4-22-2013 12:00 AM

An Architecture for Believable Socially Aware Agents

Arvand Dorgoly
The University of Western Ontario

Supervisor
mike katchabaw
The University of Western Ontario

Graduate Program in Computer Science
A thesis submitted in partial fulfillment of the requirements for the degree in Master of Science

An Architecture for Believable Socially Aware Agents
(Monograph)

by

Arvand Dorgoly

Graduate Program

in

Computer Science

A thesis submitted in partial fulfillment
of the requirements for the degree of
Master of Science

The School of Graduate and Postdoctoral Studies
The University of Western Ontario
London, Ontario, Canada

Abstract

The main focus of this thesis is to solve the believability problem in video game agents by integrating necessary psychological and sociological foundations by means of role based architecture. Our design agent also has the capability to reason and predict the decisions of other actors by using its own mental model. The agent has a separate mental model for every actor.

Keywords:

Believable Agents, theory of mind (TOM), Role Based Architecture, Socially Aware Agent

Acknowledgments

I express my sincere gratitude to Professor Mike Katachabaw for allowing me to conduct this research under his auspices. I am especially grateful for his confidence and the freedom he gave me to do this work.

Table of Contents

List of Figures

1 Introduction

1.1 Overview

Applications of research toward believable agents are not exclusive to video games. Human-like agents possess a great potential for research in a variety of fields such as economics, psychology, and sociology for testing validity of their theories [1, 2]. Human-like agents' applications in warfare [3, 4], crowd simulation [5], interactive drama [6-12] and educational role playing games [13, 14] have proven to be cost effective and beneficial.

Believability is largely dictated by the audience's personal and social perspective. What seems acceptable as a normative behavior in North America may seem odd in Asia. For this reason, it can be difficult to find a universally acceptable definition for believability. In most agent models there is an emphasis on designing specific aspects of human social life while ignoring others. To further complicate the believable agent-modeling problem, the very nature of human social behavior is still at the heart of much ongoing research. This diversity of approach and the lack of a unifying definition for believable agents are two factors that make this design problem a serious challenge. One of the perspectives often dismissed in agent models is the evolution of behavior driven by observations of other agents in their social environment.

Loyall presents widely accepted criteria for believability in agents [15]: personality, emotion, self-motivation, change, social relationship, consistency of expression, and the illusion of life. For the illusion of life, Loyall outlined several characteristics such as appearance of goals, concurrent pursuit of goals, being reactive and responsive, situated, resource bounded, exist in a social context, broadly capable and well integrated.

Most effective agent models apply appraisal theories to model reaction to external stimulus and encode personality traits. Despite the careful attention paid to designing the

inner workings of the agent model, little attention is given to modeling the formation of knowledge about other agents. One of the most fundamental elements of social interaction is the formation of beliefs about others. In some literature this concept has been referred to as a theory of mind [16].

This thesis addresses the problem of believability in autonomous agents by proposing emotion, Memory and planning components, and integrating them through the role concept. This integration makes the agent behavior coherent based on roles formalization without imposing predictable actions. The agent behavior and decision making is a reflection of its current state, including social context and roles that define agent's relationship to the current context. Additionally another revolutionary feature in our model is the creation and application of other agents' attributes in decision making. The agent planning and action selection in general is a direct function of how it perceives the world state, in other words from its individual perspective structured by its roles. Our design enables the designer to easily add more diversity to agent's deliberative and reactive behavior without going through fundamental changes.

1.2 Background

Many frameworks and agent models have been designed for achieving the goal of autonomous believable agent. In this section, we will briefly discuss some of the major contributions in this area. The FATIMA [17, 18, 19] architecture is one of the most well designed models that enables agents with decision making abilities, embodied emotion and personality traits. It has been used in several projects such as Fearnot! [20], orient [21], and the process model of empathy [18]. The ability to check different appraisal models and theories as well as a modular design make FATiMA distinguishable from other works. There are three main processes in the FATiMA core architecture: appraisal deviation, which evaluates the relevance of an event to the agent and determines appraisal variables; affect deviation, which takes appraisal variables as input from the previous process and outputs the Emotional State; and action selection. One of the innovative features in FATIMA is the notion of double appraisal [21], which works with the agent's projected action or its action's emotional impact on the other characters. The suggested way of doing this is to execute the agent's mind and use a projected action as if it were an

event; in this way, the agent considers the emotional impact of its action as if it has happened to himself.

EMA [22, 23] is another computational model of emotion. The three main assumptions in EMA are appraisal causes emotion; there is a cycle of appraisal and reappraisal, and appraisal is shallow and quick. EMA succeeds in designing a good emotional reaction to external stimulus. However, there is not any notion of autonomous goal generation in this model. Although EMA's emotional reaction and coping strategies are not domain dependent, EMA is still not a general model that can be used for a believable agent based on Loyall criteria.

PsychSim [24, 23, 24, 26] is an implemented multi-agent simulation tool for modeling social interaction and influences [27]. Each agent in this model has its own decision-theoretic model of the world, including other agents' beliefs. The basis of this model is the fact that human actions depend not only on their immediate effects, but also on how we believe others will react to them. Although the main focus of PsychSim is to take other agents' mental models into account, it unfortunately does so using static context-dependent stereotypes. Each stereotype has been embedded in an agent model. For example, the bully has several stereotypes for the teacher and for each choice of action he uses the partially observable Markov decision problem algorithm to check the utility of the action. One of the interesting features of this model is the ability to change beliefs, including other agents' nested beliefs, upon receiving a message. This task is done through an algorithm that measure factors such as self-interest, bias, and consistency. PsychSim is one of the few works that tries to implement the concept of a theory of mind. However it is limited by using predefined sets of stereotypes that have been chosen based on the role of the agent in a scenario.

1.3 Problem Statement

Most existing architectures for believable agents use emotion and personality models as well as goals to create illusion of life. However these believable agent architectures, fail to provide the required integration between agent components. The lack of internal data integration prevents the agent from having consistent meaningful behavior. Another

problem is the absence of the agent's individual preferences, such as personality and priorities in agent's processes, such as planning. Additionally although being situated in a social context is one of the important qualities in human social interaction, there are very few models that simulate this aspect or use it to enhance agent's architecture. Finally the great potential of using other agents' models to predict their decision and preferences has not been explored thoroughly.

1.4 Proposed Solution

The focus of this thesis is on integrating elements of classic emotion and personality model to provide qualities such as being reactive, being responsive, and having the presence of a goal and personality in the agent. At the same time we address the problem of being situated in the social context and acting coherently according to a consistent pattern of behavior by integrating internal data and processes. This is done by introducing a role based architecture and using role variables to customize agent processes such as planning and emotion appraisal to achieve the agent's goals.

We believe that in order to create a believable agent there are three main requirements that should be considered. First of all, the architecture needs a set of necessary data to identify the agents as individual entities. This includes goals, personality traits and the reward system that cause distinguishable preferences, choices and reactions from one agent to another. Second, the architecture is required to encapsulate this data in the agent's social context. A role based architecture provides this encapsulation and flexibility. Finally, the agent processes that include reaction to stimulus and planning should be directed by the agent roles.

To accomplish this, the agent data that includes goals, their priorities and personality traits should become integrated in its roles. The agent can have multiple roles that can be triggered by their target or context. Agent emotion reaction to any received event will be determined by emotion rules; however emotion responses are related to an agent's social context by means of role concept. The extent that an emotion can affect the agent is a function of its active roles' personality traits. Agent emotional state is an aggregation of appraised emotions from individual events which is also a function of role attributes. This

method of using personality traits based on context and active roles in the appraisal process is part of integrating agent's internal data.

The Memory component and implementation of theory of mind is another novel feature in our architecture. The agent is able to remember received events and their emotional effects. On top of the ability to remember other actors' actions, the Memory component has the flexibility to create a model of other actors by interpreting their actions. Other agents' models can be fed to the agent in the offline mode as well. Finally the agent uses these models to predict other actors' future actions.

A Role based architecture provides reactive and goal directed behaviors based on the social context. However our most significant contribution is applying the agent roles and models of other agents to the planning algorithm. Our planning algorithm uses the agent individualized reward system in utility assignments. This means different agents based on their goals and active roles have different utility. Not everyone in the game environment is looking for one universal goal. Using roles to calculate state utility makes our agent planning process more believable and less predictable. Furthermore the planning algorithm base takes into account uncertainty of an open world like a game environment. In facing uncertainty that has a root in other actors in the environment during planning, each agent uses another agent's mental model stored in its Memory to predict their choices. This consideration of other actors' decisions on world states and its usage as a factor in planning has not been explored before by other agent models in a dynamic manner.

In each step the agent decision is a function of its active roles, personality traits, its mental interpretation of other agent preferences and action consequences. The final result of the planning module is a sequence of actions that uses agents' roles in every step of the plan evaluation. This close integration of the agent role and planning process is unique in believable agent architecture.

1.5 Contribution

The main focus of this design is to solve the problem of Creating Believable agents. There have been some efforts in the past to solve this problem; however there is still a

huge gap with what has been defined as a believable agent and currently existing models. This gap will be discussed further in Chapter 2. Our work makes a contribution to create a more believable agent by designing an architecture that has the following qualities:

- Role based architecture: All the agent attributes have been designed and embedded by means of role components. The agent can have more than one role. This mechanism ensures that the agent performs coherent behaviour in one specific role. The agent role is a set of attributes that define its overall perception and priorities in a specific context for which that role has been defined.
- Theory of Mind: One of the main factors in the agent decision making is considering other agents in its planning, and keeping the record of their previous interactions in the Memory. The agent interprets other actors' behavior in order to be able to predict them in the future.
- Memory: The agent is able to remember previous interactions in the world as well as their emotional effects. The Memory module prevents the agent from repeating the same mistakes over and over again. Storing emotion as well as events is a novel approach that aims to imitate human Memory. Humans are not only able to remember their past experiences, but are also able to associate past experiences with particular categories of emotions.
- Socially Situated: The role based architecture facilitates the agent's interpretation of the world through the lens of its roles. A socially situated agent does not follow rigid behaviour, but rather has the flexibility to change and evolve its decision making according to what has been expected from it. The expectation according to the social context has been defined by the agent's roles. The agent's roles will be considered in all the agents' process such as: goal selection, intention recognition, planning, emotion reaction, state utility calculation, etc. Consideration of active roles in the agent process ensures that the agent's behavior has been affected by its social context. The case study in Chapter 4 thoroughly discusses the quality of being socially situated in our model.

- Planning: Our planning architecture takes into account all of the agent's roles attributes. The planning module guarantees that in all steps the agent evaluates

states or actions based on its active roles. Furthermore, our planning system facilitates dealing with the world's non-deterministic nature by introducing a non-decision node which will be discussed further in Chapter 3. Anticipation for nondeterministic situations makes the agent more believable in its decision making process.

- Acting in dynamic environments: Our planning approach ensures that the agent can deal with dynamic and non-deterministic environments like a game. This aim has been facilitated by: a) considering action consequences in the planning, b) considering other agents' possible actions, c) making the best plan by taking into account factors a and b. The embedding of personality traits and values systems (discussed in Chapter 3) enables the agent to choose the best possible action rather than staying with one particular set of actions. This best possible action will be defined not only by the agent's variables but also based on the social context that makes our agent to choose a different action while facing same problem that involves different parameters.
- Flexibility: The role based architecture enables the user to design their desired agent with customised qualities. This architecture is easy to understand and guarantees coherent interaction of agents in the world based on the authored role.

1.6 Outline

Chapter 2 discusses the definition and application of believability. We briefly review classic AI techniques that are widely used in the game industry. Then we will discuss the general BDI paradigm and agent models that attempt to go beyond classic AI techniques. Chapter 3 is a detailed description of our agent architecture. We present components and the data flow to create the illusion of disbelief. The planning component explains thoroughly how other agents' models can be reflected in decision making. Chapter 4 represents our case study on a prisoner's dilemma and demonstrates how our model can contribute to simulate believable socially aware characters with unique characteristics that take other actors' decisions into account. Chapter 5 concludes this thesis, providing a summary of the work and a discussion of directions for future work in the believable agent area.

2 Literature review

Believability is largely dictated by the audience's personal and social perspectives. Therefore, it would be difficult to find a universally acceptable definition of believability. The diversity of approaches and the lack of a unifying definition for believable agents are two factors that make this design problem a serious challenge.

Video Games are an interactive medium. However, the lack of emotions and personality in Non-Player Characters (NPC) is the main factor that prevents the audience from being immersed in the game environment. In current games, the player can easily fool NPCs; they do not have any character or individuality other than performing repetitive tasks in the game environment, which quickly causes boredom, frustration or loss of engagement. The motivation for embedding emotions, personality, and decision-making into NPCs is to create human-like agents. This is a central problem of game AI that cannot be resolved by current approaches. This change firstly affects the game environment and narration and secondly provides the opportunity for the video game industry to target a larger group of audiences.

In this chapter we will start with a definition for believability, and then briefly discuss applications for believable agents in Section 2.1.2. We will follow the discussion in Section 2.2 with a brief review on two classic methods for game AI: Finite-State Machines (FSM) and Rule-Based Systems. In Section 2.3 we take a deeper look at psychological foundation for emotion and personality. In Section 2.4 we will discuss BDI (Belief, Desire, and Intention) as a general architecture that attempts to address believability in agents. In Section 2.5 we will present the summary.

2.1 Believability

Characters in plays, movies or novels have been created by the mind of an artist. Once the story is created, it cannot be changed. On the other hand the main power of video games as an entertaining medium is the active role of the audience or player. The

problem here is how the designer can use this great potential of an interactive environment for players. We believe one of the main factors is to create a dynamic environment by using emergent and believable characters.

A Socially believable intelligent agent has the potential to make a significant change in audience immersion. Believable autonomous agents can open a new area in game narration. The game environment can be managed with a minimum level of supervision of the story. Consequently the player's choices and actions in the absence of a static plot like those that are already in use can make meaningful changes. Furthermore, the application of a believable agent is not limited to video games. Believable human like agents have a high potential to be used in critical areas of research such as warfare [3,4] and crowd simulation [30]. Last but not least, they are a cost-effective method to test theories on human behavior in diverse areas of research such as educational role playing [13, 14], and psychology [5].

When we talk about a believable agent in the video game industry, we are looking for an illusion of life; in other words, a suspension of disbelief. In real life situations, humans are not perfect. Their behavior and choices do not always confirm predictions. Our decision-making process does not always satisfy a utility function. At least, not all humans have the same priority and utility across different choices. In the next paragraph a well-known problem from decision theory will be discussed. In that problem, inconsistencies between the results from real experiments and decision theory's pure mathematical predictions will be presented.

In this scenario, which is often referred to as the dictator game [43], there are two different roles: the giver and the receiver. The giver is in charge of dividing and distributing a finite amount of money (let's say $100) between himself and another person. Based on decision theory, because the other person has no choice but to accept this distribution, the giver can decide to give him the minimum amount (here $1). From the decision theory standpoint it is obvious that the most logical solution for the giver is to minimize the gift and maximize his portion. However, psychological testing on human

subjects shows that people do not always do as decision theory suggests. In other words, givers usually give more than the minimum amount.

The results from testing the dictator game on human subjects supports our argument that not all humans adhere to the same utility function. The dictator game is only an example to illustrate the failure of classical approaches such as game theory [52] or decision theory in predicting human behavior in real life situations.

2.1.1 Believability in Recent Literature

In order to come up with a general, well-defined model for believable agents, we first need to recognize the requirements and then build them into the agent. One of the widely accepted sets of criteria in the field of believable agents is Loyall's definition [15]. He analyses requirements for believable agents from two main perspectives. The first is artistic and the second is experience with the agents. In the following paragraphs we will briefly discuss his criteria:

- Personality: Loyall defines personality as the most important requirement for a believable agent. In Loyall's definition, personality is what gives all the agent actions and behavior consistency. There are many personality models in psychology, but Loyall does not suggest any particular model. What he does suggest as a definition of personality is rather complicated to implement. From a theoretical perspective, there are many different ways to define a personality while there is no single model or set of variables widely accepted for doing so. However, even after choosing a model or set of variables, the harder task is to represent this personality in every aspect of the agent. From Loyall perspective, just as in a movie, two characters impose their personality on every single action. Two agents should be able to represent their personalities in the same manner. By his expectations, imposing different characters to the same event should have versatile impacts.
- Emotion: The second criterion in Loyall's list is emotion. He states that emotions are necessary for believable NPCs. It is a fundamental requirement for human-like agents to not only demonstrate emotions but also to understand others' emotional reactions. In order to implement this aspect the agent needs knowledge that defines

what is emotion and represents that based on their personality. The way emotions are felt and how they affect others is yet a complicated problem on its own. The other important feature from the emotion perspective is how an agent will be able to communicate its emotional state. When an agent feels sorrow it should be able to show that emotion in the environment by bursting into tears. Additionally, the agent's emotional state should be reflected in the agent decision making process. An agent who is furious will not be able to take a rational action just as a human in rage.

- Self-motivation: Self-motivation here identifies a type of agent that does not simply react to the environment but also has some internal goals and motivations. Most of the previous works in the field of believable agents focuses on believable reaction to outside stimulus but does not pay as much attention to embedding autonomous goals in an agent to cause it to follow its own intentions. How and when and how much an agent feels these inner motivations are some important problems to consider in order to make the agent more believable.
- Change: The agent's growth and change are fundamental characteristics of a believable design that cannot be arbitrary. They should happen based on the agent's experience and characteristics. In classical drama, character change is one of the leading and turning points in the sequence of a developing a story. However, when it comes to automating this process in a game environment, there are many obstacles that the designer must overcome.
- Social Relationship: Humans are social entities. It is a requirement for a believable agent to be able to engage in social interactions and maintain this relationship with a social network.
- Consistency of Experience: The agent behavior and emotion expression should be consistent. As mentioned, the personality part of the agent needs to have one unified spirit that could be demonstrated through its actions. From the macro perspective, the entire agent's behavior should be consistent. From the micro angle, this consistency should form every single action of the agent.

- The illusion of life: The illusion of life categorizes wide range of variables in Loyall's classification including appearance of goals, concurrent pursuit of goals and parallel action, reactive and responsive, situated, resource bounded and existence in a social context.

In conclusion Loyall describes how an agent architecture designer should take several factors into account which are taken for granted in conventional media as well as a plan for applying those factors to the agent. For example, an actor in a movie can walk, eat and get engaged in a conversation without any difficulty but for an agent to be able to do several tasks in parallel will not be as easy. Until now we gave a thorough definition for believability in human-like agents. In the following we introduce the application of this type of agents.

2.1.2 Applications for believable agents

One of the main applications for believable agents is in the video game industry. Believable NPC helps players to immerse themselves in the game. Repetitive and inconsistent behavior based on designer's decisions or hardcoded techniques can easily break the immersion. On the other hand, human-like agents can add a new dimension to players' experience in the game by evoking emotion such as empathy. Furthermore, the new generation of video game architecture seeks to involve dynamic narrative in the story telling in order to provide the player with a unique experience of his own.

The future of video games is in providing a unique narration for each player, by facilitating a mechanism to use their action in making meaningful changes in the story line. This aim can be accomplished by populating the environment with emergent autonomous agents and a drama manager [6, 7, 8].

Additionally, human-like agents' application in warfare [3, 4] crowd simulation [5] and educational role playing games [7] has proven to be cost effective and beneficial. Virtual environment is another field that can receive benefits from believable NPCs. Recent research has proposed populating virtual environment with believable human-like agents

for medical purposes. This type of environment can be used as a method of rehabilitation for patients who suffer from psychiatric [58, 59].

2.2 Classic Techniques in AI

NPCs have been conventionally hard coded in games. This method is computationally desirable for the game environment. Also, from the design perspective it does not need a complicated architecture. In this Chapter we take a look at two common methods for game AI: FSM (Finite State Machines) and Rule based system. Note that traditionally, the purpose of using AI in games was not to address the agent believability. Our discussion in this section attempts to highlight the fact that the classic approach on its own is not sufficient to make the agent believable.

2.2.1 FSM

A finite state machine is one of the simplest and mostly used methods for designing intelligent agents. A Finite state machine is an abstract model consisting of a set of states and transition functions which map a given state and input condition to another state. Each state determines the agent behavior [54].

One of the biggest advantages of this architecture is its simplicity. A Finite State Machine can be easily implemented by set of if-then rules or switch cases [27] This implementation has a compelling advantage which is the low computational cost. Understandability, predictability, little infrastructure support and deterministic behavior are among other strong features that help this architecture survive since the very early days of computer game programming.

Despite all these advantages in a complex environment when range of possible actions and consequently number of states increases FSM architecture will not be a reliable solution. The reason is all possible situations should be encoded in the rules by the designer, which places a huge burden of responsibility on his shoulders. Furthermore it’s a static method that does not address the dynamic nature of video games.

