu.nl

Jeremy Soumokil
VU University Amsterdam
Department of Artificial Intelligence
De Boelelaan 1081a
1081 HV Amsterdam, the Netherlands
wizzra@hotmail.com

## ABSTRACT
Within the domain of virtual agents, Theory of Mind reasoning is considered an important attribute to enable such agents to act in an intelligent fashion. Such ideas have been applied in various domains, ranging from serious games to virtual storytelling. Another interesting domain of application of agents attributed with Theory of Mind is the entertainment gaming industry, which poses a completely different goal upon the behavior of these agents, namely to bring the players a fun experience. To this end, this paper introduces an approach to develop Theory of Mind agents for the entertainment gaming industry. Hereby, PRS is used for the reasoning of the agent, which has been extended with Theory of Mind reasoning capabilities. The behavior of the agent has been evaluated against two other agents (a simple reactive agent, and a memory-based agent) in an experiment that has been conducted in which 15 participants had to play against all agent types, and rate each one of them based upon certain criteria.

## Categories and Subject Descriptors
I.2.11 [**Computing Methodologies**]: Distributed Artificial Intelligence - *Intelligent Agents.*

## General Terms
Design, Experimentation.

## Keywords
Virtual Agents, Theory of Mind, Entertainment games.

## 1. INTRODUCTION
Within the development of virtual agents, a variety of authors stress the importance for agents to have the ability to reason about the state of mind of other agents, also referred to as the *Theory of Mind* (see e.g. [1]). This enables these agents to act in a more believable and natural way. For example, Pynadath and Marcella [11] have developed a social simulation tool in which Theory of Mind is explicitly incorporated. Bosse, Memon and Treur [2] introduce a BDI model which is able to reason about another agent's BDI model to enable social anticipation as well as social manipulation.

Example applications in which agents attributed with Theory of Mind are deployed include serious games (see e.g. [3]) and virtual conversations (see e.g. [10]). Another interesting domain of application is entertainment gaming, which has fundamentally different goals than the serious gaming domain. Entertainment games are not necessarily developed to learn something, but instead are meant to give the player a fun experience. In entertainment games, often so called non-player characters (NPCs) are part of the game world, and contribute to the overall fun factor. In the current state-of-the-art in the entertainment gaming industry, intelligent behavior of NPC agents is typically generated through often relatively simple finite state machines [13]. Laird [6] claims that a crucial element missing in these NPCs is the capability to anticipate and adapt to players. Overall, one can say that there are potential gains to be made from the ongoing developments in the field of virtual agents, in particular those involving agents attributed with Theory of Mind to improve the behavior of such NPC agents, and hence, allow them to anticipate and adapt to the player's behavior.

Whether new developments in virtual agents contribute to the overall fun factors is not a trivial matter. Sweetser and Wyeth [15] defined eight aspects that are considered as the responsible factors which sum up to an interesting entertainment game: (1) *concentration*, games require concentration, and the player should be able to concentrate on the game; (2) *challenge*, the game needs to be challenging, and match the player's skill level; (3) *player skills*, games should support the development of player's skills; (4) *control*, players feel a sense of control over their actions in the game; (5) *clear goals*, the goals should be clear to the player at all times; (6) *feedback,* players must receive appropriate feedback at the right times; (7) *immersion,* players should be very much involved in the game without having to put effort in it, and (8) *social interaction*, the game should provide opportunities for social interaction. The NPC agents within the game do not contribute to all these aspects. Livingstone [7] shows that an NPC agent's intelligence affects *immersion* (if the characters are not

**Cite as**: Evaluation of Virtual Agents Utilizing Theory of Mind in a Real Time Action Game, Mark Hoogendoorn and Jeremy Soumokil, *Proc. of 9th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2010)*, van der Hoek, Kaminka, Lespérance, Luck and Sen (eds.), May, 10–14, 2010, Toronto, Canada, pp. 59-66 

realistic, the player will not be involved in the game) and *challenge* (non-intelligent NPC agents pose less of a challenge as they are more easily deceived).

In this paper, the main research question is whether NPC agents attributed with Theory of Mind can improve the gaming experience of humans compared to more traditionally developed NPC agents. For this purpose, a PRS engine (cf. [4]) was implemented in a popular game development framework, namely Unreal Engine 3. The architecture was extended with features to allow explicit Theory of Mind reasoning using Default Logic (cf. [12]). A game was developed to serve as a case study, as well as three types of NPCs: a simple reactive agent, a memory based agent, and an agent attributed with Theory of Mind. In order to evaluate the player's experiences with these NPC agents, an experiment was conducted in which 15 participants took part, thereby playing the game with each of the NPC agents in random order, and filling in a questionnaire to evaluate each of them.

This paper is organized as follows: Section 2 presents the game scenario used, whereas Section 3 addresses the design of the various agent types. The experimental setup is described in Section 4, and Section 5 presents the results of the experiment. Finally, Section 6 is a discussion.

## 2. GAME SCENARIO

In order to develop and evaluate a Theory of Mind based NPC agent for an entertainment game, the first step taken was developing a simple game.