Mac Namee et al. called gents solely based on FSM as reactive agents and have identified them as the simplest form of intelligent agents [28]. The authors here argue that lack of an internal world model makes this model incapable of reasoning. Based on the above description, the reactive agent can easily break a player's immersion with its repetitive actions and it will be easy for the player to fool it with his moves. In conclusion the cheap cost of reactive agent implementation results in fragile performance; therefore it does not fully satisfy play experience for the audience. Still this does not mean that the believable agent architecture should fully stop using FSMs, but in order to create a socially situated agent, this approach will not be enough.

2.2.2 Rule Based system

A Rule based system is another method for designing intelligent agents. Basically a rule based system has three main parts: working memory, rules memory and an inference engine. Working memory stores known facts and assertions made by the rules. The rule memory contains if-then style rules that operate over the facts stored in working memory. In a rule based system once a rule fired in the system, it can trigger some action or state change, or the content of working memory can be modified by adding new information called an assertion [27].

The main component of a Rule based system is an inference engine. There are two basic algorithms for making inferences: forward chaining and backward chaining [54]. Forward chaining is the most common algorithm for rule-based systems, which involves matching rules to facts stored in the working memory. It does this by checking if statements. Potentially more than one rule can match the given set of facts in working memory. Then it triggers a selected rule or rules and the whole process is repeated until no more rules can be fired. Backward chaining is the opposite it starts from an outcome or a goal and tries to figure out which rule should be fired to arrive at that state.

Inference engines and specifically backward chaining provide the agent with planning ability; that's why Mac Namee et al. [28] named this category of agents "deliberative" [28]. Working and rule memory are the models of the world in the agent's mind that a reactive architecture like FSM is missing. However this method also has certain

drawbacks. First of all it needs constant maintenance of a knowledge base which in a real time environment may cause problems with performance. Furthermore inference based reasoning is computationally expensive. Finally and most importantly, due to the structure and search method, a Rule Based System will not be able to meet the real time requirements of a believable agent.

Scripting is a special case of a rule-based system. Each object and character in the game environment may have dozens of scripts that determine their behavior in a particular situation. In each frame all of these scripts for the object should be checked to see if conditions in the frame satisfy any of the attached scripts. Those scripts which have the matching conditions will be executed. Scripting has the same disadvantage as FSM; the designer must predict and hard cod all possible situations and circumstances.

2.3 Psychological Foundations

In Section 2.1 we discussed the definition of believability and in Section 2.2 we discussed traditional methods in AI. In this Section we are reviewing the psychological foundation for two of the main requirements for the agent believability from Royal's perspective: personality and emotion.

2.3.1 Personality

Many researchers define personality as one of the main requirements for believability. In Loyall definition, personality is what keeps all the agent actions and behaviors consistent. Basically, he believes that personality is what makes us distinguish between two characters in a movie. It is based on their manner, tone of voice, outfit, or even their favorite hobby. He emphasizes that personality is one of the fundamental requirements of a believable agent. However, the main problem with personality is how it should be designed and how NPC will be able to represent this concept. We will provide a preview of the personality definition in psychology. Then we will present some of the most prominent personality models and how researchers in the field of AI implement them.

Oxford dictionary defines personality as the complex of all the behavioral, temperamental, emotional, and mental attributes that characterize an individual. In [5],

Durupinar et al. encompassed the same terminology though they defined personality as a pattern of unified behavioral, temperamental, emotional, and mental traits. Samuel Ma states that personality has a high influence on decision making, action selection, expressiveness, and character behavior [29]. Although we stated that emotions also have an effect on the above parameters, the main difference between emotion and personality is the fact that personality is long lasting and persistent while emotions are temporary. In other worlds, emotions give us an idea about how the agent would behave in a particular state of mind, but personality is the causal reason that the behavior occurs within that state of mind.

The rationale of introducing personality in a multi-agent system is to provide diversity in agent behavior and their choice of actions and planning [44]. During social simulation, having different personalities enables researchers to test different strategies [28, 29, 30]. In the case of human-like agents in the virtual world, personality is a key ingredient in creating a suspension of disbelief for the audience. Personality enables the designer to make each character unique and represents this coherently through different media in the virtual world. Johns and Silverman, in [31], identified personality as a dimension of individual differences that should be considered to determine what causes people to choose different alternatives despite holding similar emotional states. It is a measure to explain why people follow different goals, and demonstrate different thinking processes and emotional responses.

Personality theorists in the psychology domain have strived for decades to define basic dimensions of personality that generate the differences among people [32]. Generally speaking there are two main stream theories: trait theories and social learning theories [29]. A class of traits theories define what traits are and assume that a unique combination of these traits are ingrained in each person's mentality. Based on these theories, future behavior and the decisions of an individual can be predicted if we have enough knowledge about their traits. However, social learning theories argue that personality can be changed from different situations and new experiences.

2.3.1.1 OCEAN Personality Model

One of the most influential personality models in the field of AI is OCEAN or the Five Factor Model (FFM) [5], also known as the Big Five. In this model, personality can be devised into five dimensions: openness, conscientiousness, extroversion, agreeableness, and neuroticism. These traits have been defined in a bipolar fashion so we can view them as a range from -1 to 1. The highest score in each trait means the character is in harmony with that trait and the lowest score demonstrates opposite qualities of that trait are being projected. In a virtual world, if a person is not very social and during a social occasion does not feel content, his score on the extroversion scale would be lower than 0 with a default range of -1 to +1. In the following paragraphs, what each trait stands for will be discussed [25, 33, 34].

- Extraversion: Extraversion influences the frequency of social interactions. That is to say, the more extroverted a character is, the more they will interact and appear outgoing. This trait also influences the interpretation of positive versus negative interactions. Extroverted agents place more importance on positive events than negative ones. Therefore, the effects of Encourage, Agree, Facilitate Problem and Gain Competence interactions increase as extraversion increases, while the effects of Discourage, Disagree, Obstruct Problem and Lose Competence are reduced.
- Agreeableness: Agreeableness influences the frequency of positive socio-emotional interactions. It determines the level of friendliness, generosity, and the tendency to get along with other people. More agreeable agents agree more often with others and encourage others more frequently. In addition, agreeable agents perform more actions for the group rather than for themselves. On the positive end of the spectrum, expressions of altruism, concern, and emotional support are shown, while the negative end represents hostility, self-centeredness, spitefulness, and jealousy.
- Conscientiousness: Conscientiousness indicates the level that the individual is governed by conscience. In this context, conscientiousness determines the level of which a person is organized and careful. Conscientiousness affects the planning algorithm. It affects the degree to which an individual considers the full consequences of his actions before taking them. Higher levels of this trait allow one

to avoid courses of action that lead to negative consequences even if they are accompanied by substantial positive actions. Furthermore, there would be an unlikelihood of choosing a course of action that would be considered dishonorable or risky. Often this may be at the expense of missed opportunities but success is also related to this trait.

- Neuroticism: Neuroticism refers to emotional instability and the tendency to experience negative emotions. A high level of neuroticism causes an individual to place more importance on negative events rather than positive ones. For decision-making purposes, this trait governs the degree to which a human being is willing to experience stress, pain, or take risks in the pursuit of goal achievements.
- Openness: Openness describes the imaginative and creative aspect of a human character.

2.3.1.2 Reiss's 16 motive Model

Reiss's 16 motives is another interesting model. Reiss contends that motives are the reason governing people's voluntary behavior. They indicate the meaning of human behavior, and may reveal a person's values. Motives often affect a person's perception, cognition, emotions, and behavior [35]. There is a motive behind each consequence of action in reaching one's goal. Furthermore, these motives can affect one's behavior unconsciously or indirectly. Consequently these motives that are created in a human mind are what form the personality.

Although the OCEAN personality model provides a general framework to investigate a human being's personality, using this model to create personality in human like agents needs careful consideration. The OCEAN model does not provide mappings from traits to human cognitive process such as emotion appraisal [22, 23], decision making or intention recognition. This lack of a well-defined mapping makes it a bit of a challenge to apply it practically in human like agents.

Another interesting personality model in scholars is Reiss's 16 basic motives theory [35]. The basis of Reiss's theory is laid in his eight hypotheses [35]. We are going to bring attention to those hypotheses in order to understand this model with more depth:

- Each of the 16 basic desires is a trait motive (Hypothesis 1). This is the first hypothesis in Reiss's model. Everything in his model is based on these 16 traits and how people decide to achieve them.
- The satiation of each basic desire produces an intrinsically valued feeling of joy, a different joy for each basic desire (Hypothesis 4). These different types of joy satisfy a need.
- Everybody embodies the 16 basic desires though individuals prioritize them differently (Hypothesis 5). People behave as if they are trying to maximize the experiences they have of these 16 intrinsic joys with their own individual priorities; this is what makes people behave differently or seek different goals. In other words, these 16 desires are encompassed in every human being but what differentiates them is how much value they place on each desire. From another perspective, one unusually weak or strong desire on one of these traits can be used to define one's personality. For example, this taxonomy can be used to demonstrate a power hungry personality when someone has a strong desire to gain power over others but has a normal set point for all other traits.
- Each basic desire is theoretically regarded as a continuum of potential motivation anchored by opposing values (Hypothesis 6). The theory of 16 basic desires holds that individuals are motivated to aim for a point of moderation, (called a set point or sensitivity). Set points are what an individual is aiming for. Once they reach that set point, the desire is fully satisfied however temporarily.
- The theory of 16 basic desires holds that motivation is based on discrepancies between the amount of an intrinsic satisfier that is desired and the amount that was recently experienced (Hypothesis 7). When a person experiences more power than he or she desires, the individual is motivated to be submissive for a period of time to balance the experience toward a desired rate. When a person experiences less power than he or she desires, the individual is motivated to be domineering for a certain period of time.
- Basic desires organize our attention, cognitions, feelings, and behavior into a coherent action (Hypothesis8). We pay attention to the stimuli that are relevant to

the satisfaction of our desires, and we tend to ignore a stimulus that does not satisfy our desires.

In conclusion, Reiss makes a case that psychological needs are linked to motivations. He provides a conceptual framework based on an analysis of the nature of basic human desires and psychological needs. Moreover, he makes a solid connection between basic needs, traits, motivation, and personality type analysis. This allows for a sensible coupling of personality descriptions within social change [6].

Reiss's model can benefit a designer with designing goal based characters that have the same motives as humans. It also provides guidelines on how to make different personalities and how to make them distinguished. However, this model is still abstract in the sense that it talks about satisfying a motive through traits within an extremely broad setting. Additionally, implementing all of these 16 traits for NPC in the world is computationally expensive and seems unnecessary. Furthermore Reiss distinguishes between individual's personalities only based on their motives and dismiss other temperament. Many scholars think that personality should distinguish between individual emotional responses and action selection. These two factors have not been considered in Reiss's model.

2.3.1.3 Summary

In this section we discussed the definition of personality from scholars. We also explained two widely accepted models of personality: OCEAN and Reiss's. The OCEAN model is general and expressive enough for explaining the nature of human differences through personality; however it is too abstract to be applied in a computer agent. Unfolding traits to low level behavior in this model is subjective to the designer choice. Reiss's model defines personality dimensions based on human motives and categorizes them into 16 motives. Reiss's model clearly explains and distinguishes how each trait is related to a motive in human beings; however it does not explore other dimensions that can be used to distinguish between two different personalities such as emotional reaction or action tendency.

2.3.2 Emotion

Emotions in [36] have been defined as current states with a specific quality and intensity as an outcome of complex physiological processes for communication. The processes include neural activity of the brain as well as physiological responses of the body. Marcella et al. state that, emotions arise from social conditions which set conflicting interpretations of the goals a human wants to achieve and what the world around him suggests[38]. From Marcella et al.'s perspective, Emotions arise from the dynamic and continuously evolving world that leaves our decisions and actions with uncertainty. Marcella et al. goes on to propose that emotions are a means of navigation during social contexts and when dealing with uncertainty if an individual has limited control over future events.

Additionally, Carnevale et al. [39] say that emotion not only affect an individual's decision making, but also their interpersonal decision making processes. He states that emotional expressions are not simple manifestations of internal experience but rather the methods people choose in order to communicate their beliefs, desires and intentions. For instance, guilt occurs when someone transgresses an accepted social norm signaling regret, which serves as an apology, and in turn contributes to avoided reprisals from others.

Some researchers prefer to categorize emotion into two classes of primary and secondary emotions [39]. Primary emotions refer to fast and reactive responses such as when one is experiencing immediate danger. The other side of the spectrum holds secondary emotions. These are based on the ability to evaluate preferences over outcome and expectations: for example, feelings of 'relief' or 'hope. A fundamental aspect of appraising secondary emotions is dependent on the situational and social context. Consequently, they are more reliant on the agent's cognitive reasoning abilities. EBDI[40,41] and WASABI discussed in [42] are two architectures that take into account these classifications.

There are different motives in the scientific research community for creating a computational model of emotion. In the field of psychology a computational model of

emotion permits the comparison of different theories in each respect so as to improve the intuitive understanding of said parameters [45].

In the engineering field, the motive for modeling emotion is an indirect one. Modeling emotion is a vital requirement in the creation of a virtual human [46]. In the field of human-like agents there are certain applications that should be considered for emotions in a design. Taking into account how intense emotions impact people's behaviors and decision making processes is an important aspect in modeling traumatic events [5]. Fundamentally developing and expressing an emotion response is one of the requirements for NPCs. The inability of a character to reveal its emotional state would possibly be interpreted by the user as missing sympathy [47]. The character that always has the same emotional state, regardless of its context, breaks the sense of immersion for the player. Expressing emotions plays an important role in increasing the believability of the character in the game. The agent should be able to feel anger and express that upon an unpleasant event such as getting hit by other agents. The agent which communicates its emotions can have a bigger impact on the player by creating a sense of empathy for him. Emotions have a great influence on humans when planning, making decisions, and during social interactions. The agent in rage will not be able to make rational decisions or assume optimal actions. A designer should find a way to link the agent's decision making process to its emotions.

So far the essentiality of emotion-embodied characters has been discussed. Ultimately, characters that interact with humans need a model to synthesize emotions and express them. With the same logic, they should be able to sense the emotional states of other actors including humans or agents. In order to propose a critical component of more effective human-computer interaction, which factors in the emotional state of the user, we need emotion models in virtual humans [48].
It might be useful to distinguish two classes of computational models of emotion: Black Box models and Process models. These two approaches differ in the degree of abstraction of intervening variables [48]. Black Box models focus on the input-output relationship. They provide little information concerning the mechanisms involved yet can help when investigating sufficient variables as well as facilitating a sound ground for practical

decision making [48]. Process modeling attempts to simulate naturally occurring processes. Each approach has its own advantages and disadvantages. The Black Box approach seems more feasible and computationally has a cheaper cost for development. However, in the case of manipulation and performing in social contexts, it needs to be studied more carefully.

There have been several emotional models developed. Ortony, Clore, and Collins classified emotions based on the stimuli that generated them to twenty-two different types of emotions which is often referred to as the OCC model [47]. These categories, based on valence reactions to situations, are constructed either as being goal relevant events, as acts of an accountable agent, or as attractive or unattractive objects [47].

In the Classification phase, the character evaluates an event, action or object. This determines which emotional categories have been affected. For this part, standard goals and attitudes need to be specified, organized, and stored by the designer of the character [47]. One way is to put all the possible events, actions, and objects in a table and for each row in the table specify which emotional categories may be affected with what level of intensity along with how this intensity should be calculated. For a limited range of actions in a simple world this method could be applicable. However as the complexity increases, it becomes more infeasible to do so and the system needs a definition of the types of abstractions to be able to handle this part. To summarize, the agent needs some sort of knowledge base as well as abstraction to identify what is going on in the world.

There are several reasons for the simplification of the OCC model [47]. As an example, if the agent uses an emotional model to change facial expression, its emotional categories should be limited to the ones it can express. Ekman proposed six basic emotions (anger, disgust, fear, happiness, sadness, and surprise) that can be communicated efficiently through one's facial expression [60]. Interestingly, he suggested that these six basic emotions are cross cultural. When the goal of the architecture is to make a believable agent, the emotional categories should not be considered if they do not add up anything from the believability perspective. Examples would include the importance of an agent to express happiness from receiving a hug from a friend and show sadness or

disappointment from an unjust slap. A character that is unable to distinguish between these two actions (receiving a hug and being slapped) would easily break the player's immersion. Similarly, being able to sense gratitude and gratification is based on two different trigger conditions: gratification results from a praiseworthy action the character did themselves whereas gratitude can stem from another's action. A question that arises here is whether this distinction adds any more believability to the agent. If yes, how could the agent express them in different manners?

Another flaw of the OCC model is the negligence of explanation in how different emotions should merge together. For instance, when an agent is in a low mood while receiving a gift (this may make him less sad) or when an agent is extremely angry while receiving a hug (this may make her less angry but could not bring their mood to a full cheerful state). How these different emotions are mixed and interacting with each other have not been mentioned in the OCC model.

Lastly, some emotional categories in OCC such as "be happy for" are based on the desirability of action consequence for the action target. The agent needs to be provided with a specific agent model so that it could decide the desirability of action for others. In [17], the author discussed empathy in resolving this issue and there was research that offered methods for creating the player model. Ortony simplifies the OCC model by excluding categories that need another agent model through five positive and five negative categories. These include joy, hope, relief, pride, gratitude, love, and distress, fear, disappointment, remorse, anger and hate, respectively.

2.3.2.1 Summary

Emotion is an important element that can aid the audience by immersing them in the game scene. Simulating emotion has another application to test the validity of theories and design effective interactive applications. In the field of psychology, the nature of human emotion has been studied. Although there has not been a universal agreement on a universally acceptable definition of emotion, some models such as Eckman and OCC attempt to explain the mechanism of emotion reaction in human beings. However applying these models to the agent without careful consideration is not feasible.

2.4 Believable Agents

Traditional AI techniques such as FSMs and Rule Based Systems are not very effective when it comes to creating believable behavior as discussed in Section 2.2. The designer needs to hard code all possible situations and behavior and any mistake or unexpected event can easily break the audience immersion. In Section 2.3 we discussed some psychological foundations such as emotion and personality that have been defined as main requirements for believability. In this Section we briefly discuss a modern approach toward making agents more believable.

2.4.1 Modern Approach

BDI is a general framework which defines rational agents based on belief, desire and intention, which represent information, motivational and deliberative states of the agent perspective [50]. From the philosophical point of view, this framework discusses how a living being takes or cause an action in the world.

The BDI model provides a reasoning pattern for action selection based on rational choices. Instead of jumping from a desire to an action in one step, there will be two processes: first determining an intention, and then choosing an action. This separation makes the platform stronger than a simple rule-based system. Also, it results in more believable characters that have the ability to reason about their environment and make decisions based on their internal mental/emotional state, beliefs and motivations.

BDI discusses a very simple statement that if human beings have a desire to satisfy their hunger and if they think eating a cake is a way to achieve that desire, then they will eat a cake, or at least develop an intention to eat a cake once they get hungry. This basic example illustrates how BDI interprets actions. We need to embed this natural interpretation in the agents in order to make them believable.

One of the most important aspects of BDI which has been pointed out in [50] is the notion of commitment to previous decisions which makes a balance between reactivity and goal-directed behavior. At the same time it gives the agent the stability and consistency in reasoning. Two parts have been identified for this notion: the commitment

condition and the commitment termination. The former is what the agent is committed to maintain (for example: belief and desire) and the latter is the condition under which the agent gives up the commitment.

Most of the models that implement BDI platform use traditional AI techniques such as rule based systems, fuzzy logic and different planning systems in their components. However, as we mentioned before, the problem with traditional AI techniques is the infeasibility to manage all necessary requirements for a believable agent.

On the other hand, one of the requirements for a believable agent is the ability to function in a social context. The Theory of Mind (TOM) discusses the process of decision-making based on forming and using recursive models of others. The BDI architecture helps us to provide the agent with this nested belief structure and to use it in decision making. In virtual characters, this option helps actors to predict the behavior of other characters and take an action based on this prediction. PsychSim[24] is one of the few works that tries to implement the concept of a theory of mind.

EBDI (an architecture for emotional BDI agent) is another generic architecture for an emotional agent that expands BDI by implementing practical reasoning techniques separately from the emotion mechanism. It adds the influence of primary and secondary emotions as has been discussed in Section 2.3.2 into the decision making process of the BDI architecture. Additionally, it has the flexibility to have different emotional models and reasoning engines plugged in. In EBDI, when there is new information from the environment, the agent generates belief candidates which together with current intentions trigger emotion updates. Based on the new emotion status and the new information, together with current intentions as a guide, the agent re-evaluates its beliefs. Then it generates desires based on its beliefs and intentions. Finally, under emotional influences, it picks the best option or intention. In this stage the secondary emotions are triggered.

2.4.2 Believable Models

There are many agent models that have been designed for achieving the goal of autonomous believable AI. In the following paragraphs we briefly discuss some of the major contributions in the area of believable agents

PsychSim [14] is an existing multi-agent simulation tool for modeling social interaction and influences [13]. Each agent in this model has its own decision-theoretic model of the world, including other agents' beliefs. As has been mentioned earlier, PsychSim is one of the few works that tries to implement the concept of a theory of mind. However, it is limited to using predefined sets of stereotypes that have been chosen based on the role of the agent in a scenario. The basis of this model is the fact that human actions depend not only on their immediate effects, but also on how we believe others will react to them. Although the main focus of this PsychSim [22] is to take other agents' mental models into account, it unfortunately does so using static, context-dependent stereotypes. Each stereotype has been embedded in an agent model.