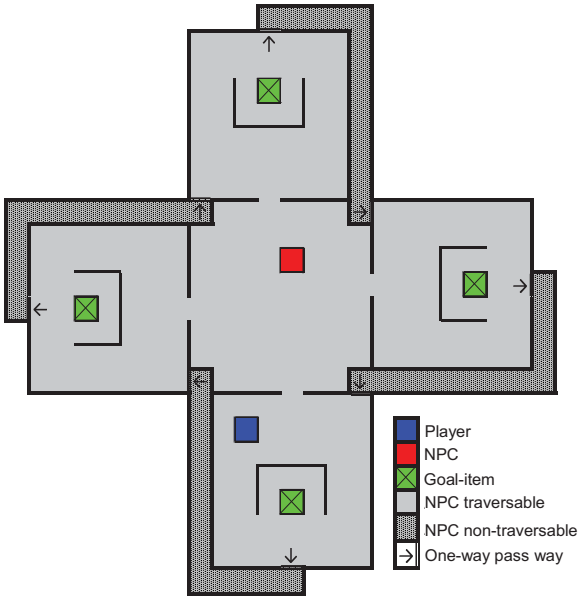


Figure 1. Game World

The game scenario consists of a game world (see Figure 1) in which the player has to collect four items. The items are depicted by the crossed colored boxes, and are placed behind a wall (indicated by the lines in the figure). The player is free to move around in the game world and can use specific one-way passways. These are only accessible through a door which only opens when the item next to the door is still present. Passways are shown as shaded areas in Figure 1. The player is opposed by one enemy, the NPC, which roams around the game world. This NPC has the ability to chase the player with its slightly higher velocity, and can kill the player by touch. The NPC is not able to use the one-way passes, which gives an advantage to the player. A screenshot of the game is shown in Figure 2.

Figure 2. Screenshot of the game

## 3. AGENT DESIGN

This Section addresses the design of the NPC agent. In order to make a comparison possible, three agent types have been created, namely a simple reactive agent, an agent with a memory, and finally a complete Theory of Mind agent. Before going into details on the specific design of these agents, the general framework in which the agents have been created is explained.

### 3.1 Agent Design Approach

For the expression of explicit Theory of Mind reasoning, a BDI architecture is a suitable choice as this is based upon the same mental notions as those used to express and reason about mental models of others. Subsequently, PRS (for Procedural Reasoning System, cf. [4]) was adopted for its roots in BDI foundations. A conceptual overview of the PRS architecture integrated in the game world is shown in Figure 3.

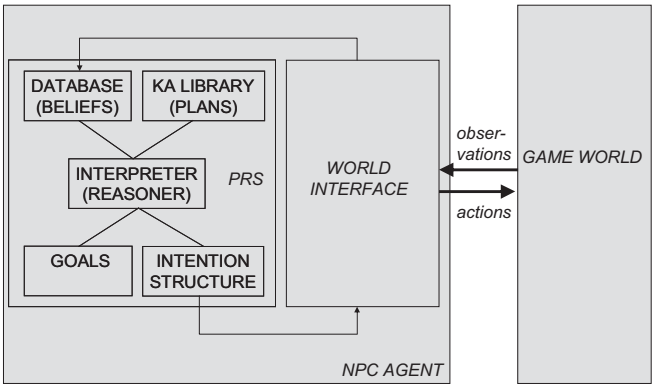


Figure 3. PRS in the game world (PRS architecture from [4])

The figure shows that the NPC agent and the game world are separate components that communicate with each other through an interface (i.e. the world interface component). The world interface component in turn, is connected to the PRS reasoning structure. The choice for this modular architecture was made to

keep the reasoning of the agent independent of the precise constructs in the game world.

The PRS part of the agent functions according to the specification in [4] and its implementation is rooted in the formal dMars specification (cf. [5]). The components within the structure can be described as follows:

- The *database* component is essentially the current set of beliefs the agent has. Beliefs are stored and referred to as 'facts' as these are true from the agent's perspective.
- The *goals* component contains the goals the agent pursues. These are expressed as conditions over some interval of time, which allows a variety of goal types including achievement goals, maintenance goals, and goals to test for certain conditions.
- The *Knowledge Area Library* contains the plans the agent has. Each of these plans consists of a body, describing the steps of the procedure, and an invocation condition, specifying under what conditions a plan is applicable.
- The *Intention Structure* encompasses all the tasks that have been chosen to be executed (also referred to as intentions). These can also include tasks that are executed at a later time point. A single intention is a plan, possibly combined with subplans that are expressed in the plan's main body. Intentions can also be dropped from the intention list, for example due to particular changes that no longer require or permit execution of such a plan.
- Lastly, the *system interpreter* is the glue between the aforementioned components. At any particular time point, a set of beliefs is present, as well as a set of goals. Given this combination, certain plans are applicable (i.e. the invocation condition is satisfied). At least one of these plans will then be chosen to become an intention.

The plans are specified following the dMARS specification (cf. [5]). Hereby, plans are specified by means of five elements:

- An *invocation condition* of the plan.
- The *context* of the plan, which specifies a situation formula that must be believed by the agent for a plan to be executable.
- The *body* of the plan includes a tree representing a flow-graph of the steps in the plan. Within the plan body, three possible constructs are used, namely (1) QUERY, which allows one to query whether the agent currently holds a certain belief; (2) EXECUTE, which specifies the execution of an action, and (3) ASSERT which declares that a new fact is to be added.
- A *maintenance condition* which is a condition that must be true to continue execution of a plan.
- A set of *internal actions* that express what should be done in case a plan succeeds or fails.