In the field of believable agents, the FATIMA architecture is one of the most well designed models that enable agents with decision making abilities, embodied emotion and personality traits. One of the innovative features in FATIMA is the notion of double appraisal, which works with the agent's projected action or its action's emotional impact on the other characters.

The OZ project [55] was one of the very first systems that tried to integrate a wide range of capabilities such as perception, reactivity, goal-directed behavior, emotion, social behavior, natural language analysis and natural language generation, goal-directed reactive behavior. EMA[22] is another computational model of emotion. The three main assumptions in EMA are appraisal causes emotion; there is a cycle of appraisal and reappraisal, and appraisal is shallow and quick. Unfortunately there is not any notion of autonomous goal generation in this model.

2.5 Summary

In this chapter we introduced the concept of believability in agents. We briefly discussed the applications of human-like agents. We discussed the infeasibility of classic techniques in AI to facilitate believability. We also discussed emotion and personality as psychological foundations, which have been defined by Loyall as main requirements for believability. Finally, we describe a general framework and some existing models that attempt to solve the problem of believability. Still, there is a huge gap between definition believability from Loyall and current models. Our proposed model in the next chapter aims to fill this gap.

3 Proposed design

In this Chapter we propose a novel architecture to address believability in NPCs, considering the believability criteria that has been discussed in Chapter 2. Our model facilitates reactiveness as well as active pursuit of goals in role based architecture to unify the agent characteristics. In the following paragraphs we give a brief overview of the model contribution:

- Role Based Architecture: In the proposed architecture all agents' qualities and attributes have been formalized through a role concept. Every agent in the environment has a context free *default* role that defines the agent in general context. The role integrates agent goal directed behavior, personality traits, preferences and facilitates their application in the agent process.
- Theory of Mind: Theory of mind (TOM) is one of our novel features that help the agent to consider others' preferences during decision making. Theory of mind enhances the Memory module to not only remember previous interactions but also interpret them to generate a model of other actors. These models will be used in the agent in process of decision making, and provide it with the ability to dynamically update his social knowledge about other agents' personality.
- Practical approach to psychology foundations: Many believable agent architectures use personality models to create the illusion of life in their agents. However, one of the main technical difficulties with applying personality models is putting their specification into practice to create a coherent set of behaviors [5]. Part of this difficulty is due to lack of well-defined connection between traits and how they affect the agent processes. Our model categorizes personality traits to: reactive, behavioral and context free traits to facilitate this connection. On top of this categorization the agent data flow that will be explained in Section 3.1 ensures that these personality traits will be used in the agent process. This categorization for personality trait does not affect our model general quality to use any specific personality model. As long as the designer identifies a specific personality quality and how the agent cognitive process should be affected by, it can be applied to the agent.

- Customized Planning: Planning and path finding is a well-studied area in AI, however until now no model can be found in the literature that uses the agent internal variables (including the agent mental and social context such as roles, personality trait and emotional state) to evaluate the planning tree. Our proposed planning algorithm uses agent internal variables to evaluate available options in any probable state and finally chooses the best possible plan by considering both utility and the agent role.
- Memory: Memory has not been defined explicitly as a requirement for a believable agent, however to follow a plan and actively pursue goals the agent should be able to store world state. Our proposed Memory model helps the agent to store the emotion appraised from every event. The agent can remember a series of events and their affection on its Emotional State

Agent process manages and connects all agent modules and unifies them to create believable behavior based on agent roles and the current state of the world. The requirement specifications for our agent model are: a) demonstrate emotion reaction towards world events, b) facilitation of goal directed behavior in the agent. Agent process should be able to satisfy these criteria by using agent role to channel embedded individual characteristics in decision-making.

3.1 Data Flow

The agent data flow is shown in Figure 3.1. The agent process starts by receiving a message from the world. The event will be appraised by emotional rules; active roles that represent the agent social context affect this appraisal process. The received event may be saved in the Memory; furthermore it may cause the agent to update its original plan. Intention recognition selects a goal and planning generates a plan for this goal. Arrows from agent active roles to all other modules demonstrate how the agent process has been affected with active role components.

Agent role has a set of personality traits, beliefs and goals that will be discussed in detail in Section 3.5. The Memory module including the agent TOM profiles is discussed in Section 3.4.

In the following paragraphs we take a closer look at each of these steps:

- Perception: The agent receives the information about the world state through message passing (arrow 1). The Perception module task is to receive these messages, unfold the message and pass the event to the emotion module.

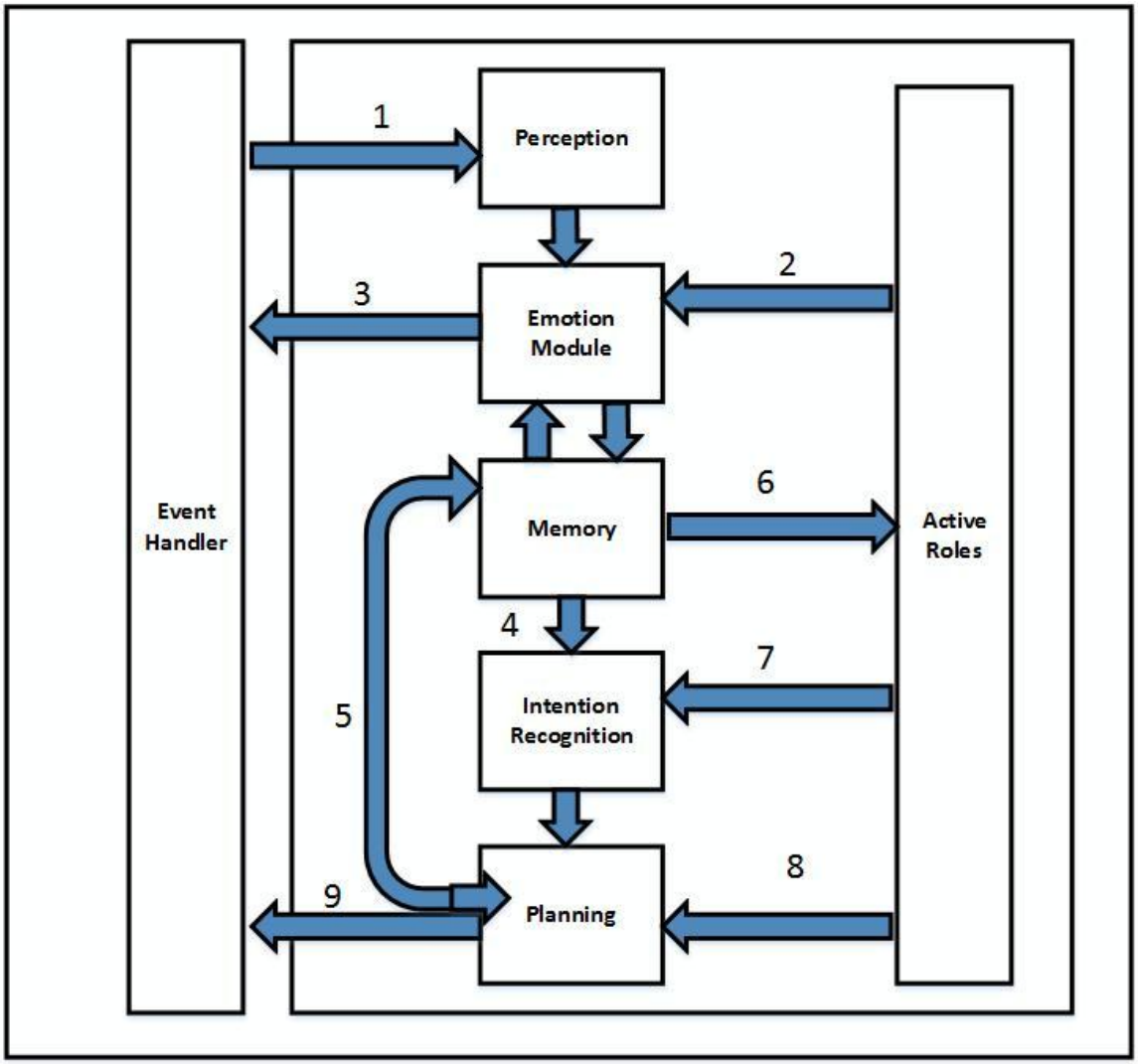

Figure 3.1: Data flow in the agent architecture.

- Emotion Module: Emotion and Memory work together to appraise emotion from the received event. In the first step, the emotion module uses general emotion rules to appraise an emotion. For example if the agent loses his wallet, it will make the agent upset. However, there are other types of events that need the Memory module

consultation, if an event confirms a goal state achievement it cannot be appraised simply by applying emotion rules.

Collaboration of emotion and Memory is necessary to apprise the group of events that makes any change on the agent active plan or previously stored goal state. Memory checks the received event with agent goals and plans; if it detects any accomplishment or failure, emotion rules will be applied. This separation of data and process imposes a technical difficulty for processing back and forth between the emotion and Memory modules. However, it makes our system flexible for using any emotion model in the emotion module.

Emotion from an event will be applied to emotion reactive rules. For example, if the agent fails to achieve a long term goal it may burst into tears. It is important to note that the agent does not plan to execute impulsive actions (arrow 3) whereas the emotion rules dictate impulsive action execution upon the appraised emotion. The final step in the emotion module is to integrate the result of appraised emotion into the agent Emotional State.

Emotion expression, impulsiveness, and how the agent copes with an appraised emotion could be determined in our model in one processing cycle by applying emotion rules and consulting with active roles as well as Memory. The Emotion Traits threshold, emotion rules in response to a received event and integration of these appraisals to update the agent Emotional State in each processing cycle will be affected by active roles (arrow 2). This will be described further in Section 3.3 and 3.5.

- Memory: The event and appraised emotion will be stored in a Memory Cell. The Memory updates facts which have been changed. The agent TOM module processes the event, and updates agents' profile that were involved in the event based on actions tags; this will be discussed further in Section 3.4.1.4. The Memory module is also responsible to keep the track of the agent planning and validating the next action in the sequence of events by checking its preconditions with the current world state. If the Memory validates an action, it will be executed by sending the

confirmation to the planning module, otherwise the action is invalid, and so the current stored plan and the agent should make a new plan (arrow 5). If the current plan has reached the final state, the Memory informs the intention recognition to set a new goal for the agent (arrow 4). The record of achievement or failure of the plan should be stored in the agent active roles (arrow 6).

- Intention Recognition: Intention recognition processes the agent active roles (arrow 7) and sets the agent intention on the most important goal. The goal importance and the intention process to select the most important will be thoroughly explained in Section 3.5.1 and 3.6. This goal will be passed to the planning module.
- Planning: The Planning module receives the agent goal and the current world state (including the agent active roles, arrow 8) comes up with a plan that takes the agent from the initial state to the goal state. This plan will also be stored in the Memory (arrow 5). If the planning module receives the confirmation from Memory that the current plan is still valid, it will execute the next action in the plan action sequence (arrow 8).

In this section we briefly discussed the agent high level process and how all modules work in collaboration with each other and the agent active roles. In the following section we briefly discuss preliminary terms in our design and then move on to discuss each component thoroughly.

3.2 Preliminaries

Our main system consists of three main sub-systems: Authoring System, Event Handler and Agent (Figure 3.2). The Authoring System enables the designer to create the environment including facts, actions as well as agents. The designer can create his own agent with given roles. The Agent role formalizes its deliberative and reactive behavior in the environment. The Agent demonstrates the design's innovative features to simulate a believable behavior based on its authored roles and traits. Finally the Event Handler's main task is to collect actions from all agents in the world and output the final result based on the designer's given scenario. At runtime, the Event Handler receives actions and distributes the result. In this Section we give a brief description of our main preliminary terms.

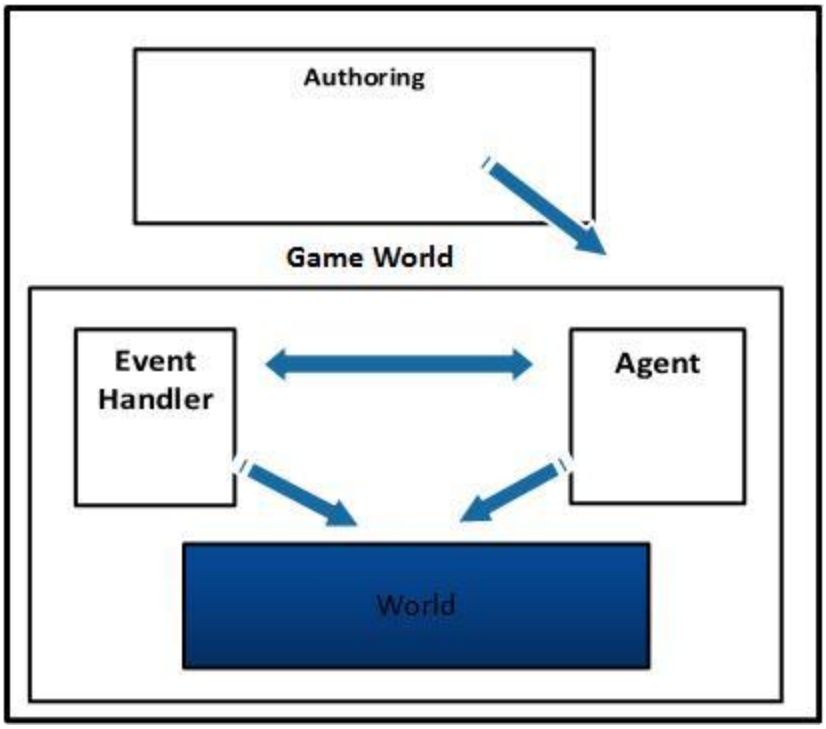

Figure 3.2: System main components

3.2.1 Fact

The fact representation is similar to first order logic in that it provides an evaluation of the truth or falsity of facts, but differs in that numerical values can be assigned to a fact as well. Each fact has a name, takes a predefined number of literals, and outputs either a number or true/false value. Facts can also have a target. For example, the fact Happy (Sue) targets Sue. A numeric value enables a quantitative comparison between facts from the same types. Quantitative comparison between facts greatly enhances state utility calculation for planning and decision making where the agent will be able to evaluate different world states and choose the one with the highest utility. A world state is a collection of facts with their environmental variables (Figure 3.3).

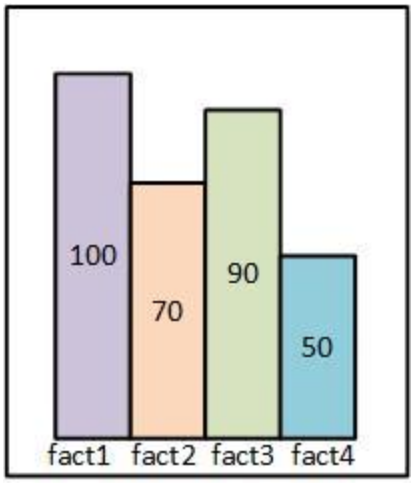

Figure 3.3: Schematic world state with its facts.

3.2.2 Action

An Action is a single atomic step that has been identified with a name. Each action has a set of preconditions, and post conditions with their associated probability (Figure 3.4). Like world state presentation, post conditions and preconditions are represented by conjunctions of facts. As long as preconditions are satisfied, the action is valid to be executed. Upon the action execution, only one post condition will affect the world state at run time. The post condition of Sue Attacking Kathy could be Kathy getting injured or possibly dying but only one of them will become true. The agent is informed about the probability of each post condition while the actual post condition after the action execution will be determined by the Event Handler.

In the action structure, there are certain types of preconditions that the agent could not plan to make true. More precisely, the action can only be chosen if the agent's current state satisfies the precondition. For example, in the case of Sue attacking Kathy, a precondition could be that Sue's affinity towards Kathy is below a certain threshold. If this precondition has been satisfied by the current relationship between Sue and Kathy, attack is a valid action. Otherwise, Sue will not plan to first decrease the affinity level in order to attack her later on in the scenario. However, for other preconditions, the agent could plan to make them happen; for example, Sue will enter the room that Kathy is inside to attack her.

In order to remove the need for an ontology system, we chose to use a tagging system for actions. There are two types of tags that will be used to annotate the action. The first group is emotion appraisal variables such as desirability, desirability for other, praiseworthiness and etc. from the OCC model. The second is personality trait variables. Emotion appraisal variables help the agent appraise the emotion of a received action based on emotional rules. The second type of tags will be used in the agent personality traits and completing TOM profiles will be discussed in Section 3.5.2.2 and 3.4.1.4. The main premise behind personality tags is that the agent has a greater tendency to choose actions with traits that closely correlate to their own personality, and when other agents perform actions, the agent can interpret other agents' personalities by means of these tags.

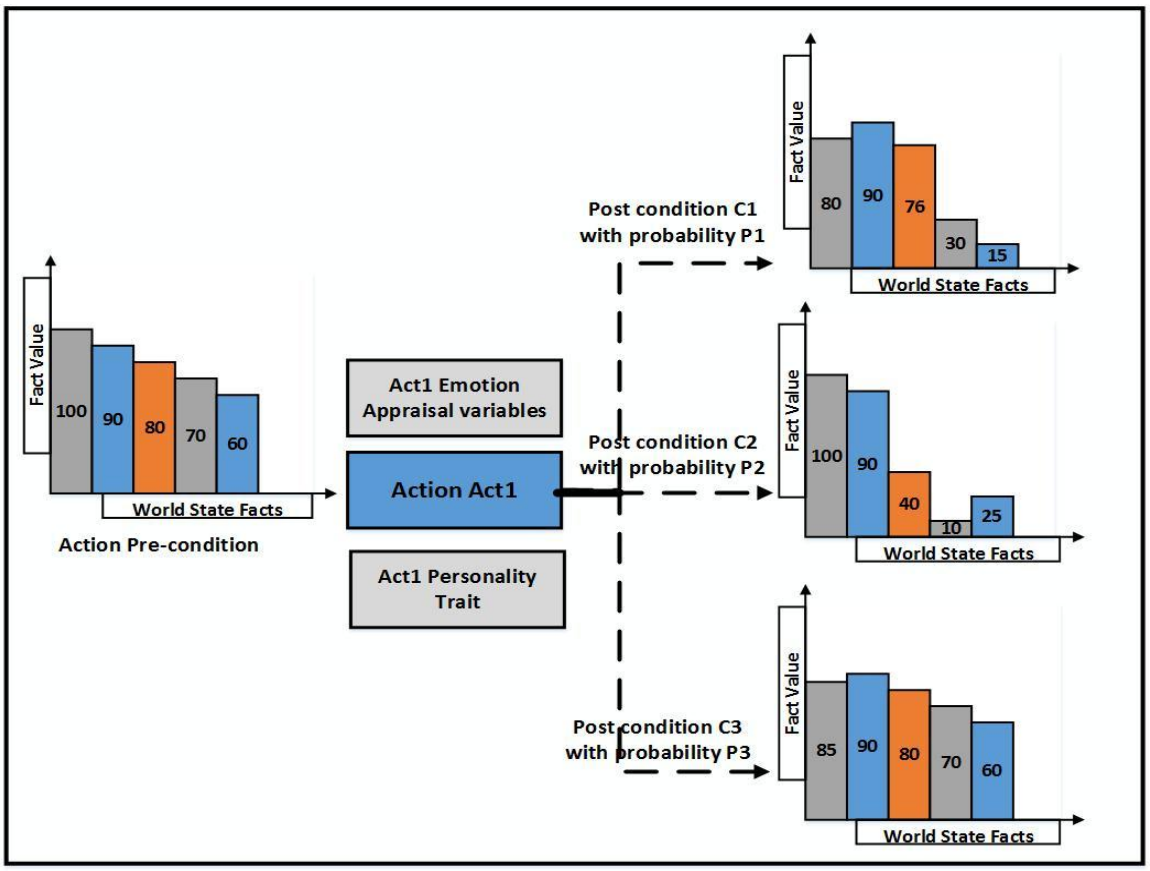

Figure 3.4: Action with a precondition and three post conditions

Action personality tags may come from a classic personality model trait like OCEAN. The designer can create his own meaningful tags in the context of the story. For example, the designer can generate a tag like extroverted and stipulate that social activities require a higher degree of the extroversion trait whereas individual activities demand a lower level. Tagging a group of actions with the extroversion trait makes it necessary to use it as a personality trait in some of the agent's roles.

3.2.3 Event Handler

The agent passes its selected action to the Event Handler, which then determines the action's post condition (Figure 3.5). The Event Handler applies the necessary changes in the world state based on the post condition and distributes the action, along with a corresponding new world state, to other agents through message passing as an event. For example, if Sue chooses to slap Kathy, she passes this action to the Event Handler. The Event Handler then distributes "Kathy was being slapped by Sue" to other agents. How this action affects each individual agent's state or their social relationships will not be included in the message.

More interestingly, the Event Handler has the potential to manage the story at run time based on the designer's previously written plot. When the agent executes an action that has more than one post condition, the Event Handler determines which one will affect the world state. It manages the story at run time by collecting actions and determining results according to the scenario or world rules. For example, the action of shooting a rabbit has two post conditions: the rabbit may get killed with a probability of 30 percent or it may run away with a probability of 70 percent. If the agent shoots a rabbit, the Event Handler receives this action. If shooting killed the rabbit, the Event Handler has to modify the world state according to that. Note that all events should not be scripted in the Event Handler by the designer; instead they should be determined dynamically based on the world's state.

In conclusion, the Event Handler collects agents' action and distributes the result. It keeps the world state consistent by modifying facts upon newly-received actions. Finally, it notifies the agent about new settings in the global environment. For example, if the

agent walks in to a room, the Event Handler is responsible for describing the room to the agent, including who and what takes place in it.

3.2.4 Message Passing

Messages inform the agent about the world's current state, or of events that have taken place. When the agent chooses an action, it should be passed to the Event Handler, upon which the Event Handler distributes this action as an event inside of a message to other agents in the world. The message will be processed by an agent perception module.

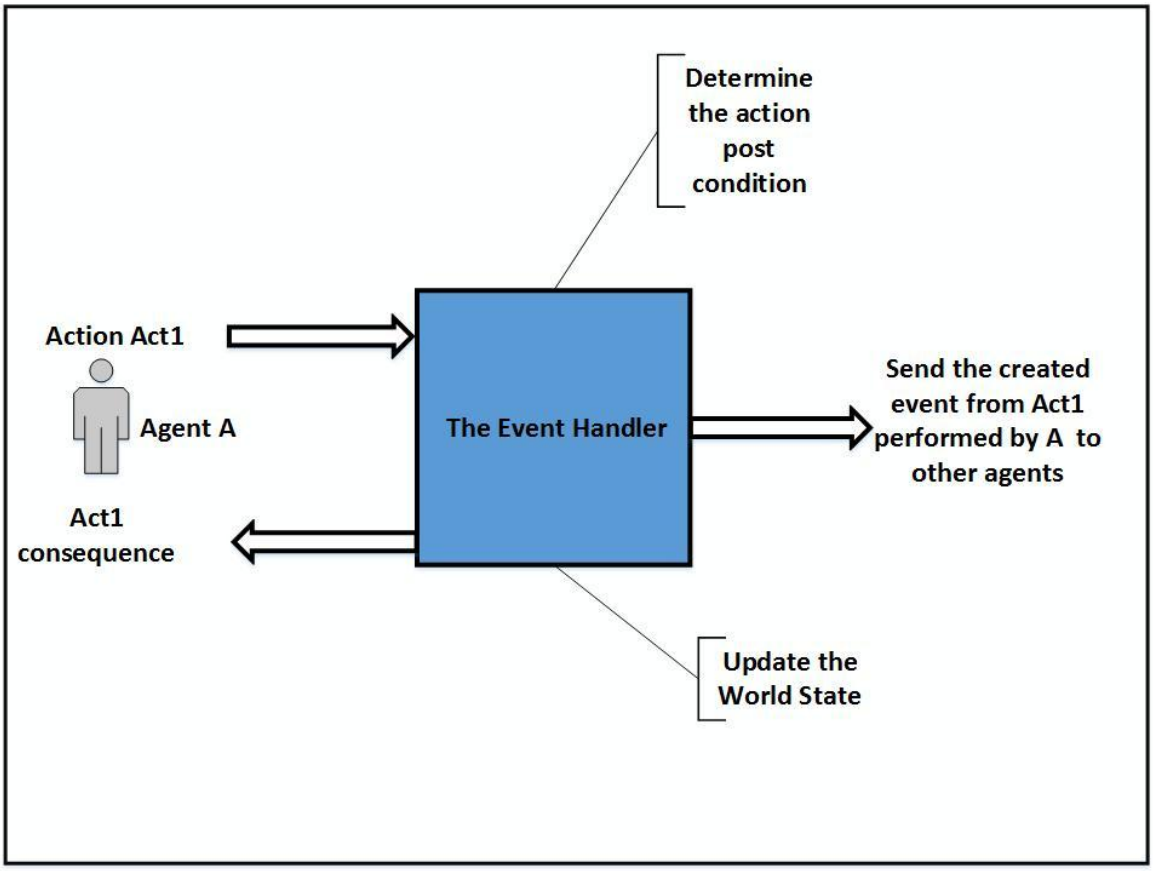

Figure 3.5: The Event Handler functionality in the runtime

3.2.5 The Authoring System

Authoring System is a design tool to create the environment as well as agents. The agent will be defined by at least one role and its goal, personality trait, and TOM profiles. The designer can use the Authoring Tool's functions to modify different characters in his or

her own proposed scenario or story. The Authoring Tool facilitates implementation of various traits and roles that generate emergent behaviors at runtime. The Authoring Tool can also provide users with template roles. These template roles can be customized to define different characters by means of different attributes in the same social context that has been define by the role.

3.3 Emotion

Loyall has defined emotions as one of the first requirements for believability. Presence of emotion reaction, effect of emotion state on decision making, and their role in regulating social relationships are a requirement specification of the emotion module in our proposed model. In everyday language emotional state (mood) and emotion are used interchangeably; however many scholars believe that it's necessary to distinguish them from one another. Gebhard discusses that emotional state is not generally related to one event, action or an object, whereas emotion can be associated with only one event [50]. Gebhard describes emotional state as an affective state which has a great impact on human's cognition functions such as decision making, motivation, and appraisal.

In order to take this difference in to account, our model proposed two components for managing agent emotion data: a) Emotion Traits, and b) Emotional State Dimensions. Emotion Trait is output from immediate event appraisal by emotion rules, for example love, hate, anger, gratitude. The agent Emotional State is an aggregation value of appraised Emotion Traits.

To follow a data driven approach and create a more robust and reusable architecture we decided to separate emotion data representation (Emotion Traits and Emotional State Dimensions) as discussed below from emotion rules. As it has been discussed earlier in Chapter 2 there has not been any universal agreement on one acceptable emotion model among scholars. Separation of data and process for emotion subsystem enables the designer to place his own desirable emotion model.

Emotion rules dictate how the agent should react to an external stimulus. They should be applied in three consecutive phases; first the event will be appraised based on appraisal

rules. The result of this process is an Emotion Trait like anger or gratitude. Second the appraised trait will be checked for any reactive action. Finally the Emotion Trait updates the agent Emotional State.

3.3.1 Emotion Trait

The Emotion Trait is the immediate emotion appraised from an event by applying emotion rules. For example, accomplishing a goal makes the agent happy and losing a wallet makes him upset. An example of Emotion Trait in an OCC model is: Pride, Shame, Love and etc.

The Emotion Trait structure should minimally have the following variables:

- Name: Each Emotion Trait has a name such as: Pride, Anger.
- Range: Each Emotion Trait has a valid range which defines minimum and maximum value that it can be assigned to it.
- Threshold: Threshold determines the degree that this Emotion Trait needs to achieve to be felt or triggered an action tendency. An Appraised Emotion Trait with the value below the Threshold will be discarded. The Threshold of the same Emotion Trait for one agent is the same.
- Value: Value should be within the valid range that will be determined by applying emotion rules.
- Emotional State Dimension Tag: Each emotion has one or more Emotional State Dimension Tags that identify which Emotional State Dimension it belongs to. The mechanism of this mapping can follow any emotional state representation model.

An Emotion Trait can be used to represent the agent emotional reaction to what is happening in the world. The designer can assign the same values to different agents' trait variables (Emotional State Dimension Tags and Threshold). However, our model has the potential to override trait variables in the personality section of the agent and therefore create distinguishable behavior. For example, by assigning a low threshold to an agent's anger trait, the agent would demonstrate anger more easily.

3.3.2 Emotional State

Emotional State plays an important role in decision making and action selection. Fundamentally it is not each individual Emotion Trait based on appraised events that affects the decision making process, but their aggregation that plays a leading part in action selection. Note that the effect of emotion in the agent decision making based on the agent Emotional State is different from emotional reaction apprised from a received event. Emotional reaction will be applied simply by applying emotion rules whereas Emotional State affects the process of decision making. For example, the agent who has been frustrated for a while may deliberatively make a plan for suicide bombing. In our system each action can have a set of non-procedural preconditions; Emotional State is one of the main leading factors to validate these non-procedural preconditions. For example, in order to initiate an attack, the agent should be very desperate or angry, as long as the agent Emotional State is close enough to this state, attacking is a valid action. However, if the current Emotional State does not satisfy this precondition, the agent could not plan to make it happen.

The Emotional State data model can have several dimensions. Each dimension minimally should include:

- Name: Each dimension will be identified by a name.
- Range: Each dimension has a valid range.
- Value: It shows the accumulated value of all Emotion Traits that has been added up to each specific dimension based on their Emotional State Dimension Tags until now.
- Decay Rate: Each dimension has a decay rate.

Emotional State dimensions and process of accumulating all appraised emotions is a computationally desirable approach to manage the agent Emotional State. Once the trait has been defined it will be added up to its associated dimension. One problem with using a model like OCC with 22 traits is a heavy computation burden to maintain value of each trait over time and update them.

3.3.3 Emotional Rules

Emotion rules define how the agent should react to a received event. Genuinely they define how the agent appraises an emotion towards a particular event or should possibly react based on the appraised emotion. Here is the list of minimum functionality that emotion rules need to satisfy:

- Emotion Reaction: Emotion Reaction rules are responsible to appraise immediate emotion responses to received events. In our model we chose to follow the OCC approach model by using tags such as desirability, desirability for other, praiseworthiness, etc.
- Updating Interpersonal Relations: The agent interpersonal relationship affinity level is updated based on appraised emotion.
- Action Tendency: Action tendencies refer to a group of actions that are agent impulses toward an external event. Action tendency is placed in the emotion section because they are a direct result of the agent emotion response to an external event. Action tendencies are If-then-else rules that can be applied once the appraised emotion has passed its threshold, for example crying when one is extremely sad or punching the attacker right away.
- Goal based: At each update if the goal state has been reached the agent may generate satisfaction/fear-confirmed or relief /disappointment. Goal based rules need an interface to the agent role to be informed of all goals and their last stored state in the agent.
- Updating Emotional State: Apprised emotions from a perceived event update the agent Emotional State.

The main reason that we distinguish between different groups of emotion rules is their functionality and their interface with other agent's modules. Although emotion rules are the same, the architecture provides the flexibility to customize emotion responses through personality traits in the role architecture. This customization brings diversity to our agents' emotion expression as well as making each individual behavior situated in its social context.

3.4 Memory

Memory and, in particular, the ability to remember previous experiences have not been explicitly mentioned in the literature as a requirement for a believable agent, but a socially-aware agent without memory does not make much sense. Many believability requirements from Loyall such as: social relationship and consistency of experience would need a memory as well. An agent should be able to remember and preferably learn from past experiences, not repeating the same mistakes over and over again, which easily breaks the audience's immersion. The TOM module in Memory works as a learning engine which interprets received events based on predefined traits to complete other actors' profiles.

Once the agent has experienced a particular event it should be able to remember that event with its associated emotional affect. If this emotional effect is positive, the agent may develop a tendency to pick the same option in future. In the case of a negative emotional effect, the agent avoids the situation or picks a different option. For example, if the agent is looking for a social activity and one option is to spend time with a group of people who have bullied the agent before, and the other option is to go to a movie with a friend, hanging out with bullies while the agent has not enjoyed their company in the past is not an optimal choice.

One of the main functions of Memory module is to help the agent remember what has happened, and adding learning to this feature increases the agent's knowledge based on world dynamics. It is necessary for a socially-believable agent to develop causal reasoning for others' actions and current events. For example, when the agent needs to maximize its utility based on interaction with others, trusting an unreliable agent several times illustrates an inability to learn about that agent's personality.

Fundamentally learning from past experiences enables the agent to predict the future, which consequently rescues him from repeating the same mistakes, particularly during social interaction with others. An agent who observes the same pattern of behavior from the same agent a few times should, eventually, be able to pick up the associated personality trait present in the observed agent. For example, if Bob observes violent

actions from Sue previously, he develops knowledge that, in general, she has a tendency towards performing violent actions.

TOM is not exclusive in only storing other agents' personality traits; it also enhances an agent's strategies over time. A utility-based approach, hand-in-hand with learning, makes the agent a more proficient planner due to the fact that he is not blindly looking for maximizing his utility but also taking into account uncertainty and other agents' actions. For example, if an agent's goal is to earn as much money as possible, more money means higher utility. When the only way to gain profit is trading their goods with other actors in the environment, Memory enables the agent to find out what are the most profitable trades, as well as the most profitable trading partner. The task of finding the best trader cannot be done unless the agent could remember previous interaction with others.

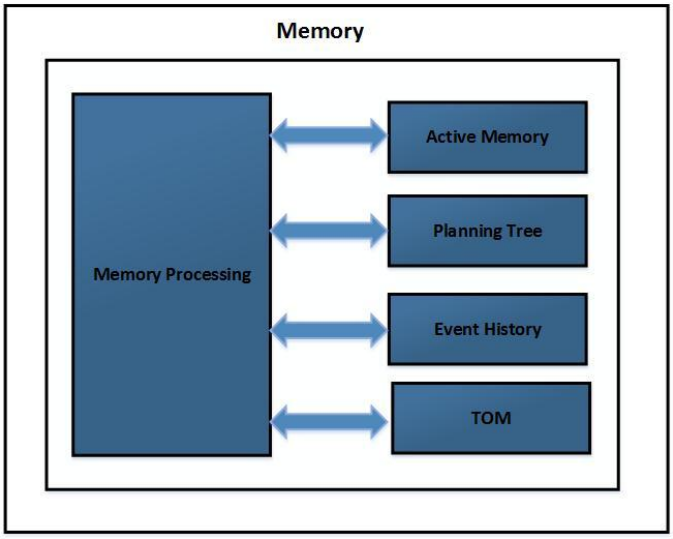

Figure 3.6: Overview of Memory module.

Our Memory functionalities can be categorized into three main groups:

- Current World State: Active Memory consists of all facts that the agent knows about the current state of the world. Memory is responsible to update these facts upon perceiving new events. Keeping track of the current world state is necessary for the agent planning process and action selection and also validating the planning tree that is now in the Memory.

- History of Events: The main and very basic functionality of the Memory component is to record all observed events in Memory Cells associated with their emotional affect (if they have one).
- Theory of Mind: The main premise of theory of mind is that how we act depends not only on the immediate effect of our action but also on how we believe others will react [13]. In the same article, Sally [13] describes different reasoning levels, suggesting that a 0^{th}-level reasoning is exemplified when an individual only considers their immediate desires, beliefs and traits, while a 1^{st}-level reasoner expects others to act with regard to their own desires and beliefs, and consider others in their planning.

  The concept of theory of mind (TOM) has a long history in psychology. However, in the field of believable agents there are very few models which use this powerful concept to enhance agent planning. FATiMa, with its double appraisal process and Psychsim, by introducing the appraisal theory of mind, are among these models.

  Pure decision theory approaches are conventional methods for decision making in multi-agent systems. Their biggest drawback is that they do not take into account preferences of others. Humans, in their day-to-day decision, naturally make use of each other's' mental models. For example, in a competition, or while compromising on a deal, dismissing the fact that other parties involved have their own set of priorities leaves the decision maker with a semi-optimal outcome. TOM facilitates the prediction of this factor and implicates it in agent decision-making equations.

  TOM enables the agent to recognize other's patterns of behavior. To implement this feature, one option is to provide the agent with exact mental models of other agents, such as emotional state, roles, and associated personality traits. Although the mentioned method removes the computational burden, it does not have any parallel case in humans. We chose a more realistic approach by enabling the agent to recognize other agents' patterns of behavior and to make predictions in a trial-and-error process with their incomplete model.

  The TOM component is a critical feature in the agent planning process, since it not only determines the desirability of states or actions from other agents' points of

view, but also uses these elements to deal with the uncertainty of other's action during planning.

In order to perform all of the above general functionalities, Memory consists of two main parts. The first one is a data structure to store events and other agents' profiles, and the second part is a processing unit which manipulates these data structures and coordinates them with other modules such as role and planning.

3.4.1 Memory Structure

Memory structure includes four different components that store received data: Active Memory, Planning Tree, Event History, and TOM as illustrated in Figure 3.6.

3.4.1.1 Active Memory

Active Memory consists of all of the facts, and their associated values, that the agent knows about the current state of the world. The agent perception mechanism, by definition, is open world assumption or OWA, meaning that those facts which the agent does not know will have their values considered unknown rather than false. The world and value of each fact is independent of the observer, and these values become known as the agent discovers the world through exploration.

The agent knows as much about the world as it has seen or been told through received messages. Interestingly, a group of agents can have different, or even contradicting, facts in their Active Memory, as illustrated in Figure 3.7. In each update, the agent saves the most recent state in its Active Memory and removes facts with differing values following each update. Furthermore, the agent does not have access to other agents' active memories; the only way that one agent's knowledge can be shared is through the agent voluntarily distributing them.

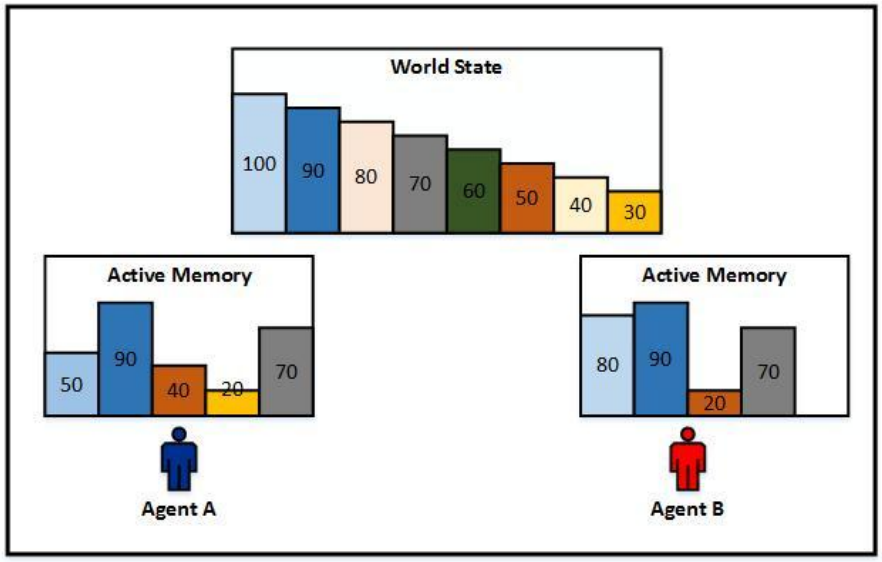

Figure 3.7: Two agents' active memories can have a different value for facts.

3.4.1.2 Planning Tree

If an agent decides to achieve a goal, the agent will generate a plan that is a sequence of actions to achieve it. The planning process will be discussed thoroughly in Section 3.7. The result of this process is a sequence of actions that will be kept in the Memory. Each time the Active Memory is updated, the next action in the sequence will be checked for validity. The action is valid if all of its preconditions are satisfied. If a received event makes the stored plan invalid, the agent needs to consider re-planning.

3.4.1.3 Event History

Event History is an array of Memory Cells, each of which has two separate components: Event and Emotional Effect as illustrated in Figure 3.8. Each Memory Cell can be linked to a profile in TOM, or one of the agent roles. The mechanism of this connection will be discussed shortly.

After the perception unit filters relevant events and passes them to Memory, the event needs to fulfill at least one of the criteria below for it to be stored in Memory:

1. Emotionally Significant: In order to be emotionally significant the event should be related to either the agent, or one of its roles, targets, or goals. Furthermore it

should appraise an emotion in the agent. If the emotional impact of an event makes the agent regulate its relationship with others, i.e. causes any change in the agent's affinity towards them, a Memory Cell will be linked to that role as discussed in Section 3.3.