An example of a plan that can be adopted by an NPC agent is shown in Figure 4, specified in the format that is interpreted and fed into the PRS implementation that interacts with the game world. The particular plan contains directives to kill an enemy. The invocation condition signifies that a sub-goal to Kill $X must present for the plan to be activated. Note that variables (distinguished from values by the prefix '$') stated in the invocation condition and the context are automatically bound. In this case, $X would be bound to "player". This binding process only occurs when the plan is invoked, and is not updated during plan execution. Additional context expresses that the plan is only valid when the player is nearby (Near $X). When these conditions are met, the plan is executed. The first element of the plan body consists of a query that checks whether or not the player is in reach (IsInReach $X). The first element behind the QUERY line indicates the next line that is to be processed in case the query is positive, and the second element the line to jump to in case the query result is negative. An omitted element indicates plan termination. In case the player is in reach, line 2 specifies that the player is killed. Thereafter, the plan jumps to the part specifying the actions for success, which in this case involves the assertion of the fact that $X is killed (Killed $X), and the fact that the sub goal is no longer pursued. In case the player was not in reach, it is checked whether the player is still visible, and in case he is not, the plan ends unsuccessful. Otherwise, the player is chased, followed by the check whether he is in reach again, etcetera.

| | | |
|---|---|---|
| **@Engage Enemy** | | Plan name |
| SG_Kill $X | | Invocation condition |
| Near $X | | Context |
| => | | Plan Body delimiter |
| [1] QUERY IsInReach $X | [2,10] | Plan Body |
| [2] EXECUTE Kill $X | [20] | |
| [10] QUERY IsVisible $X | [11,] | |
| [11] EXECUTE Chase $X | [1] | |
| [20] ASSERT Killed $X | [90] | Internal action for success |
| [30] ASSERT SG_Search $X | [90] | Internal action for failure |
| [90] ASSERT ¬SG_Kill $X | | |

Figure 4. Example plan.

Based upon such plans, three agents have been designed. Below, the specific plans for each of the agents are described in more detail.

## 3.2 Simple Reactive Agent

The simple reflex NPC agent behavior that has been implemented is straightforward: it follows a fixed pattern along all the locations of the goal items, and in case the player comes in sight, it starts chasing the player until it gets close enough to kill the player or until the player has moved out of sight. The single plan for this simple agent is shown in Figure 5. It expresses the following: if the nearest room is Y, and the room which is planned after Y is Z, then you first of all check whether the player is visible. If so, you chase the player, and check whether he is in reach. If he is not in reach, you check again whether he is visible, etc. Eventually, the player is either in reach, and killed (resulting in a successful completion of the plan), or no longer visible. If the latter is the case, the nearest room is patrolled, and in case the player is visible in the room, the aforementioned sequence of actions is performed again. In case the player is still not visible, the NPC agent moves to the next room (Z) and the plan to kill the player has failed (in

this case indicated by the empty space before the comma in the part between the square brackets).

```
@Simple Reactive Agent

SG_KillEnemy $X
NearestRoom $Y
NextRoom $Y $Z
=>
[1] QUERY IsVisible $X              [10,20]

[10] EXECUTE Chase $X               [11]
[11] QUERY IsInReach $X             [12,1]
[12] EXECUTE Kill $X                [50]

[20] EXECUTE PatrolLocation $Y      [21]
[21] QUERY IsVisible $X             [10, 22]
[22] EXECUTE MoveToward $Z          [23]
[23] QUERY ArrivedAt $Z             [,1]

[50] ASSERT ¬SG_KillEnemy $X
```

Figure 5. Simple reflex agent plan

## 3.3 Memory-Based Agent

The second agent type is the memory-based agent. This agent has a memory expressing whether certain goal items are still present at the locations. Briefly, the agent has the following characteristics:

- The agent only patrols rooms that still have goal items.
- If the NPC agent sees the player, the player is chased. If the NPC agent loses sight of the player, it returns to the last known location of the player.

Again, PRS plans were generated that perform precisely this behavior. The first plan in Figure 6 was created to chase a visible enemy (i.e. the last element in the bullet list above). It can be seen that the plan fires based upon the context that the enemy X is visible. First, a query is performed to see whether the enemy is still visible (as the context is only verified upon plan invocation and not updated afterwards). If the player is visible, the player is chased. In the case that the player is in reach, the NPC kills the enemy, and the plan terminates successfully. If the player is no longer visible, the location at which the player was last seen is selected to go to. In case the player is also not visible there, the NPC agent moves towards the direction at which the player is most likely at. If the player is still not visible, the plan terminates. In the situation that the player has been along the locations past, the chasing is started again.

The second plan in Figure 6 expresses a PRS plan which concerns the movement along the goal items. Hereby, for the sake of the simplicity of the plan, it has been decided to add an elementary action to patrol the non-empty room, which terminates in case the player becomes visible again.

```
@Memory-Based Agent plan for chasing

SG_KillEnemy $X
IsVisible $X
=>
[1] QUERY IsVisible $X                  [10,20]

[10] EXECUTE Chase $X                   [11]
[11] QUERY IsInReach $X                 [12,1]
[12] EXECUTE Kill $X                    [30]

[20] EXECUTE MoveToLastLocation $X      [21]
[21] QUERY IsVisible $X                 [10, 22]
[22] EXECUTE MoveTowardDirection $X     [23]
[23] QUERY IsVisible $X                 [10,30]

[30] ASSERT ¬SG_KillEnemy $X

@Memory-Based Agent plan for passing the locations
SG_KillEnemy $X
¬IsVisible $X
=>
[1] EXECUTE PatrolNonEmptyRooms         [2]
[2] QUERY IsVisible $X                  [,1]
```

Figure 6. PRS plans for the memory-based NPC agent

## 3.4 Theory of Mind Agent

The last NPC agent uses a Theory of Mind approach and explicitly uses a BDI model it has of the player. To describe the development process, the required extension of the described PRS architecture is explained. Thereafter, the BDI model of the player is addressed.

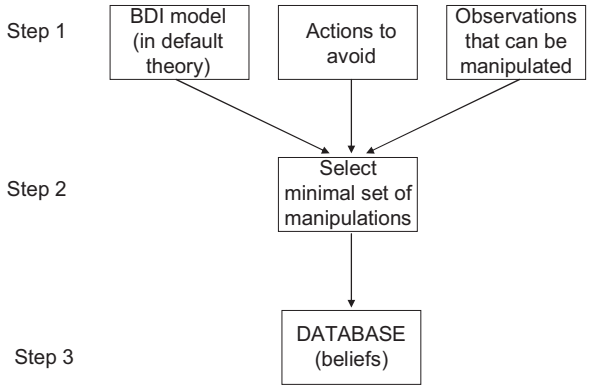


Figure 7. Theory of mind coupled to PRS

Theory of Mind combined with PRS. PRS combined with a Theory of Mind approach allows the NPC agent to reason about the BDI model of the player and manipulate the player (i.e. the player's observations) in such a manner that the player no longer performs actions that ultimately lead to winning the game. This is realized by developing specific plans that manipulate the occurrence of certain observations of the player. Reasoning about the BDI model to derive which observations the player should have to avoid him from performing useful actions is done through a Default Logic approach (cf. [12]). The approach consists of three steps, which are described below. The steps are briefly summarized in Figure 7.

The *first step* in the process is that the BDI model (as specified by means of states and state transitions in Figure 8) is translated into a default theory which is defined as pair $\langle D, W \rangle$. W is a finite set of logical formulae, called the background theory, that formalize the facts that are known certain. D is a set of default rules. A default rule has the form: $\alpha: \beta_1, .., \beta_n / \gamma$. Here, $\alpha$ is the precondition, which must be satisfied in order to derive conclusion $\gamma$. The $\beta$s, called the justifications, have to be consistent with the derived information and W. As a result $\gamma$ might be derived and more default rules can be applied. However, the end result (when no more default rules can be applied) still has to be consistent with the justifications of all applied default rules. For more details on Default Logic, such as the notion of

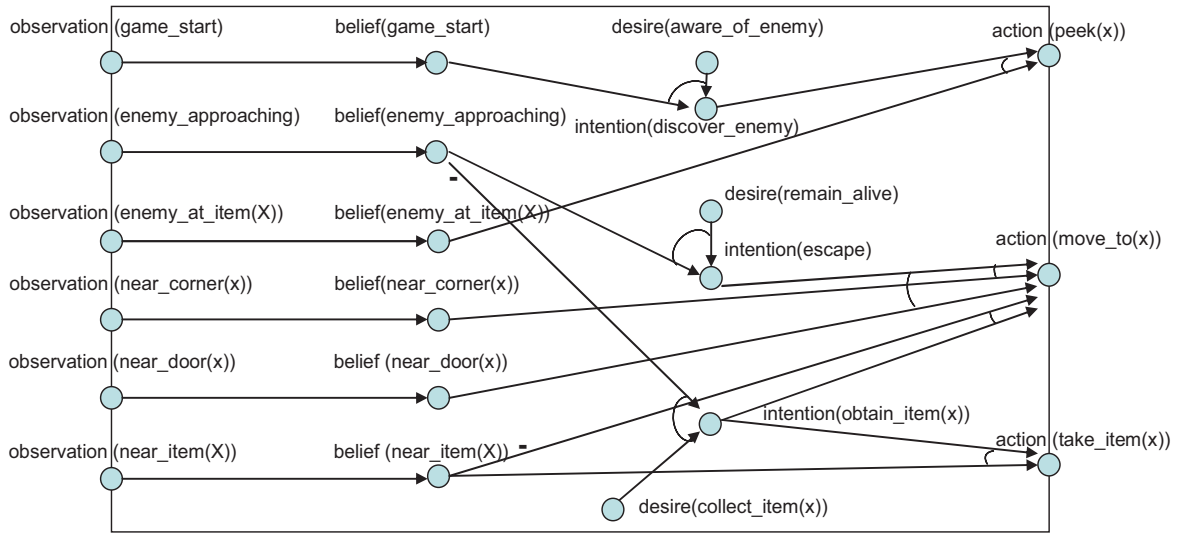


Figure 8. BDI model used for Theory of Mind

extension, see, e.g., [12, 8]. In order to translate the BDI model into the Default Theory, the connections in the model are translated into rules that constitute the background theory. The observations are translated into default rules, expressing that an observation can be assumed in case no explicit information is known, i.e. : $\gamma / \gamma$.