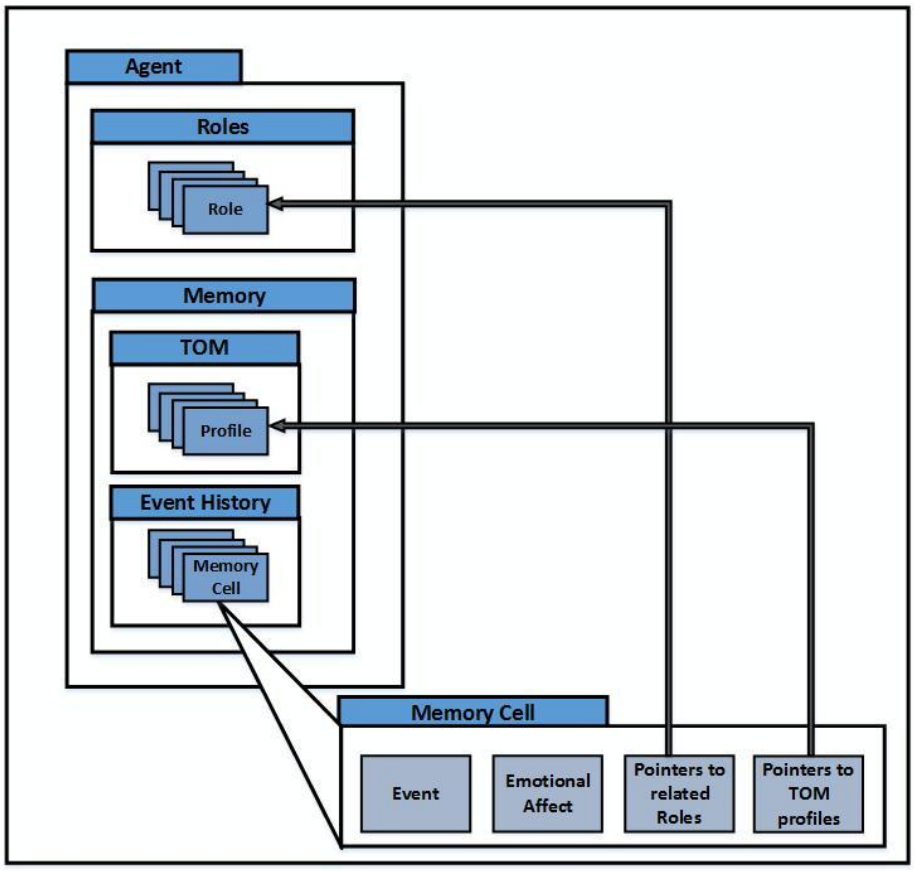

Figure 3.8: Memory Cell components in the Event History

2. Informative: An event is informative if it adds new information about other agents' states or confirms a previously-observed trait. For example, a total stranger who walks past the agent on a side walk will not occupy the agent's attention for more than few seconds. However, if the agent were to be witness to a bank robbery or a violent act, their attention will be occupied for much longer presumably. What should be counted as an informative event, then, depends on the scenario, belief system, and what aspects of other agents' personality are

important enough to be saved in Memory. For example, the designer can define a thief by following set of rules:

Grabbing(X, Y) = True

Own (Z, X) =True

Z! =Y

=>

Thief (Y) =True

By having the above rules in the belief module, if the agent cares about recognizing a thief, and it receives an event which confirms a robbery, that event will be saved in the Memory. Each Memory Cell may have a pointer to one or more profiles in TOM which helps the agent remember what made them update a trait value in a profile. This mechanism enables the agent to revise their Memory upon receiving new facts. Bob may tag Sue as a thief while she is driving a red Ferrari but later, if he realizes that it was her father's car which she had borrowed, he may modify his previous judgment of her.

1. Request for Re-Planning: If an event forces the agent to update or change their plan, the event will be saved in a Memory Cell. This category may or may not generate emotion in the agent.

3.4.1.4 TOM

Theory of mind (TOM) enables an agent to take the preferences and personalities of other actors into account in their decision-making and interaction. TOM consists of an array of profiles that formalize an agent's knowledge about other actors. There are two approaches to provide the agent with the mental model of other agents as illustrated in Figure 3.9: a) the designer authors profiles; or b) the agent updates profiles according to received actions. Both approaches result in taking other agents' profiles into account. Updating profiles is more realistic; however, it is a process of trial and error.

Additionally, it demands that the agents become engaged in many interactions for its final model to become similar to the other agents' personalities. On the other hand, authoring is a fast and straightforward method that is necessary when agents need to know each other in advance (for example for story reasons) but there has been no prior game execution to permit the generation or updating of TOM within the game, in such a case pre-authoring is the only way for the agents to have the proper TOM. Authoring enables the designers to take control over the story as they can specify exactly what one agent thinks of another.

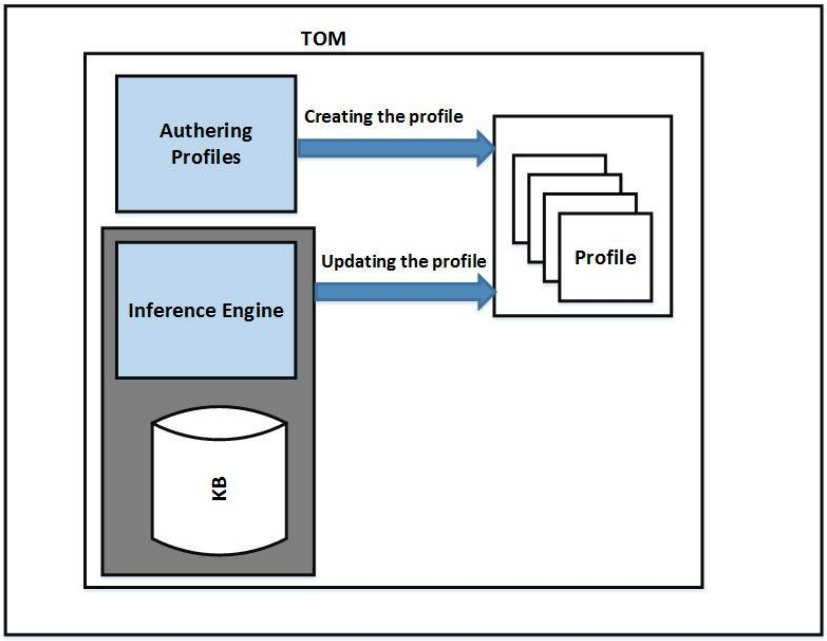

Figure 3.9: TOM module with two options for pre-authoring and learning

Ideally, an agent needs a knowledge base and some type of ontology system to interpret other agents' personalities based on actions. Such interpretation is used to predict future situations. However, using an ontology system puts a heavy burden on the system's performance. This burden encourages us to use action tagging that has been described earlier in Section 3.2.2 for the recognition of other agents' personalities.

Action tags can be as simple as 'good/bad' or 'moral/immoral', for example. In our architecture, the type and variety of tags completely depends on the designer and what they are looking for from an agent's personality according to a scenario. For example, in

simulating bullying in school, slapping someone can be tagged with ‘violent’, ‘bully’, and ‘bad temper’. The observer of an action uses these tags to update the profiles of actors involved in the action. For example, if Kathy slapped Sue and Joe saw it, he will change Kathy’s profile based on slap tags. Our model is flexible in a way that it can be used to implement any personality model such as OCEAN, as long as the designer provides a consistent mapping from traits to actions. In a given state with a set of available actions, TOM finds their priority based on the actor’s profile and actions’ tags.

3.4.2 Memory Functions

Memory functions are responsible for creating an interface between Memory modules and other agent components as it has been illustrated in Figure 3.10. Upon receiving an event from the emotion module, these functions filter the event independently and make the necessary changes by coordinating with other agent modules. This is illustrated in Figure 3.1. Note that all received events from the perception module will be passed to both the emotion and Memory modules.

- Updating Active Memory: Upon receiving a message, it will be checked for consistency with Active Memory and values will get updated. These updates may cause the agent to revise their plans.
- Re-Planning: If the new world state makes the agent’s previously generated plans invalid, the current world state will be sent to the agent planning section.
- Updating Event History: If the event appraises an emotion, it will be recorded in a Memory Cell. This process works as a filter. It checks the message and, as soon as it satisfies one of the mentioned conditions, it is qualified. If the event is emotionally-significant, the generated emotion will be saved as well as the event.
- Updating Profiles: Action tags help the agent to update the actor TOM profile. It is important to remember that not all actions have a tag. If the action has a tag then the agent will update a performer profile. Walking is not a significant action; however, stabbing someone is significant in identifying the other person’s personality. More interestingly, the agent profile is not necessarily accurate, since the agent only saves their own interpretation of the other actor’s action(s).

- Calculate Action Probability: The agent can pass a group of available actions and a potential actor to the TOM module in order to calculate the probability of their successful execution of those actions based on their own and the other actor's profile.

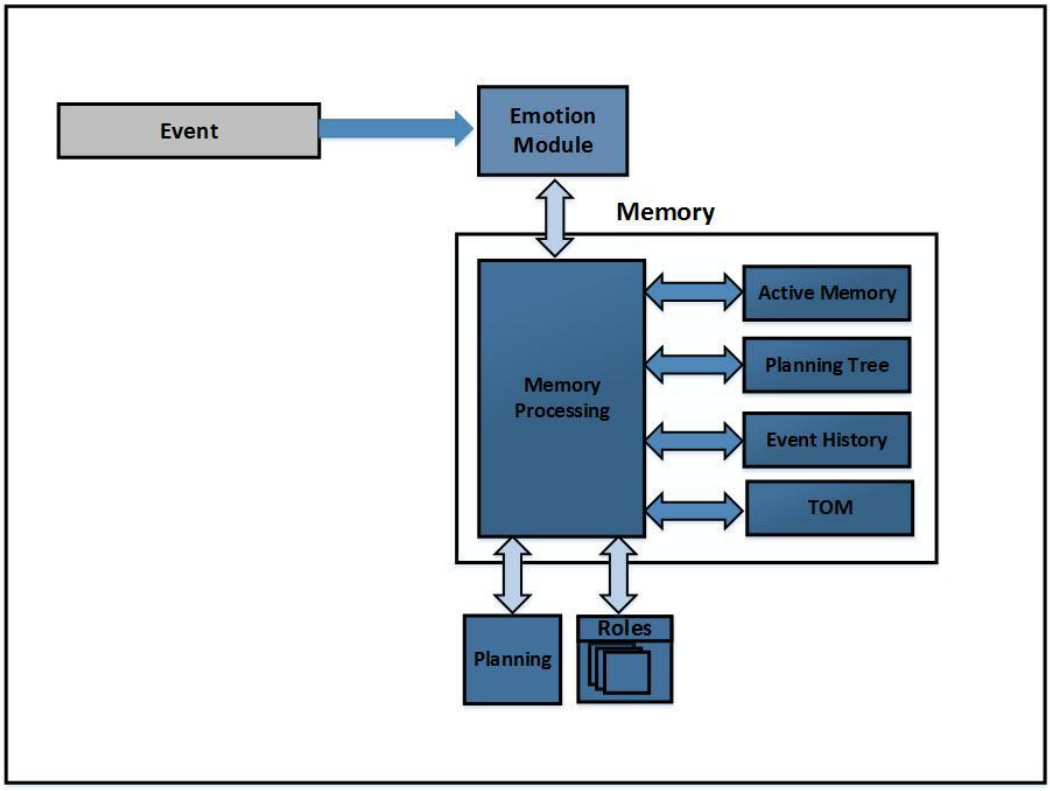

Figure 3.10: General overview of the Memory

3.5 Role

Role theory defines a role as the relationship of one person to another person, group or object. It formalizes an agent's relationship with its environment and with other agents, including other agents' expectations based on this relationship. Acton identifies a role as a concept that can channel this formalization through belief, desire, and intention [56]. You defines a role as a coherent set of standard behavior, actions, norms, values, and goals in his model [57].

Roles store agent information that is bound to social context in which the role *target* is situated. There is one exception to this, namely, *default roles* that have no target and are

always active; this will be discussed in detail below. The role structure defines the relationship of an agent with the role target. This formalization defines the role's influence on agent behavior when the role is active. The complete structure of roles is summarized in Figure 3.11.

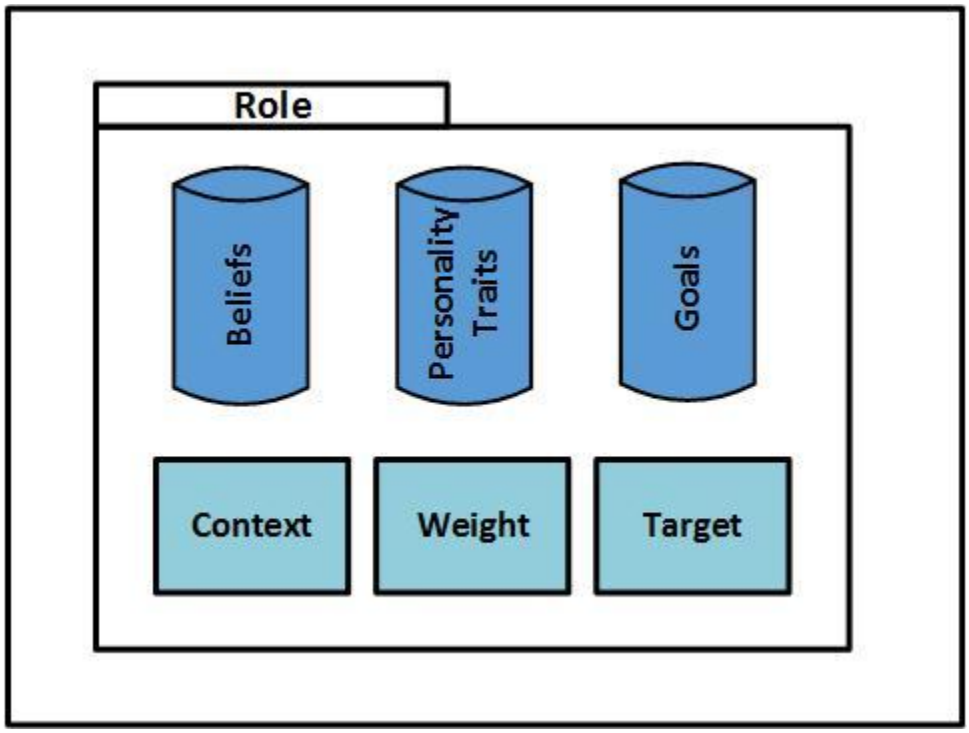

Figure 3.11: Role's components

- **Name**: Each role has a unique name, such as mother, friend, secretary, or boss.
- **Personality Traits**: Each role stores zero or more personality traits. This allows active roles to contribute to the expression of agent personality traits. A default role contains the core personality traits.
- **Target**: Roles may store a particular target, but this is not necessary (i.e. target-free roles can be defined; default roles are an example of this). Role targets may be an agent, a group of agents or an object. For example, a *mother role* may target the mother's children, or a *teacher role* may target the teacher's students.
- **Context**: In addition to a target, roles may define a context—or series of facts—that cause the role to be activated. For example, a *secretary role* may have no target, but may be activated only when the agent is at its workplace

- **Weight**: Each role has a weight that identifies its importance. The degree of influence of each role when multiple roles are active is determined by their relative weights.
- **Beliefs**: Roles may contain beliefs that influence the agent's appraisal of events, the world state, and action tags when the role is active. Our system has two belief components: Reward System and World View. Reward System identifies desirability of events and facts; World View provides a high level categorization for actions.
- **Goals**: Roles may store zero or more goals that are activated in the agent model when the role becomes active. This allows roles to activate richer behavior than simple reactions to the environment.

Our design dictates that all agents in the environment possess one or more roles, including—minimally—a default role that defines the agent's relationship to the environment. The default role characterizes default agent behavior when no relevant social context is influencing such behavior. In this way, the default role models context-free elements of the agent's beliefs, personality, and goals. For example, if Bob is *usually* a greedy and impatient person (though these qualities may change in particular social contexts) then greediness and impatience are formalized in Bob's default role.

All non-default roles store a target, a context, or both. These elements define the conditions under which the role should be activated, in the presence of the target, under the conditions defined by the context, or a combination of both. For example, suppose Bob is friends with Jake. Bob stores a *friend role* which contains the identity of Jake, as its target; this *friend role* is activated during Bob's planning process if the plan involves Jake. The *friend role* also contains beliefs, personality traits, and goals associated with Bob's perspective of his friendship with Jake. In this way, non-default roles model a sort of context-bound adjustment of how beliefs are activated, personality traits expressed, and goals pursued in particular social contexts.

Our role-based architecture may assign multiple roles to the same agent as illustrated in Figure 3.12; for example, Bob may be friends with Jake, he may be Sue's husband, and

Kathy's boss. More interestingly, more than one of Bob's roles can be active at the same time; each active role stores its weight, which denotes its importance (and therefore influence) in Bob's planning process. Naturally, this may activate conflicting goals and beliefs from different roles; such a case models Bob's internal conflict in difficult social situations. For example, Bob's default role may strongly forbid killing another human being, but when a *soldier role* is activated he may be compelled to kill his enemies. During a battle, Bob experiences a conflict between his normative moral code and his duty as a soldier as the two roles compete to determine his propensity for lethal violence.

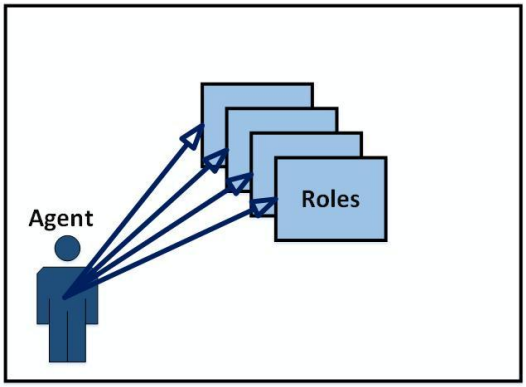

Figure 3.12: The agent can have multiple roles.

Our Role-based architecture has several advantages:

- **Reusability**: Once a role has been defined it can be reused for different agents. For example, a *guardian role* can be defined for all mothers and father relationships in a game environment. Furthermore, basic design patterns such as class-based or prototype-based inheritance can be applied to generate specializations of general roles, such as a *mother role* and *father role* derived from the basic principles of a *guardian role*.
- **Customizability**: Designers are at liberty to customize: (a) the structure of relationships that activate roles, (b) the structure of traits and goals stored within roles, and (c) the prioritization of various roles and various traits within each role.
- **Understandability**: The concept of roles based on social relationships can be easily understood by non-experts; it is intuitive. One of the main problems with current

agent architectures is that they employ complicated, exotic modeling patterns that are difficult for non-experts to understand.

- **Flexibility**: Allowing agents to have several roles activated in different contexts offers flexibility. For example, Sue can be a mother, teacher and wife, and only act according to those roles when they are relevant to her social context.
- **Ease of social knowledge representation**: Our architecture provides an easy solution to import social knowledge to NPCs. The ability to define the individualized reward system, norm, and value in a role empowers agents to demonstrate social intelligence without using a complex ontology. For example, Henry might frown upon cheating on an exam. As long as he can recognize what "cheating" is in an exam context, he can avoid it.
- **Suitability for dramatic narratives**: The ability to mix and match roles, and customize them makes our role-based architecture a perfect choice to create interactive drama.

3.5.1 Goal

Goal is a desirable state of world that consists of one or a conjunction of several facts. OCC model identifies three types of goals: Active, Replenishment and Interest. Inspired by OCC categorization, in our model all goals that can be accomplished by planning are in the class of Replenishment goals; one Replenishment goal that the agent is currently planning to accomplish is the active goal. However the agent can not directly take an action to achieve Interest goals; they are usually presented by a static fact with importance values. An example would be if Bob is fan of a football team, his team winning the championship makes him happy but he could not do anything as a fan to influence the match result.

The reason that an agent does not actively pursue an interest goal is either because a) the agent does not have any control over their state, for example being betrayed by another person in prisoner's dilemma, or b) the agent role enforces a particular preference for one world state over another, which is a factor that the agent needs to consider in the planning. For example, seeing the happiness of a friend can be defined as an interest goal,

which causes the agent to avoid a sequence of actions or have less utility for states that would make this goal false. However one goal that has been defined as an interest for one can be defined as Replenishment for the other.

3.5.1.1 Replenishment

In our architecture we define two classes of Replenishment goals: a) Story Concern b) desire based. The nature of both categories is the same in a sense that they both produce goal directed behavior. Story Concern goals drive the plot forward by compelling an agent to achieve a specific world state. Desire based goals prevent agents from becoming idle if they have no active goal in any of their currently active roles. Generally desire based goals have a lower priority in comparison to Story Concern goals, which means if there is an unsatisfied Story Concern goal the agent will not start planning to achieve a desire based goal. The mechanism of this selection will be discussed further in the intention recognition module.

Story Concern goals explicitly can be defined by the designer to create dilemma or a specific scenario: for example in prisoner's dilemma, Alex has the goal to spend fewer years in prison. Once the agent completes a sequence of actions in a plan to achieve a Story Concern goal, it will be discarded. On the other hand, desire based goals can be used to represent repetitive but goal directed behavior; for example the agent gets gradually hungry a few hours after its last portion of meal.

Dependent on the designer's choice, Story Concern goals can be used as a mechanism to dynamically determine the story path. The designer can provide multiple alternative scenarios for different conditions of a Story Concern goal state and then, at runtime, the goal final state causes the Event Handler to apply one of the alternative scenarios. This alternative scenario can be applied by assigning a new role to the agent. For example, if the agent Story Concern goal of getting married is accomplished successfully, a spouse role will be assigned to the agent. Dynamic role assignment facilitates our model to be integrated with a non-deterministic story narration, consequently makes our agent model more efficient to be used in an interactive drama.

a) Story Concern

Story Concern goals mainly can be used as a mechanism to lead the agent through the story on the plot that the designer wants to simulate. The agent does not develop a desire or an urge to achieve these goals; in each planning cycle the intention recognition module chooses an unsatisfied Story Concern goal with the highest importance. The highest important Story Concern goal will be determined by applying Formula (1) to all Story Concern goals in the agent active roles and selecting the one with the highest value. In Formula (1), $w(g_{SB_i}, R(g_{SB_i}))$is the importance of the Story Concern goal in its associated role and $w\left(R\left(g_{SB_i}\right)\right)$is the importance of the role that Story Concern goal belongs to in the agent's current social context.