In the *second step*, the NPC uses the default model for reasoning. In order to do so, it is first of all assumed that the NPC agent knows what observations can be manipulated, denoted by the set $O_{man} = \{o_1, \ldots, o_n\}$. Furthermore, it also has knowledge about what actions should be avoided: $A_{avoid} = \{a1, \ldots., an\}$. Using $O_{man}$, the NPC agent creates extensions of the default theory above, by first adding single observation elements to the background theory, i.e. $W \cup o_i$ . Note that also the negation of the observation is added. Thereafter, it calculates the *stable sets* that follow from this addition (using SModels, cf. ([9]), $S = \{S1, \ldots.. Sn\}$. In case it holds that none of these sets contain an action that should be avoided (i.e. $\neg\exists S_i \in S, a_i \in A_{avoid}\ [a_i \in S_i]$), then $o_i$ is sufficient to avoid the player from succeeding. This is then passed on to step 3. Otherwise, combinations of two observations ($W \cup o_i \cup o_j$) are added, etcetera.

The *third and final step* is to add the observations that should be manipulated to the PRS database, resulting in only those plans being activated that indeed eventually result in the manipulation of these observations, and hence, the avoidance of the actions.

BDI model of the player. The BDI model used for the Theory of Mind agent is shown in Figure 8. In the figure, the circles represent the states, whereas the arrows connecting the states represent causal relations. A minus at the beginning of an arrow denotes an inverse relationship between two states (e.g. belief b does not hold, so action a is performed). Furthermore, an arc that connects two arrows expresses a conjunction. It can be seen that there are quite some observations that can be performed:

- The game has just started
- An enemy is approaching
- An enemy is present at goal item X
- I am near corner X
- I am near door X
- I am near goal item X

Based upon these observations, similar beliefs are formed. Furthermore, desires are present. In this case, three main desires are present, namely (1) to be aware of where the enemy is; (2) to remain alive, and (3) to collect all goal items. The second desire is most dominant as the player can no longer get goal items once he/she is dead. Initially, the belief that the game has just started, and the desire that the player wants to be aware of the enemy leads to the intention to discover the enemy. This intention becomes an action to peek at a location X once the player has a belief that the enemy is at location X (which is caused by an observation). A second intention within the model is to escape in case the enemy comes too close. This is caused by the desire to remain alive and the belief that an enemy is approaching. As a result of this intention, the player will either move around a corner, or move to a door that is near (both cannot occur at the same time). Finally, an intention to obtain a goal item X is shown in the model as well, caused by a desire to collect all such items and a negation of the belief that the enemy is approaching. In case the player is not near an item, the player moves towards the item in order to take it.

Utilizing the BDI model of the player. The model specified in Figure 8 has been translated into Default Logic as described above. In this case, merely two observations can be influenced, and hence, be manipulated by the NPC agent, namely enemy_approaching, and enemy_at_item(x). The result of running the three step algorithm introduced above (which is peformed each time a new BDI model of the player is loaded) is that by making sure that the NPC agent is always approaching (i.e. in sight of the player), the player never takes the goal items, and hence, never wins.As a result, this fact is added to the PRS system. Three suitable plans are present as expressed in Figure 9. The first one specifies a plan for the beginning of the game for the NPC agent. Hereby, it can be the case that the NPC agent is either already observing the player, resulting in the subgoal to kill the player being asserted, or otherwise to move to a watchpoint waiting untill the player becomes visible. The second plan expresses that the NPC agent will patrol the backdoor exit of a location where the player is at in case the goal items associated with the door is

still present. Finally, the third plan specifies that the NPC agent tries to avoid a player from escaping when he is trapped. This occurs when the player has just collected a goal item, and is unable to use the backdoor. The NPC agent will then patrol the entry location (such that the player cannot escape), and will start to chase the player in case the player is visible. More plans exist for the Theory of Mind NPC agent, but these have been left out for the sake of brevity.

```
@ Plan for going to nearest watch point to become visible in the
beginning of the game

StartGame
establish(enemy_approaching) $X
WatchPoint $Y          // Only one instance exists
=>
[1] QUERY IsVisible $X                    [20,2]

[10] EXECUTE MoveToward $Y                [11]
[12] QUERY ArrivedAt $Y                   [13,1]
[13] EXECUTE StandGuard                   [14]
[14] QUERY IsVisible $X                   [20]

[20] ASSERT SG_KillEnemy $X

@ Plan for going to the goal item that the player is going for

SG_FindPlayer $X
GoalItemLocation $Y
IsExitDoorLocationOfItem $Z $Y
IsNear $X $Y
establish(enemy_approaching) $X
=>
[1] QUERY IsVisible $X                    [20,2]
[2] EXECUTE MoveToward $Z                 [10]

[10] EXECUTE PatrolLocation $Z            [11]
[11] QUERY IsVisible $X                   [20,]

[20] ASSERT SG_KillEnemy $X               [21]
[21] ASSERT ~SG_FindPlayer $X

@ Plan to not let player escape when he's trapped

SG_FindPlayer $X
GoalItemLocation $Y
IsEntryLocationOfItem $Z $Y
IsNear $X $Y
establish(enemy_approaching) $X
 =>
[1] QUERY IsVisible $X                    [30,2]
[2] QUERY GoalItemAt $Y                   [, 10]

[10] EXECUTE MoveToward $Z                [11]
[11] EXECUTE PatrolLocation $Y            [12]
[12] QUERY IsVisible $X                   [13,]
[13] EXECUTE Chase $X                     [14]
[14] QUERY IsInReach $X                   [20,10]

[20] EXECUTE Kill $X                      [31]

[30] ASSERT SG_KillEnemy $X               [31]
[31] ASSERT ~SG_FindPlayer $X
```

Figure 9. PRS plan of Theory of Mind agent

# 4. EXPERIMENTAL SETUP

In order to evaluate the developed agents and investigate the added value of a Theory of Mind attributed agent, an experiment was conducted. The setup of this experiment is described in this section. A group of participants was asked to play the game with each of the different NPC agents types (simple reflex, memory-based, and Theory of Mind). The order in which the participants played against the various agent types was randomized to avoid an order effect. Initially, the participants were given a brief explanation of the game. Thereafter, they played one condition (either against the simple reflex, the memory-based, of the Theory of Mind agent), after which they had to fill in a questionnaire. This process was repeated until the participant rated all the different NPC agent types. Of course, the key of the experiment is what questions to ask. In total, twelve statements were defined. These were inspired by the two aspects considered important in entertainment games that are influenced by the NPC agents, namely *immersion* and *challenge* (see the introduction for more details). The participants had to rate on a seven point scale: entirely disagree, disagree, slightly disagree, neutral, slightly agree, agree, and entirely agree. The statements are expressed below.

First a number of questions were posed concerning the experience with the game itself, without specifically addressing the NPC agent which acted as an opponent (which might of course still have an influence on the rating of the statement):

1. The game was fun to play
2. The goal of the game was hard to accomplish
3. My own strategy in the game was flawless

Besides this, a number of questions were posed about the NPC agent and its behavior:

4. My opponent behaved pro-actively
5. My opponent anticipated on my actions
6. My opponent behaved human-like
7. I was able to figure out my opponent's course of action
8. I was forced to change my tactics
9. My opponent reacted slow
10. My opponent cheated
11. My opponent was easily deceived
12. My opponent executed his actions well.

# 5. RESULTS

In total, 15 participants took part in the experiment, 14 males, and 1 female. All were experienced gamers, with an age between 20 and 32. The average ratings of the participants are presented in Table 1 (where 1 equals the least agreement with the statement, and 7 the most, with the exception of "#lives lost", in which the number represents the number of lives). The standard deviation is shown between brackets. It can be seen that the Theory of Mind NPC agent scores relatively good (from an *immersion* and *challenge* perspective) on the statement about the goal being difficult to achieve (statement 2), the fact that the players were forced to change tactics (statement 8), and the fact that the opponent is not easily deceivable (statement 11). This also shows in the number of lives lost by the players during the game. Furthermore, the simple reflex agent performs a lot worse compared to the memory-based and the Theory of Mind agent on most statements.

Table 1. Average ratings and standard deviation

| Statement | Simple reflex | Memory-based | Theory of Mind |
|---|---|---|---|
| *1. Fun to play* | 4.80(1.15) | 4.93(1.16) | 5.00(1.36) |
| *2. Goal hard* | 3.07(1.62) | 3.93(1.83) | 5.07(1.83) |
| *3. Flawless strategy* | 4.27(1.75) | 3.47(1.68) | 3.40(1.72) |
| *4. pro-active opponent* | 4.33(1.59) | 5.60(1.40) | 4.27(1.53) |
| *5. anticipating opponent* | 3.27(1.58) | 4.87(1.51) | 5.00(1.65) |
| *6. human-like opponent* | 4.07(1.53) | 4.93(1.03) | 5.20(1.21) |
| *7. Figure out opponent course of action* | 4.47(1.51) | 4.93(1.28) | 5.07(1.53) |
| *8. Forced to change tactics* | 4.40(1.68) | 4.27(1.75) | 5.40(1.40) |
| *9. Slow response opponent* | 2.73(1.22) | 3.20(1.66) | 2.40(0.74) |
| *10. Cheating opponent* | 2.80(1.32) | 2.87(1.60) | 2.80(1.70) |
| *11. Easily deceivable opponent* | 4.20(1.61) | 4.13(1.68) | 3.53(1.41) |
| *12. Opponent executed actions well* | 4.67(1.23) | 5.47(1.41) | 5.73(0.96) |
| *# lives lost* | 5.60(1.30) | 6.27(1.53) | 7.67(2.74) |

In order to assess how significant these results are, an ANOVA has been performed of which the results are shown below. In the Table, three columns are shown presenting the results, namely *P(SR, MB)* indicating what the P-value is of the difference in responses between the simple reflex and the memory-based, *P(SR, TOM)* for the P-value between the simple reflex and the Theory of Mind NPC agent, and *P(MB, TOM)* the P-value between the answers given for the memory-based and the Theory of Mind agent. Note that the results that are significant (with $P < 0.05$) are highlighted by a bold font.