Formula (1):

$$importance_{g_{SB_i}} = w(g_{SB_i}, R(g_{SB_i})) \times w\left(R\left(g_{SB_i}\right)\right)$$

As soon as the agent achieves the final state in the plan to achieve the Story Concern goal, the goal will become deactivated permanently. This class of goals can be used to manage sub stories in the main plot, or managing the whole path of the story. For example, in a family drama the agent can have several Story Concern goals such as: getting married, buying a house, and having kids.

Here is a minimum structure to include Story Concern goals in the role structure:

- Name: Each goal has a name to make it distinguishable from others.
- Set of facts and their associated value: The goal world state is a conjunction of facts that each can have a different importance in the agent reward system. For example in the prisoner's dilemma the goal state will be defined by number of years in prison and outside of prison. One has negative valence and the other positive.
- Importance: The goal importance determines priority; higher priority determines the agent urgency to achieve the goal sooner. Goal importance helps the designer to create a meaningful sequence of milestones in the agent life span.

- Activation Status: Activation status will be used to check if the agent has already achieved the goal in the environment or not. Story Concern goals become deactivated and discarded after they are achieved.

The importance of Story Concern goal will be determined by applying Formula (1) to all active roles' Story Concern goals

b) Desire based

The agent should make a plan to achieve a desire based goal in the same manner as Story Concern goals. However, unlike Story Concern goals which are discarded once they have been achieved. Desire based goals need a mechanism to ensure that the agent develops a desire to achieve these goals again after the most recent fulfillment, as illustrated in Figure 3.13. A Decay rate variable in the desire based goal structure ensures that the agent develops an intention over time to achieve a desire based goal again. The presence of desire based goals prevents the agent from wandering around in the environment without any purposeful behavior, they also provide a great potential to represent the quality of being resource bounded in agents. Desire based goals can be used to embed needs or general routines in the agent. Their nature in directing agent motive has been inspired by Reiss 16 motive personality model that has been discussed in Section 2.3.1.2.

If the agent does not have any Story Concern goal left, the intention process begins evaluating desire based goals to choose one with the highest urgency. The urgency (importance) of each desire based goal has a direct correlation with the difference between its current value and threshold as well as its importance that drives from Formula 2. In this formula $w\left(R\left(g_{DB_i}\right)\right)$ is the weight of role that g_{DB_i} belongs to. $t_{g_{DB_i}}$, $cv_{g_{DB_i}}$, $w(g_{DB_i}, R\left(g_{DB_i}\right))$ are threshold, current value and weight of g_{DB_i} in the agent role that g_{DB_i} belongs to. More distance and higher importance indicate a higher urgency to achieve the goal.

Formula (2):

$$importance_{g_{DB_i}} = w\left(R\left(g_{DB_i}\right)\right) \times (t_{g_{DB_i}} - cv_{g_{DB_i}}) \times w(g_{DB_i}, R(g_{DB_i}))$$

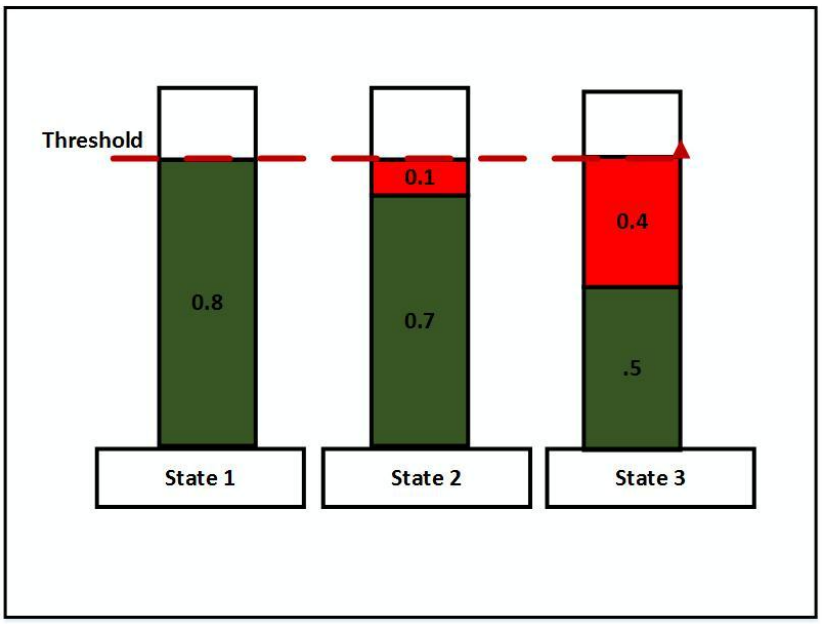

Figure 3.13: The agent desirability to achieve the goal increases as the current value decreases (due to decay rate)

The minimum set of variables for this class of goals is listed below:

- Name: Each goal has a name like: hunger, socializing.
- World State: Replenishment goals only will be defined with one fact.
- Range: Each goal has a valid range; this can be visualized with a tank that the agent needs to fill.
- Current Value: Is a current state of the goal according to the agent interactions in the world until a precise moment.
- Threshold: Threshold is a number in the goal valid range that the agent should minimally achieve to feel satisfied. This number varies depending on the agent personality. For example, a greedy person has a high threshold for gaining money.
- Importance: The goal importance will be determined by motivational trait of personality. Goal importance will be used in intention recognition to set the agent next goal.
- Decay rate: It determines the goal decay rate and makes sure that the agent starts planning for the goal after a period of time. For example, if Bob satisfies his hunger

by eating food, sooner or later, depending on this decay rate, he feels the urge to satisfy his hunger again.

3.5.1.2 Interest Goal

Interest goal is a group of goals that the agent does not actively plan to achieve. Unlike the first two classes of goals, Interest goal consists of only one Boolean fact and associated value. The goal state can be calculated based on non-Boolean continuous variables but the assigned value should be represented by true or false. For example, if the agent happiness has been defined as an IBG it still can be a function of several factors in the environment; however its representation as an IBG is either true or false. IBGs are facts in the world and based on their valence the agent either tries to preserve or avoid them, without explicitly planning to achieve them. For example if Alice is Bob's enemy, she may have the goal to see Bob unhappy (happiness (Bob) = false). This makes any state that Bob is unhappy more desirable for Alice. However, Alice will not make a plan explicitly to make Bob upset when Bob's sadness is her Interest goal. Our model has the flexibility to define the same goal as an Interest for one agent while the other agent considers it as a Replenishment goal. Another example is when the agent cannot make any impact on its desired goal state, such as the prisoner's dilemma where the agent can have a goal of not being betrayed. Regardless of how important this goal is to the agent, practically there is nothing that he can do to avoid being betrayed. But in his planning tree evaluation any state that he has being betrayed has a low desirability.

In the planning process the agent considers Interest goals; a state that preserves or contains Interest goals with positive importance has a higher desirability, whereas the agent avoids states that contain Interest goals with negative importance. This desirability naturally applies to the agent processing by state utility calculation. Finally this makes the agent develop a preference to take action in the path towards particular states and to avoid others.

For Replenishment goals there is a valid range and a threshold to indicate their level of achievement. However, IBGs state follows true/false principles, and there is nothing in

between. This makes the agent to only appraise emotion if the IBG state changes in comparison to the very last recorded state.

To achieve all these requirements, Interest goals need to have following variables:

- Name: Each goal has a name to make it distinguishable from others.
- World State and the associated value: Each goal has a fact and its associated value. As we have mentioned an IBG should only contain a Boolean fact.
- Importance: The goal importance is a positive or negative numerical value in a valid range.
- Last Status: The IBG emotion affect depends on its last stored status.

3.5.2 Personality Traits

Personality unifies all human behavior, temperaments, and mental traits. Human beings each have their own preferences and traits that affect decision making unconsciously. In the field of game and interactive drama, it is challenging to find one model that could reflect all aspects of the agent personality. OCEAN and Myers-Briggs both suffer from lack of deliberative goal selection and accurate mapping between traits and cognitive process. Alternatively, Reiss's model solely focused on deliberative goal seeking, doesn't give an explanation of reactive behaviors or effect of personality traits on emotion generation.

One of the biggest practical obstacles when borrowing a personality model from psychology is to define a well-defined mapping from traits to actions. Assigning multiple traits to the agent by itself does not create a personality for NPCs. Lack of an accurate mapping makes the personality model a doll mask that does not unify agent attitudes.

Our proposed solution to make personality traits more practical is to break them down into 3 general categories and provide an appropriate interface from each category to the agent component. There are three groups of personality trait in our model: Reactive, Behavioral and Context Free. Reactive personality traits affect how the agent appraised an event (they can override an Emotion Trait threshold) or how Emotion Trait should be integrated in an Emotional State. Behavioral Traits explicitly represent the agent

tendency or avoidance toward performing an action with certain attribute. Context Free Traits are mainly used in long shot decision-making such as planning.

3.5.2.1 Reactive Traits

The Emotion section defines all Emotion Traits and rules for emotion reactions, their integration to Emotional State Dimension and action tendencies. Motivation to define this class of personality traits is to define a way to customize each agent emotion responses. This customized reaction should be preserved in the agent life span as a consistent pattern of behavior.

The main functionality of the emotion module is to create a more believable agent by facilitating appropriate emotion responses through applying predefined rules. However, appraisal of the exact same emotion response toward one stimulus from different agents is as unbelievable as absence of emotion.

This class of personality traits affects the agent in one of the following ways:

- Adjusting Threshold: The Emotion Trait's threshold can be overridden to a higher or a lower degree depending on whether the agent is either more prone or has a high tolerance to a certain stimulus. For example a Person with a low tolerance (threshold) for anger becomes angry easier.
- Mapping Emotion: In the emotion module all agents follow the same set of rules for emotion appraisal. However in real life being exposed to the same event does not necessarily affect everyone in the same way. This attribute links one trait to another one.

Personality should unify an agent through all steps in its performance. Once these traits have been defined in the agent role, they work as a permanent filter on top of emotion rules in the role context. Separation of reactive traits from the emotion component and emotion model provides the flexibility to change any part independent from another.

3.5.2.2 Behavioral Traits

Behavioral traits make the agent represent a consistent pattern of action selection. Personality traits in this part are the same as tags that the designer has previously used in

action tagging. The designer can use any personality model or action categorization that he or she prefers, as long as tags have a meaningful connection to the action.

One of the fundamental factors in decision making is the agent behavioral traits exhibited when the agent is facing several available actions. This class of personality traits makes the agent not only represents goal directed behavior but also to prefer certain actions over others according to a consistent pattern of behavior in correlation with the agent social context or a set of active roles. For example, a friendship role demands the agent execute a more cooperative behavior even if he is not a very cooperative person in general. One behavioral trait can be defined in several roles; in decision making the agent uses the action context to find the most related role to formalize its behavior.

In addition to the trait degree, the trait weight also impacts the decision making. More dominant traits in the agent role will be represented with a higher weight. Figure 3.14 represents how an agent facing two different actions has a greater tendency to choose one action over the other based on his role personality traits.

The general structure for personality traits minimally needs the following components:

- Name: for each personality trait there should be an action tag with the same name. This enables the agent to use action tags in decision making.
- Value: Each personality trait has degrees that indicate the agent adherence to the trait. An agent can have a cooperation trait in his role as a friend and also in his default role but each with a different degree. This makes the agent more cooperative in one role in comparison to the other.
- Importance: Importance value determines the trait dominance in the agent action selection.

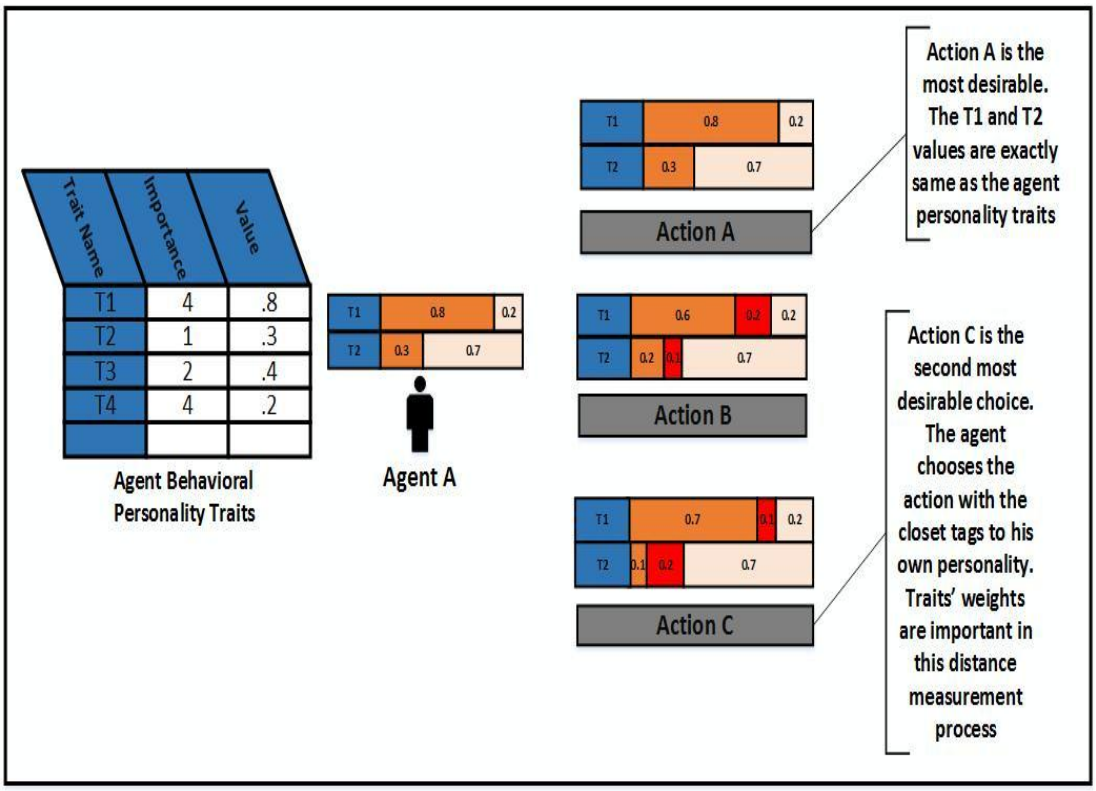

Figure 3.14: The agent behavioral trait.

In Figure 3.14, A is the most desirable action because T1 and T2 are the only traits in the action and their values are exactly the same as the one authored in the agent. In the absence of A, the agent needs to make a choice between C and B, which makes C the most desirable one. C and B both have the same accumulative difference for T1 and T2; however the trait's weight is another factor in calculating action desirability. T1 has a higher weight in comparison to T2, which makes C the more desirable action.

3.5.2.3 Context Free Traits

Context Free Traits can only can be defined in the agent default role. Their application and functionality is fixed and is independent from the agent social context. Context Free Traits are factors that affect the agent planning process explicitly. Care of consequence is an example of this category of traits; in state utility assignment an agent always has the same care of consequences in weighting its own utility against others. Context Free Traits can include qualities such as risk aversion or utility directed behavior.

3.5.3 Belief

Beliefs are mental attitudes characterizing how the world is viewed through the agent eye or a particular role. Beliefs determine how one should interpret events, world states, as well as actions. In our design belief is a very powerful component for encoding common sense knowledge into the agent architecture without using an ontology system. It facilitates the agent to evaluate the world state according to its roles. It also helps the agent to develop a more general categorization of actions independent from their tags and post conditions. The belief component consists of two main parts: Reward System and World view.

3.5.3.1 Reward System

As we have mentioned each world and goal state has a set of facts. Meanwhile based on a scenario there can be several facts that have not been included in the agent goals. However, the agent may encounter these facts and it will be considered vital from the believability perspective to demonstrate an appropriate reaction. In the social interaction the agent may not have any goal that defines happiness as a desirable state, but upon seeing its friend's happiness it should be able to at least understand the valence and desirability of this state. The reward system is the set of facts with their associated importance that the designer authors according to the agent characteristic and possible situation in a scenario. The Reward system plays a vital role in leaf evaluation that will be discussed in Section 3.7.2.1. The agent determines desirability of facts targeting other agents by filtering them through its own rewards system.

3.5.3.2 World View

World view provides the agent with a high categorization of actions. For example the agent can categorize cheating, lying to a friend, and betrayal in the group of immoral actions. This categorization enables the agent to develop a higher level of knowledge about others social behavior independent from action tags or post conditions. This higher knowledge enables the agent to make more accurate prediction in its TOM. If the agent wants to predict the next action of its opponent and traits in its opponents profile are not helpful in predicting the next probable action. The solution is to compare previously

performed action categories of the opponent with available actions. If an agent has been acting in a certain category, it has a high tendency to choose future actions from the same category. For example if Alice saw Bob helping a stranger, she identifies him as being more willing to perform actions from the same category (Like: being kind or honest) in future. Figure 3.15 is a schematic illustration of this module.

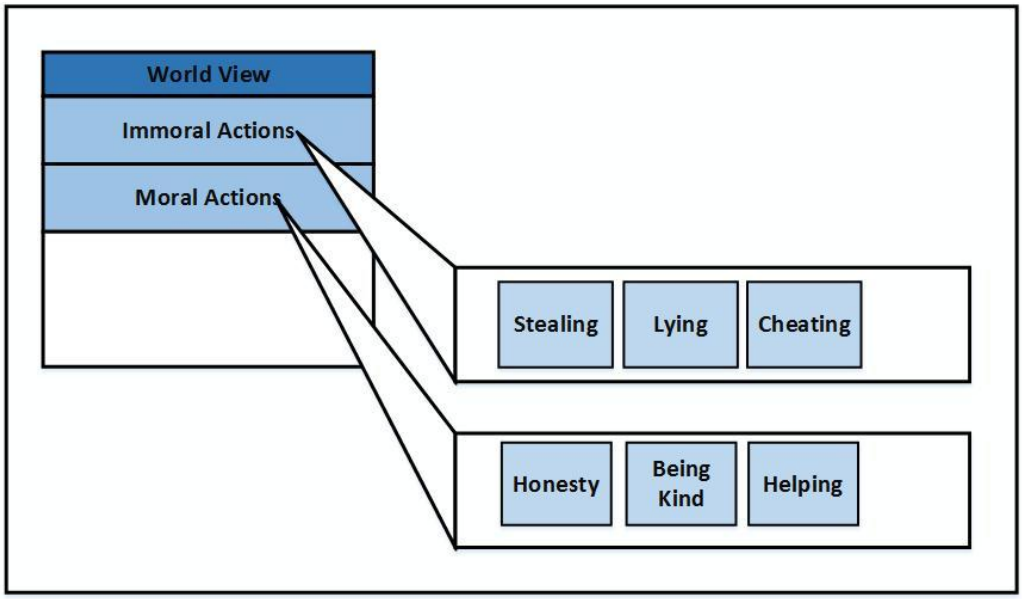

Figure 3.15: Schematic view of the Agent World view.

3.6 Intention Recognition

The Intention Recognition Module processes the planning agent's roles and the current world state, and chooses one candidate goal to be the active goal—i.e., the *intention*—to be pursued by the planning agent. The agent can have multiple Replenishment goals in each active role, but at the same time he only pursues achieving one of them, the active goal. The Intention Recognition Module follows process (1) to select an intention from the agent's goals.

The process bellows first attempts to find the Story Concern goal with the highest priority among all active roles. If there was not any Story Concern goal available it will check active role for the Desire Based goal with the highest priority.

Process (1):

Declaration:

Roles = Set {Set {<i, ActivePursueGoals, ReplenishmentGoals>}}

The set of roles associated with the planning agent, expressed as a tuple of role importance (i), active pursue goals, and Replenishment goals

StoryBasedGoal = Set { set <f, i>, p}

The set of the planning agent's Story Concern goals expressed as a tuple of the goal facts and their associated importance set <f, i>, and the goal priority (p)

DesireBasedGoal = Set {< f, i, p> }

The set of the planning agent's desire based goals expressed as a tuple of the goal fact (f), the goal importance (i), and the goal priority (p)

ActiveRoles(SetOfRoles, StateOfTheWorld)

Return the subset of SetOfRoles that are active given the StateOfTheWorld

HighestPriorityDesireBasedGoal (SetOfRoles)

Return a desire based goal in Role's set of goals that has greater or equal priority with respect to all other Replenishment goals in Role

HighestImportanceStoryBasedGoal(Role)

Return a Story Concern goal in Role's set of goals that has greater or equal importance with respect to all other Story Concern goals in the agent active Roles.