Table 2. Statistical results

| Statement | P(SR, MB) | P(SR, TOM) | P(MB, TOM) |
|---|---|---|---|
| 1. Fun to play | 0.7542 | 0.6669 | 0.8864 |
| 2. Goal hard | 0.1811 | 0.0037 | 0.1011 |
| 3. Flawless strategy | 0.2128 | 0.1828 | 0.9155 |
| 4. pro-active opponent | 0.0282 | 0.9078 | 0.0193 |
| 5. anticipating opponent | 0.0083 | 0.0065 | 0.8187 |
| 6. human-like opponent | 0.0802 | 0.0326 | 0.5209 |
| 7. Figure out opponent course of action | 0.3682 | 0.2888 | 0.7979 |
| 8. Forced to change tactics | 0.8331 | 0.0880 | 0.0606 |
| 9. Slow response opponent | 0.3874 | 0.3736 | 0.0985 |
| 10. Cheating opponent | 0.9017 | 1.0000 | 0.9126 |
| 11. Easily deceivable opponent | 0.9126 | 0.2378 | 0.2988 |
| 12. Opponent executed actions well | 0.1091 | 0.0134 | 0.5494 |
| # lives lost | 0.2094 | 0.0135 | 0.0955 |

The results indicate that the participants experienced that the goal of the game was much harder to achieve against the Theory of Mind NPC agent, compared to the simple reflex NPC agent. This significant difference is not the case with the simple reflex and the memory-based NPC agent. Furthermore, the participants consider both the memory-based agent as well as the Theory of Mind agent more anticipating than the simple reflex agent. Also, the rating of the statement about whether the opponent actions are executed well is significantly better compared to the simple reflex agent, whereas the difference between the simple reflex and the memory-based agent is not. For the number of lives lost the same holds. One strange element is the fact that the memory-based agent is rated more pro-active than the Theory of Mind agent. This has to do with the fact that the users consider the initial behavior of the agent (waiting at the waypoint to spot the player) not pro-active according to their definition of pro-activeness whereas patrolling the non-empty rooms is (which the memory-based agent did). Positive non-significant results are the fact that they still consider the NPC agent fast (statement 9), and the fact that they thought the opponent was never cheating (statement 10).

# 6. DISCUSSION

In this paper, one of the key elements from the domain of virtual agents, namely the ability to reason using a Theory of Mind, has been applied within the domain of entertainment gaming. The entertainment gaming domain is an interesting application area due to the fact that the current state-of-the-art in the domain can profit significantly from the developments in the virtual agents research field. In addition, also for the research in virtual agents, the fact that the entertainment gaming domain places different constraints on the eventual behavior of the agent compared to other domains makes it interesting: the players are "simply" to be entertained. In order to evaluate the usefulness of Theory of Mind in this domain, an agent with such abilities has been designed. The design of the agent was evolved around PRS (cf. [4]). To allow the agent to reason explicitly about mental models of others, and select plans accordingly, the PRS agent has been extended with appropriate reasoning capabilities, in this case in the form of Default Reasoning (cf. [12]) using the mental model of the player. Using this reasoning the agent is able to identify how it should manipulate the world of the player to avoid the player from reaching its goals, and select plans accordingly.

To evaluate how suitable the designed agent is, a game has been used as a case study. Hereby, two additional agents have been designed to allow for a comparison, namely a simple reactive agent, and a memory-based agent. 15 participants played the game, and were asked to rate the various agents they played against. Based on the results, on several terrains the Theory of Mind NPC agent turned out to perform significantly better compared to the simple reactive NPC agent. However, some expected results did not turn out to be significant (such as the pro-activeness of the agent, the deceivability, and the fact that players were forced to change tactics). Of course, these results depend

highly on the chosen group (all game programmers with certain specific ideas about intelligent behavior). Furthermore, the game itself was relatively simple which might not completely show the advanced Theory of Mind behavior of the agent. For future work, a larger experiment is planned as well.

Of course, more work exists related to the application of Virtual Agents in game environments. In [6] a Quakebot is proposed which has been designed using Soar, and is able to anticipate to opponents. However in the anticipation, the bot does not use an explicit model about the opponent, but assumes the opponent has the same model as the bot itself. In the approach proposed in this paper, any opponent model can be inserted. Furthermore, in this paper the Theory of Mind model is also used to select plans to avoid the opponent from reaching a certain goal, which is not the case for the Quakebot. Related to serious gaming, [16] describes a virtual reality training based system for which Virtual Agents have been developed. Due to the nature of the domain of application, the criteria of the agents developed are however completely different from the ones addressed in this paper (i.e. the ability to learn something versus "the fun factor"). Other agent architectures for the development of Theory of Mind agents exist as well. In [11] for instance, an architecture is proposed for a social simulation tool called PsychSim which explicitly incorporates the notion of Theory of Mind. Due to the long tradition of PRS and its clear specification within the dMARS architecture, the choice has been made to use that architecture. The evaluation presented is however independent of the precise architecture used. Furthermore, as opposed to PsychSim, in the architecture presented in this paper, a Default Reasoning approach is utilized. In [2] specific reasoning rules are proposed to utilize a BDI model of other agents, and hence, reason using a Theory of Mind. Also in [14] such reasoning rules are identified. In this paper, Default Reasoning has been used as a reasoning mechanism. This makes it possible to derive all stable sets given that the information about the observations is not complete, and not everything can be manipulated, making the reasoning approach more generic than the specific reasoning rules as proposed in the aforementioned work. Similar work has been done in the user modeling community, in which Default Reasoning is utilized to discover the beliefs of a user (see e.g. [17]). The domain however differs from the application domain presented in this paper.