The Actual intention recognition Process:

IntentionRecognition(Roles, WorldState):

```
CurrentlyActiveRoles:= ActiveRoles(Roles, WorldState)
If CurrentlyActiveRoles = nil
    Return nil
Else
    ActiveGoal:= HighestImportanceStoryBasedGoal
    (CurrentlyActiveRoles)
    If CandidateGoal = Nil
        CandidateGoal:= HighestPriorityDesireBasedGoal
    (CurrentlyActiveRoles)
Return CandidateGoal
```

The Intention Recognition Module first activates high importance Story Concern goals, and then resorts to high priority Desire Based goals only when there are no Story Concern goals to activate. Note that although in process(1) the importance of Story Concern and desire based goals are modeled as variables in fact they drives from Formula(1) and Formula(2). In the intention recognition active roles, and their weight has been applied through using two mentioned formulas that make the agent active role to be selected based on active roles. The candidate goal, or intention, will be selected by the Intention Recognition Module and will be passed to the Planning Module.

3.7 Planning

After intention recognition, the agent starts planning. Planning consists of searching for a sequence of actions—called a *plan*—that results in a goal state. Agent planning has two phases; in the first phase the agent acquires a planning tree either by building one or retrieving one that was pre-authored for the situation. In the second phase it traverses the tree to find the best plan among all possible plans. The planning tree should not be confused with the planning agent's plan; a planning tree is a tree of potential action

possibilities, whereas a plan is a sequence of actions that lead to a planning agent's goal state.

The planning process starts by receiving the active goal from the intention recognition module and ends with a plan—sequence of actions—that the agent needs to perform to achieve the goal. This process is illustrated in Figure 3.16. We will first briefly discuss construction of the planning tree, and then discuss different types of nodes in the planning tree as well as processing steps performed by agents that are peculiar to each node type.

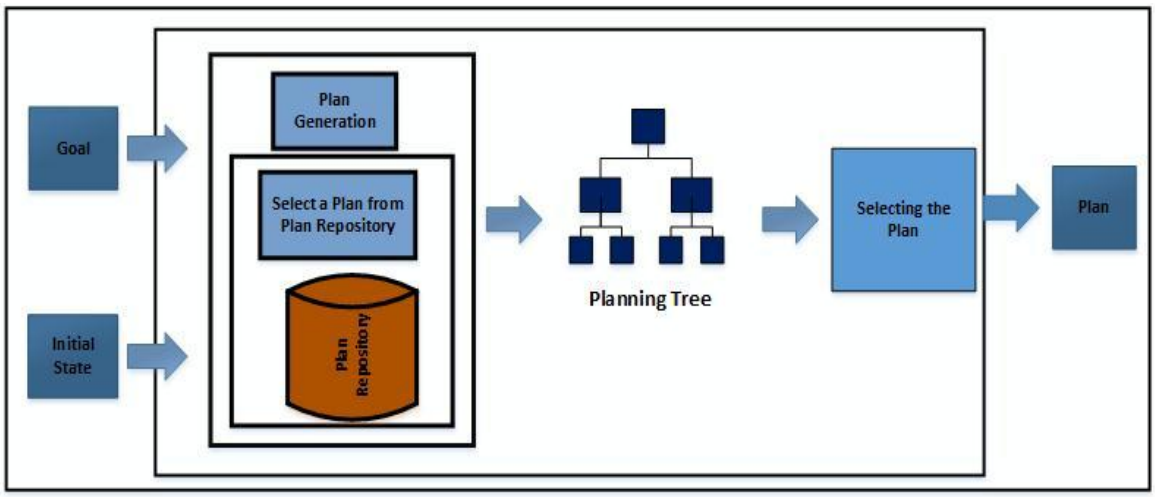

Figure 3.16: General overview of planning process.

3.7.1 Acquiring Planning Tree

The Acquiring Planning Tree module receives two inputs: (a) the intention from the Intention Recognition Module, and (b) the initial state of all agents in the planning agent's environment. Unsurprisingly, the Acquiring Planning Tree outputs a planning tree. The initial state of an agent includes its Emotional State, active agent roles, and the current state of the world. The planning tree output by this module is a simulation of the world environment that consists of world state and various action sequences that lead to goal satisfaction.

Note, however, that degree of goal satisfaction may vary; that is, a particular sequence of actions that lead to a goal state (i.e., a particular traversal of a completed planning tree) may be more or less satisfying to the planning agent than another such sequence. This "multiple solutions" property of planning trees makes planning tree evaluation (described later) a crucial part of the planning process.

The design of the planning tree evaluation process imposes some constraints on the structure of planning trees. Branches descending from planning tree nodes model potential actions and/or consequences of actions committed by agents. The existence of a branch must be consistent with a simulated world state that corresponds to the combination of (a) the initial world state input to the Planning Tree Generator, and (b) additional and/or modified state that arises from applying all actions and consequences corresponding to traversing the planning tree from its root to the node from which the branch descends. Put another way, each branch must represent a valid "next action or consequence" in the series of actions and consequences modeled by the unique traversal from the root node to the node from which the branch descends. Planning trees are composed of the following elements:

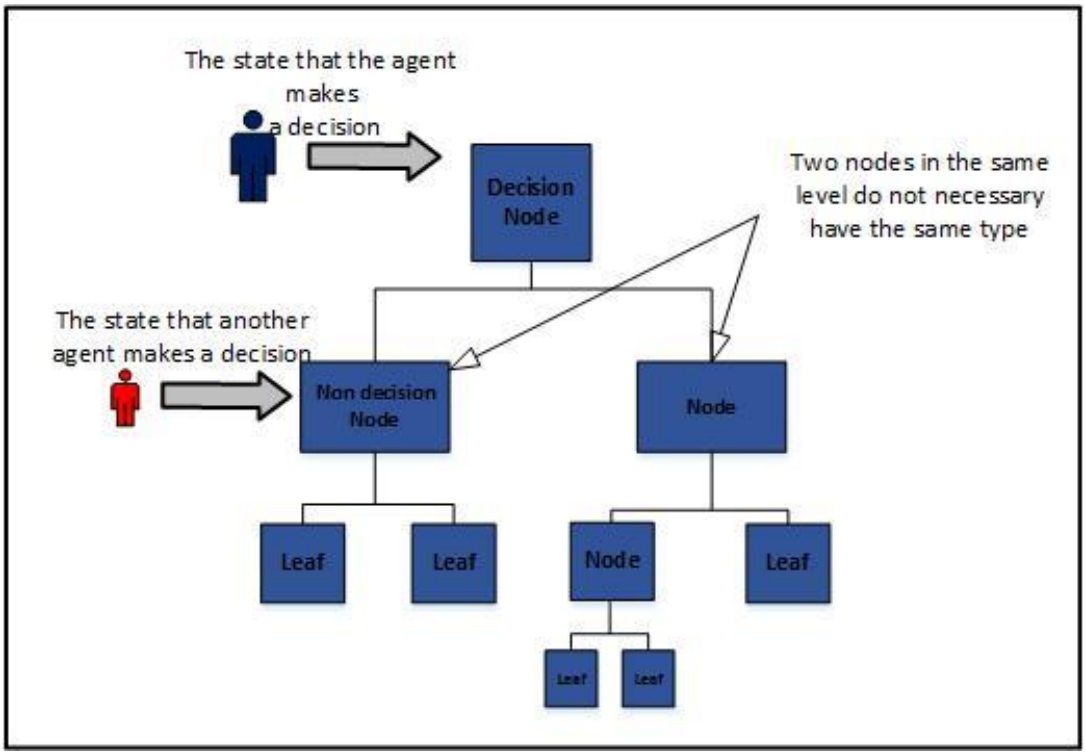

Figure 3.17: General planning tree.

Decision nodes: These are internal nodes with descending branches that represent different potential actions that could be chosen by the planning agent according to the simulated world state (Node type A in Figure 3.17)

Non-decision nodes: These are internal nodes with descending branches that represent one of two things depending on the meaning of the branch that connects the non-decision node to its parent. If the branch represents an action that has multiple consequences, then the branches descending from the non-decision node represent the set of potential consequences associated with the action as illustrated in Figure 3.17 by node type C. Otherwise, the branches descending from the non-decision node represent potential actions that could be taken by agents other than the planning agent, which is represented with node type B in the same figure.

Leaf nodes: Ideally, leaf nodes represent goal states with respect to the planning agent's intention. Leaf nodes store the full set of facts that correspond to the sequence of actions and consequences modeled by the unique path from the root node to the leaf. Leaf nodes

that do not correspond to goal states only exist in the event that the Planning Tree Generator was not able to find any paths from the world state to a goal state. The facts stored in a leaf describe the set of post-conditions applied by all actions in the path from the initial state till the current leaf node.

The only constraints on planning trees as a whole are that the root node must be a decision node and the tree must be of finite height.

There are at least two potential approaches to provide the agent with the planning tree: (a) choosing a template plan from a planning repository, (b) generating the planning tree. Both approaches result in a planning tree with the same structure that the agent can use in the second phase.

Choosing between these two alternatives entails a tradeoff between (a) time vs. space complexity, and (b) development effort. In the first case, a plan repository may require a large amount of storage, but plan retrieval will be faster than generation because it will consist of a simple query to the repository. In the second case, the developer must weigh the effort required to develop a sufficiently complete repository against the effort required to produce an adequate planning tree generation algorithm for discovering plans that are likely to lead to goal satisfaction in a finite number of steps.

The first option mentioned above is to use a planning repository that can be provided to the agent offline by an expert or a designer. A planning repository is indexed by the initial state(s) and goal state(s) associated with each planning tree. Producing a planning tree is then a matter of matching inputs with appropriate index entries and output one of the planning templates.

The second option is to generate the planning tree each time the agent aims to achieve a goal based on the initial state. This approach is more flexible to be applied in dynamic environments like games. As it has been discussed in Chapter 2 one of the main motivations to design believable agents is failure of classic methods in facing unpredictable situations. Generating the planning tree at runtime addresses this problem by considering initial and possible intermediate states. A plausible solution to this

problem is to apply forward chaining from the world state in search of a goal state. By design, each world state includes set of atomic facts that have an assigned numerical or Boolean value; every action is a name coupled with pre- and post-conditions—facts with predetermined values; and actions with more than one post-condition will only yield one of the post-conditions at runtime. By repeatedly applying actions as functions that map a world state containing the necessary pre-conditions to one or more new world states—each containing a post-condition—the Planning Tree Generator can track its simulated world state and construct a valid planning tree.

This approach can be combined with an *admissible heuristic* that produces a subset of available actions that could lead to a goal state. The designer can define these admissible heuristics based on system criteria. Because the agent should be able to perform the planning in real-time, putting time constraints on the planning tree generation is a rational option. This time constraint prevents the agent from staying long enough in an idle state during planning to break the audience immersion. Depending on other parts of system requirements, limiting the planning tree by number of CPU cycles and memory usage is another applicable heuristic in order to prevent other system's processes from starvation while the agent planning is using all the resources. The designer can also put a limit on the depth of the planning tree; this means if the goal could not be achieved within a certain depth the agent drops the planning tree generation. Lastly the designer can keep the branching factor not to exceed more than a certain number. These *admissible heuristics* can help to limit the risk of dealing with a potentially very large planning tree.

Naturally, nodes that have reached a goal state are stored as leaves and not expanded. Since planning trees must be of finite height, tree generation can abort after some number of iterations chosen by the implementer. The tree is then either (a) pruned such that all leaves correspond to goal states, or (b) left "un pruned" as the algorithm failed to find any paths from the current world state to a goal state. Such a tree construction algorithm allows the agent to turn planning into a path finding problem, seeking a path from the current world state to a goal state, or to "the best non-goal state" in the case that no such path exists in the planning tree.

3.7.2 Selecting an alternative: Processing a planning tree

After the planning tree has been generated in the first phase, the agent must select a plan—a sequence of actions described by a top-down traversal from the root to a leaf. The first step in selecting a plan is to visit all the tree nodes and annotate them with a *utility* value based on (1) the node type, (2) the planning agent's roles; (3) the planning agent's embedded personality traits, (4) the planning agent's reward system, and (5) annotations of descendent nodes (6) the planning agent's TOM profile that has been discussed in Section 3.4.1.4. As such, the agent visits nodes bottom-up, breadth-first.

The planning tree contains three different types of nodes that have been briefly mentioned in the planning generation section. The annotation process for each node type is described below.

3.7.2.1 Leaf

Leaf nodes are nodes with no descendants. Each leaf in the planning tree contains one or more facts that correspond to all sequences of the unique action leads to this node from the initial state. All actions' consequences leading to a leaf node will be presented in the leaf node, so the agent performs the state utility only once and it evaluate action consequence within a path and not individually.

Facts in each leaf may be categorized into the following two disjoint sets: a) facts that are members of the planning agent's interest-based goals or the active goal and b) facts that are not members of the planning agent's interest-based goals or the active goal and they do not target the agent as it has been illustrated in Figure 3.18. We differentiate between these two sets because the agent follows two different processes to evaluate them (one for each set). Leaf utility is calculated by Formula (3), where COC is the agent care of consequence, SD is the agent self-desirability and OD is other desirability.
Formula (3):

$$U_{L_i} = \sum (1 - COC) \times (SD) + (COC) \times (OD)$$

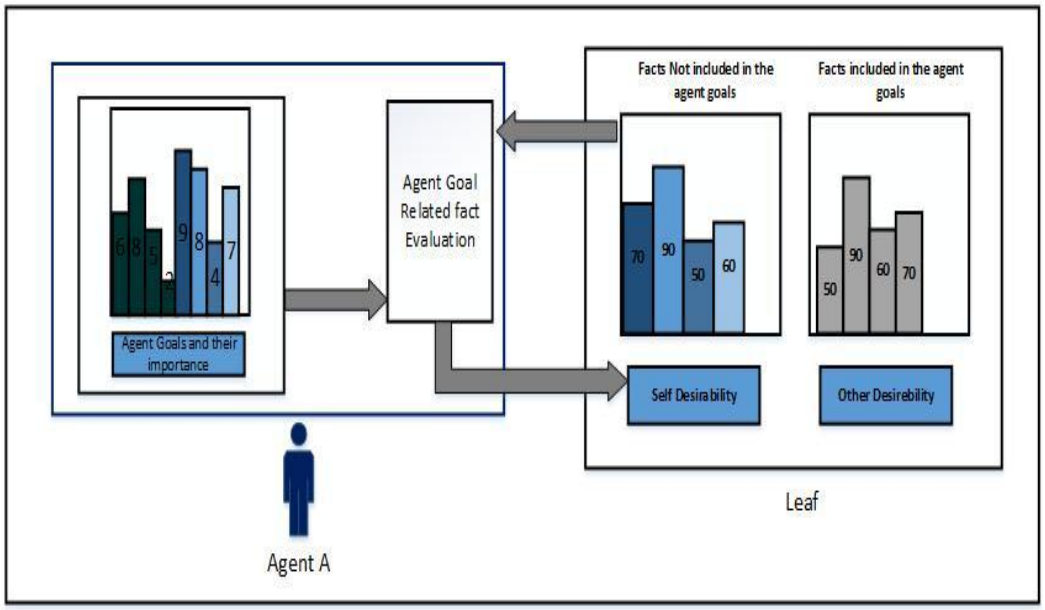

Figure 3.18: The agent processing facts related to its goals

A) Self-Desirability

Self-Desirability determines the level of leaf desirability based on its determined facts value that is: 1) in the agent goal scope. The agent goal scope for each planning tree includes all the planner agent active roles' Interest goals and the active goal. The goal scope excludes other Replenishment goals in the planner agent active roles in order to put the focus of leaf utility calculation in the planning tree on evaluating facts only related to one specific goal. 2) Targets the agent. Facts that can be categorized in one of the above categories will be applied to Formula (4) Self-Desirability considers role and agent reward system in its calculation. The importance of a certain fact is not necessarily the same for two different agents. Due to the open world assumption the agent cannot make any assumption about unknown fact values. As such, the agent only evaluates facts with known values based on Formula (4).

Formula (4):

$$Self - Desirability_{L_i} = \sum_{i=1}^{n} D(f_i)$$

For evaluating facts related to the active goal of the planning or those that target the agent, Formula (4) is sufficient. $D(f_i)$ is a desirability of a fact based on the agent active roles' reward system. Generally the agent uses Formula (5) to calculate a weighted average for a given fact based on the agent active roles. In Formula (5) $D(R_j, f_i)$is the desirability of fact f_i in the role R_j reward system and $w(R_j)$ is the importance of roleR_j.

Formula (5):

$$Desirability(f_i) = \sum_{j=1}^{n} \frac{D(R_j, f_i) \times w(R_j)}{w(R_j)}$$

In the case of interest-based goals there is an additional step. The agent compares each interest-based goal state with its state in the initial node, and it only applies Formula (4) if the goal state got changed in the planning process, the reason and how this evaluation takes place has been explained in Section 3.5.1.2.

B) Other Desirability

Other desirability is a metric to evaluate all facts that are not either in the agent goal scope or does not target it. Despite Self-Desirability; other desirability is utility consideration for part of the world state in the leaf that is not related to the planning agent through its roles.

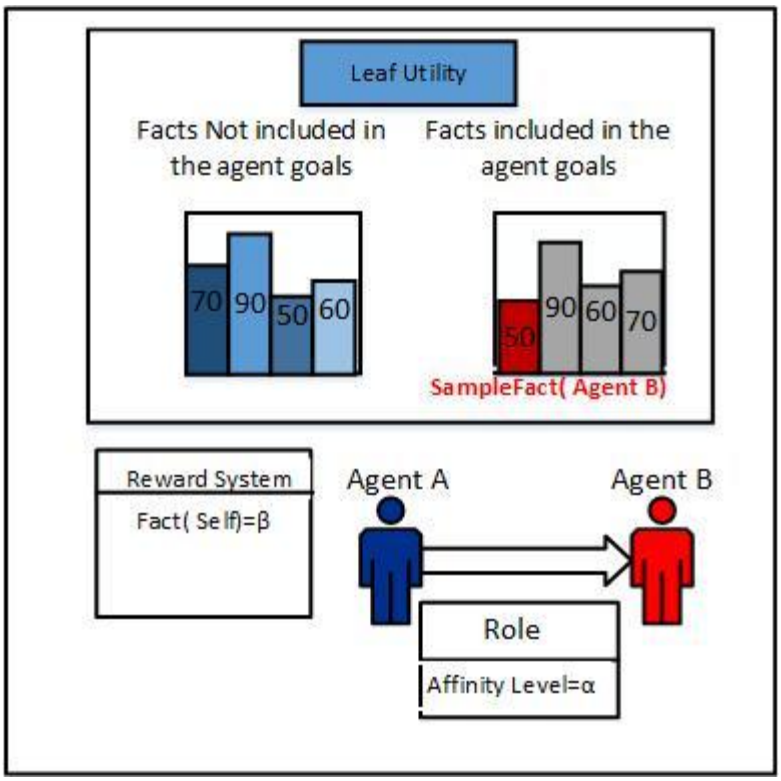

Figure 3.19: Other-Desirability from agent A perspective

Figure 3.19 illustrates agent A evaluating "samplefact" that targets agent B by applying Formula (6). Formula (6) applies A affinity *($A(T(f_i))$)* toward B as a mediator for applying the desirability stored in A's reward system for f_i if it were targeting A, *($D(f_i)$)*. I.e., A considers whether it would like to be (or is rewarded by being) targeted by f_i, and scales that desirability according to how much it likes or dislikes (or how much it has affinity for) B. In this way, measures of selfish reward and affinity are combined to model empathy toward other agents during the decision-making process.

Formula (6):

$$desirability\ for\ f_i = D(f_i) \times A(T(f_i))$$

$$where\ T(f_i) is\ f_i\ target$$

General calculation to calculate the leaf node other desirability has shown in the Formula (7). In Formula (7), $D(f_i)$ is desirability of fact f_i in the planning

agent reward system that has been discussed in [3.5.3.1] and $A(T(f_i))$ is affinity level of the planning agent towards the target of fact(f_i).

Formula (7):

$$OD = \sum_{i=1}^{n} D(f_i) \times A(T(f_i))$$

Agent affinity is a real number bounded, inclusively, by -1 and +1. Therefore, an undesirable fact for an enemy with affinity level of -1 is actually desirable for the planning agent, even when the fact is not relevant to any of its goals and affects the evaluation of that branch. Intuitively, this mechanism is similar to how humans function in real life; they usually "put themselves in the other person's shoe" as well as considering their affinity towards that person.

The final step in visiting a leaf node is to compute the weighted sum of Self Desirability and Other Desirability, according to Formula (3). Notice that care of consequence is a context free trait that mediates consideration of selfish and empathic contributions to the utility measure.