## 7. ACKNOWLEDGMENTS

The authors would like to thank the game studio W!Games for providing the necessary assets for creation of and participation in the use case scenario, and more in particular, Michiel Walstra, Ben Meijering and Mike van der Voort for their involvement in making this project a success. Furthermore, the authors wish to thank the anonymous reviewers for their useful comments that helped to improve the paper.

## 8. REFERENCES

[1] Baron-Cohen, S. (1995), Mindblindness, MIT Press.

[2] Bosse, T., Memon, Z.A., and Treur, J. (2007), A Two-level BDI-Agent Model for Theory of Mind and its Use in Social Manipulation. In: P. Olivier, C. Kray (eds.), Proceedings of the Artificial and Ambient Intelligence Conference, AISB'07, Mindful Environments Track. AISB Publ., pp. 335-342.

[3] Harbers, M., Bosch, K. v.d., and Meyer, J.J. (2009), Modeling Agent with a Theory of Mind. In: Baeza-Yates, R., Lang, J., Mitra, S., Parsons, S., and Pasi, G., (eds.), Proceedings of the 2009 IEEE/WIC/ACM International Joint Conference on Web Intelligence and Agent Technology, IEEE Computer Society Press, pp. 217 – 224.

[4] Ingrand, F. F., Georgeff, M. P., and Rao, A. S. (1992). An architecture for Real-Time Reasoning and System Control. IEEE Expert: Intelligent Systems and Their Applications, vol. 7 (6), pp. 34-44.

[5] d'Inverno, M., Luck, M., Georgeff, M., Kinny, D., & Wooldridge, M. (2004). The dMARS Architecture: A Specification of the Distributed Multi-Agent Reasoning System. JAAMAS 9 (1-2), pp. 5-53.

[6] Laird, J. (2001), It Knows What You're Going To Do: Adding Anticipation to a Quakebot. In: Andre, E., Sen, S., Frasson, C, Muller, J.P. (eds.), Proceedings of the 5th International Joint Conference on Autonomous Agents, ACM Press, pp. 385-392.

[7] Livingstone, D. (2006). Turing's Test and Believable AI in Games. ACM Computers in Entertainment, vol. 4, pp. 1-13.

[8] Marek, V.W., and Truszczynski, M. (1993), Nonmonotonic Logics, Springer Verlag.

[9] Niemelä, I, and Simons, P. (1997) SModels - an implementation of the stable model and well-founded semantics for normal logic programs. In: Dix, J., Furbach, U., and Nerode, A. (eds.), Proc. LPNMR'97, LNAI, vol. 1265, Springer Verlag, pp. 420-429.

[10] Peters, C. (2005), Foundations of an Agent Theory of Mind Model for Conversation Initiation in Virtual Environments. In: Heylen, D. and Marcella, S. (eds.), Proceedings of the AISB '05 symposium on Virtual Social Agents: Mind-Minding Agents. Hatfield, England.

[11] Pynadath, D.V., and Marsella, S.C. (2005), PsychSim: Modeling Theory of Mind with Decision Theoretic Agents. In: Kaelbling, L.P., and Saffiotti, A., Proceedings of IJCAI 2005, Professional Book Center, pp. 1181-1186.

[12] Reiter, R. (1980) A logic for default reasoning. Artificial Intelligence, 13:81-132.

[13] Schwab, B. (2004). AI game engine programming. Rockland, ME, USA: Charles River Media, Inc.

[14] Sindlar, M.P., Dastani, M.M., and Meyer, J.J. (2009), BDI-Based Development of Virtual Characters with a Theory of Mind. In: Ruttkay, Z., Kipp, M., Nijholt, A., and Vilhjalmsson, H.H. (eds.), Proceedings of IVA 2009, LNAI 5773, Springer Verlag, pp. 34-41.

[15] Sweetser, P., and Wyeth, P. (2005), Gameflow: A Model for Evaluating Player Enjoyment in Games, ACM Computers in Entertainment, Vol. 3, pp. 1-24.

[16] Swartout, W., Gratch, J., Hill, R., Hovy, E, Marsella, S., Rickel, J., and Traum, D. (2006), Toward Virtual Humans, AI Magazine, vol. 27, pp. 96-108.

[17] Van Arragon, P., Modeling default reasoning using defaults. User Modeling and User-Adapted Interaction, vol. 1, pp259-288.