3.7.2.2 Non-Decision Node

As discussed earlier in Section 3.7.1 a non-decision node is an intermediate node with multiple direct descendants. When such nodes are visited by the planning agent, they are annotated with probabilities associated with each of their descending branches. A non-decision node may be generated for two possible reasons: (1) there are multiple potential post-conditions associated with a single action taken by the planning agent (node type B in Figure 3.17, and (2) planning depends on the decision of another agent (node type C in the same figure). In the first case, the non-decision node separates multiple post-conditions that bring the agent to the non-decision node, from the same antecedent decision node. Probability of each post condition has been provided by the designer based on the scenario. In the second case, non-decision nodes separate individual choices of action associated with a single post-condition, each branching from the same decision node, from multiple actions that other agents may commit in response to each choice of action. In both cases, the branches that descend from non-decision nodes represent

uncertainty; which branch descending from a non-decision node is actualized in the game world will be determined at runtime, either by the world's probabilistic determination of action outcomes, or by other agents' decision-making processes.

The two cases for non-decision node generation are associated with two different processes for calculating the probabilities attached to the node. In the first case of multiple post-conditions for a single action has been given to the agent by designer as shown in Figure 3.20. In the second case of waiting for another agent to make a decision, TOM module uses the decider profile as it has been discussed in Section 1.5.1.4 to assign probability to other agents' available actions, as shown in Figure 3.21.

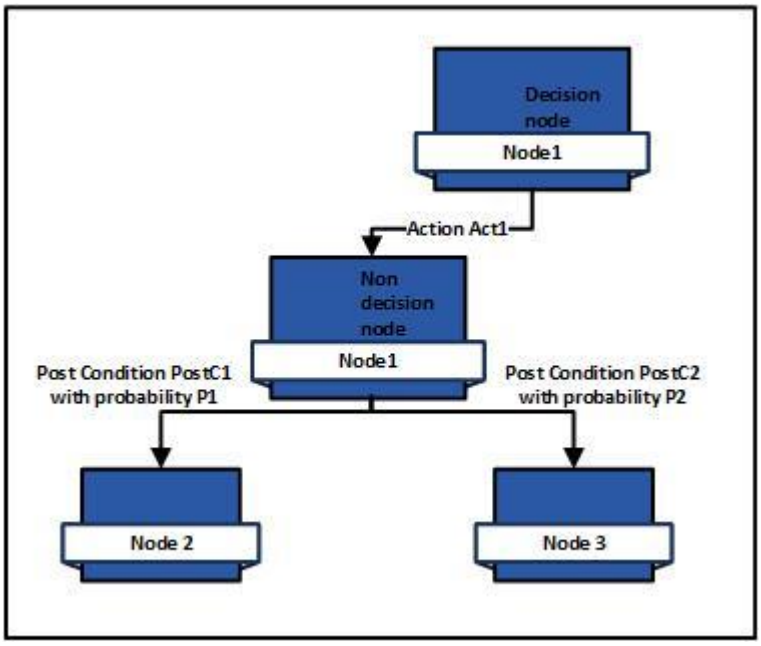

Figure 3.20: A Non-decision node because of possible post conditions

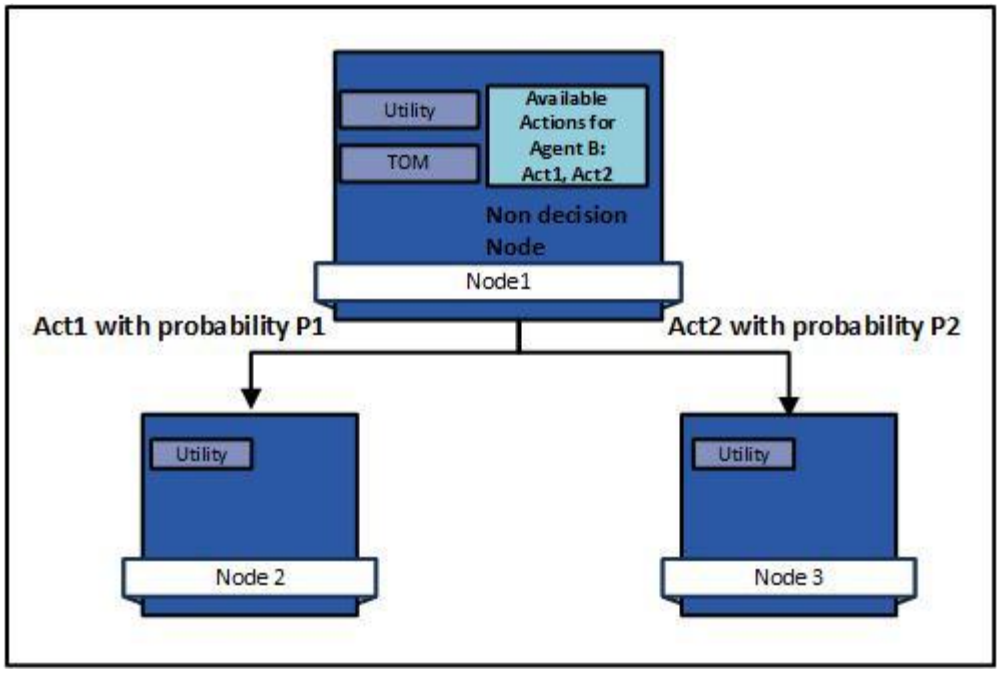

Figure 3.21: A Non-decision node because of multiple available actions for another actor.

Since branches from non-decision nodes represent an exhaustive set of alternatives with probabilityP_i, the probabilities associated with each non-decision node must sum to one, as described in Formula (8). In the runtime only one of these possibilities will be applied to the world state by the Event Handler.

Formula (8):

$$0<\text{Pi}<1, \quad \sum_{i=1}^{n} P_i = 1$$

All non-decision nodes have more than one edge with a probability between zero and one associated with each edge. The utility measure associated with each non-decision $U_{ND}$is simply the weighted sum of the utility of its direct descendantsU_i, weighted by each branch's estimated probability (P_i). This calculation is summarized in Formula (9).

Formula (9):

$$U_{ND} = \sum_{i=1}^{n} P_i \times U_i$$

In the case of calculating probabilities with the help of TOM, the agent performs a series of steps to ensure the constraints detailed in Formula (8). If the agent does not know the decider from before, he will use the stranger profile that has in his TOM by default.

1) Trait weight: the planning agent compares each action with the profile of the decider, that is, the agent who is supposed to make a decision to select a branch descending from the non-decision node. Each action has been tagged with several traits and their associated value as it has been mention in Section 3.2.2, for example attacking action can be tagged to be violent by degree of 1. For each action tag (T_i) the agent calculates the distance of the given tag value in the action ($T_i(\alpha)$) from the associated trait in the decider profile($T_i\left(B_{tom_A}\right)$) based on Formula (10). In Formula (10) agent A that is evaluating the planning tree uses agent B profile in its TOM (tom_{AB}).

Formula (10):

$$W_{T_{i_{tom_{AB}}}} = 1 - abs(T_i\left(B_{tom_A}\right) - T_i(\alpha))$$

2) Action weight: The agent repeats the same procedure for all action tags, then it adds up all trait weights ($W_{T_{i_{tom_{AB}}}}$) from Formula (10) to find the action weight ($W\alpha_{TOM_{AB}}$) by applying them to Formula (11) .Higher desirability for action α indicates that, based only on TOM profile that A has for B (tom_{AB}), the decider(B) is more likely to choose action α. Of course, this is an approximation the decider profile in the planner TOM is not necessarily accurate that may cause the agent to make a wrong prediction. This inaccuracy can have several reasons: a) the TOM profile only contains some traits b) the agent model is not necessarily accurate. If the agent using pre authored profile the designer may un purpose provide it with set of inaccurate profile to simulate his own desirable situation. Furthermore in lack of pre-authored profile the agent needs many interactions to complete the profile in the runtime c) traits are only one factor involved in decision making event with a complete profile consistent with the decider model, other factors such as the decider roles and goals should be considered to make the best prediction.

Formula (11):

$$W\alpha_{TOM_{AB}} = \sum_{i=1}^{n} W_{T i_{tom_{AB}}}$$

3) Probability of each action: The last step is to assign a probability to actions according to their weights. Higher weight means higher probability, the agent calculate W_{tot} that is aggregate weight of all actions ($W\alpha_{TOM_{AB}}$) from Formula (11) for the non-decision node. Probability of edge i ($P_{i_{TOM_B}}$) will be calculated by Formula (12).

Formula (12):

$$P_{i_{TOM_B}} = W_{i_{TOM_B}} / W_{tot}$$

3.7.2.3 Decision Node

As discussed earlier in [3.7.1]a decision node is a node with descending branches that model potential actions the can be taken by the planner. The first step is to find how much the agent personality matches with each action. For each action the agent should evaluate all traits, the process is similar to finding the level of desirability of an action for another actor based on its profile with couple of modifications.

For each trait the agent first uses its role associated with the planning tree goal if that role does not have this trait the agent check its other active roles in order of their importance, finally the last option is to check the default one. The agent uses Formula (13) to determine the level of desirability for each action based on its tags. In Formula (13) in addition to the distance between the tag and trait value, importance of the role ($I(R)$) and trait weight ($W(R, T_i)$) affects the Action weight. This mechanism guarantees that dominant personality trait and higher priority roles have a higher impact in decision making. In decision nodes lower weight indicates more harmony between the agent personality and the action.

Formula (13):

$$W_i = \sum_{i=1}^{n} (T_i(A) - T_i(\alpha)) \times W(R, T_i) \times I(R)$$

In this stage the agent has a weight assigned to all available actions; bottom up approach enables it to know the utility of each action descendent node as well. The agent choice depends on how it balances between higher utility and acting according to its roles and personality traits. In our system the default algorithm is to find the base ratio which is utility of the most desirable action (the one with the lightest weight) descendent node divided by its weight. If in set of action- utility available in the decision node, there is one with the higher ratio the agent choose that one.

There can be a context free trait as it has been discussed in [3.5.2.3] called Utility Consideration that affects the agent decision. It determines how much the agent is willing to deviate from its personality trait to gain a higher utility. The agent with high level of Utility Consideration always chooses the highest utility. Low level of this trait prevent agent from any deviation from its roles' personality traits that means the agent always chooses the lightest action. We will discuss Utility Consideration as an example trait that can be used in a decision node further in Chapter 4.

The action selected in each decision node will be saved as a sequence in a plan and will be kept in Memory module planning component. The plan will be kept in the Memory section; if the plan becomes invalid the agent will start re-planning which is feeding the planning module with initial state and active goal.

3.8 Summary

In this section we explained data flow and agent architecture thoroughly. The agent data flow ensure that all of the agent interaction in the world including, emotion appraisal, state utility, planning, and emotion reaction will affected by the agent role. Personality traits have been embedding in the agent role so that they can make an affective connection with other agent components such as emotion, action selection and planning. TOM module adds a novel flexibility to our agent to be able to make a plan based on probable actions of others. Furthermore planning facilitates role based utility and action

evaluation; interestingly it has the strength to deal with elements of non-deterministic environment like other actors.

4 Proof of Concept

The main goal of our architecture is to create believable agents according to the criteria that have been discussed in Chapter 2. In Section 2.1 we discussed a classic scenario that demonstrates how a purely mathematic approach fails to predict human behavior. In Section 2.2 we also discussed a couple of widely-used classic methods in AI games, and the problems with applying them in a dynamic environment. Our proposed design solution for a believable agent has been thoroughly explained in Chapter 3. This architecture can greatly enhance the creation of believable agents by facilitating goal driven, as well as reactive, behavior based on the agent's social context. The agent's behavioral patterns can be defined through goal importance and action tendency depending on the social context that has been introduced to our model by the role-based architecture. Role-based architecture unifies agent motivation and priority, as well as reactive and deliberative behaviors. The agent planning tree uses TOM profiles to predict the other actors. The TOM module enhances the agent believability by taking into account other actors just as humans do. Furthermore, by considering other actors' decisions by means of TOM, the agent is more likely to develop a better plan. It also makes the agent more situated by considering the other actor preferences, meaning that the same agent, in dealing with different actors, does not necessarily picks the same action. In our design, the data flow that has been discussed in Section 3.1 ensures the agent is using all of its active roles during its interaction with the world.

In this chapter, as a proof of the concept a prototype has been implemented with JavaEclipse on the Microsoft Windows 7 platform. To demonstrate and assess the ability of our prototype system of delivering believable behavior according to our design goals, we applied our prototype to a classic case study problem, that of the prisoner's dilemma [51]. In doing so, we were able to show situated decision making based on agent social context, including who the agent is interacting with and what the effect of the agent's decision on other actors involved is. One of the main goals of the system is to simulate believable agents; however, an evaluation of believability requires extensive user testing which is beyond the scope of this thesis. Furthermore, believability is subject to individual and cultural biases. Due to this technical difficulty, we decide to put the main

focus of this prototype on satisfying believability based on the criteria that have been discussed in Section 2.1.1 and leave the user testing for future work.

Our proposed agent demonstrates the ability to: a) show the role based architecture potential to make the agent's decision a situated one; b) exhibit the application of the planning algorithm discussed in Section 3.7.2 and its integration with the agent's social context; c) take into account the decisions of other agents by means of TOM that have been discussed in Section 3.4.1.4. To provide a simple and objective evaluation, the system has been tested in a well-known scenario: the "prisoner's dilemma". In this chapter we will discuss the achievement of these three objectives in our simulation through the addition of variations to the prisoner dilemma.

In Section 4.1 we briefly discuss those parts of the prototype that have been used in the evaluation. Section 4.2 briefly discusses the prisoner's dilemma and its application in real life situations. Section 4.3 gives a brief introduction to the planning process proposed in our model for the prisoner's dilemma tree. Section 4.4 thoroughly describes each scenario. Lastly Section 0 describes a summary of findings.

4.1 Implementation Specification

Our current prototype role structure, TOM and planning module, Memory, role, planning, perception and Event Handler and data flow between them has been implemented according to description in Section 3.1. Figure 4.1 shows the implemented part of design with highlighted blue color. In the following paragraphs we briefly discuss each of these implemented modules and their specifications in our evaluation.

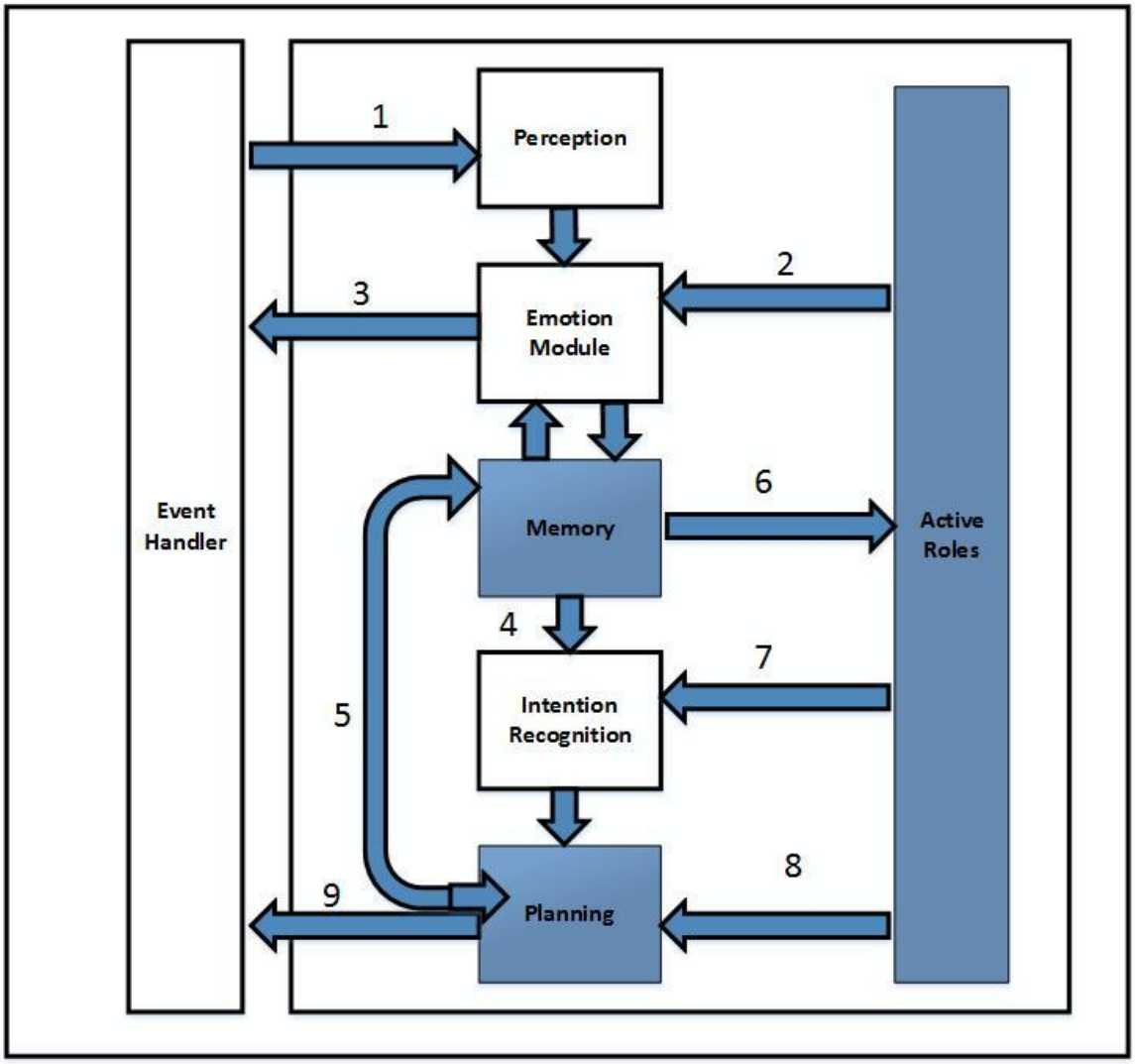

Figure 4.1: Implemented part of the design

The Authoring module, as it has been discussed in Section 3.2.5 enables the designer to create, change, and customize a role template, as well as TOM profiles, based on the description provided in Section 3.4.1.4. The Authoring Tool, as discussed in Section 3.2.5 can be used to: a) test the effectiveness of a trait; b) complicate the prisoner's dilemma by adding new roles and new situations; or c) evaluate the effect of our TOM module on the agent's decision making. One of the innovations in our agent algorithm is final state (leaf utility) evaluation based on the agent's role, as discussed in Section 3.7.2.1. Each agent has its own reward system and interest-based goals that will be used for leaf utility calculation. How much the agent values each fact or element of a potential

outcome of its action has been defined in its role reward system. For example, the cost of spending a year in prison or outside of it can be different for each individual; an ex-convict, for example, may not be as afraid of spending 5 year in prison as someone tasked with the responsibility of supporting their family. If two prisoners are members of the same band of thieves, one who rats out the other one may go free but will pay for his betrayal once his band finds out about it. Similarly, if the prisoners are enemies, one may choose to punish his opponents with little regard for the consequences to his decision. These are all factors that the designer can modify with the Authoring Tool.

The flexibility and usability of each component are other main non-functional specifications of the design. The implemented system satisfies these specifications through a role template component that accepts human-authored templates which can then be assigned to several agents. In our template role structure, TOM and the planning module have been implemented completely. We have also implemented the OCC emotion model to check its integration into our own agent emotion module. The Memory module includes Event History, TOM, and Active Memory, accompanied by data flow between the Memory and emotion components have been implemented based on the proposed description in Chapter 3 and Figure 3.1. However, in the evaluation, we did not use data flow between emotion and Memory; we leave the implementation of this part for future works. The test scenario focuses on the functionality of role, TOM, and planning.

In Section 3.4.1.4 we proposed two methods to initialize TOM profiles. The first option is to write specific profiles based on the designer's story. The second choice is to let the agent complete its profiles in the runtime by interacting with others and interpreting the other agents' personality traits according to the actions of other agents which they observe. In our current template, to test the functionality of TOM to predict other agents' actions and affect the agent's decision making, we decided to pre-author agents' profile with the Authoring Tool. This decision took place because, in the proposed problem, we are curious to check the functionality of our model in specific scenarios; furthermore, completing the other agent's profile in the runtime requires many previous experiences.

In our current template scheme, between the two alternatives of generating a planning tree, or using the planning repository that has been discussed in Section 3.7.1,]), we choose the second one. Getting a planning tree from the plan repository is well-suited for small environments with limited factors involved, so that the designer can easily provide all possible steps and agents' roles do not impact the planning tree generation. We took into account the fact that the second phase of planning which outputs the final plan is independent from how the planning tree has been generated. Figure 4.2 is a prisoner's dilemma planning tree annotated with relevant factors stored in the system. Inside the leaf nodes, there are facts that represent the results of potential decisions, where the right side indicates Alex's out come and left side is Mark's. For example, if Alex and Mark both betray each other, they both receive 5 years of prison. In the run time, the Event Handler, as discussed in Section 3.2.3, collects agents' decisions and distributes the results accordingly. The Agent decision is either to betray or cooperate and the result is a final number of years in prison for each agent.

In the prisoner's dilemma, non-decision nodes in the same level have the same set of available actions. However, our design as it has been discussed in Section 3.7.2 supports asymmetric trees as well. Each non-decision node's set of probabilities can be calculated independently of other nodes; the only important factor from a TOM processing stand point is the set of available actions (for a given non-decision node) and their given tags, as has been discussed in Chapter 3.

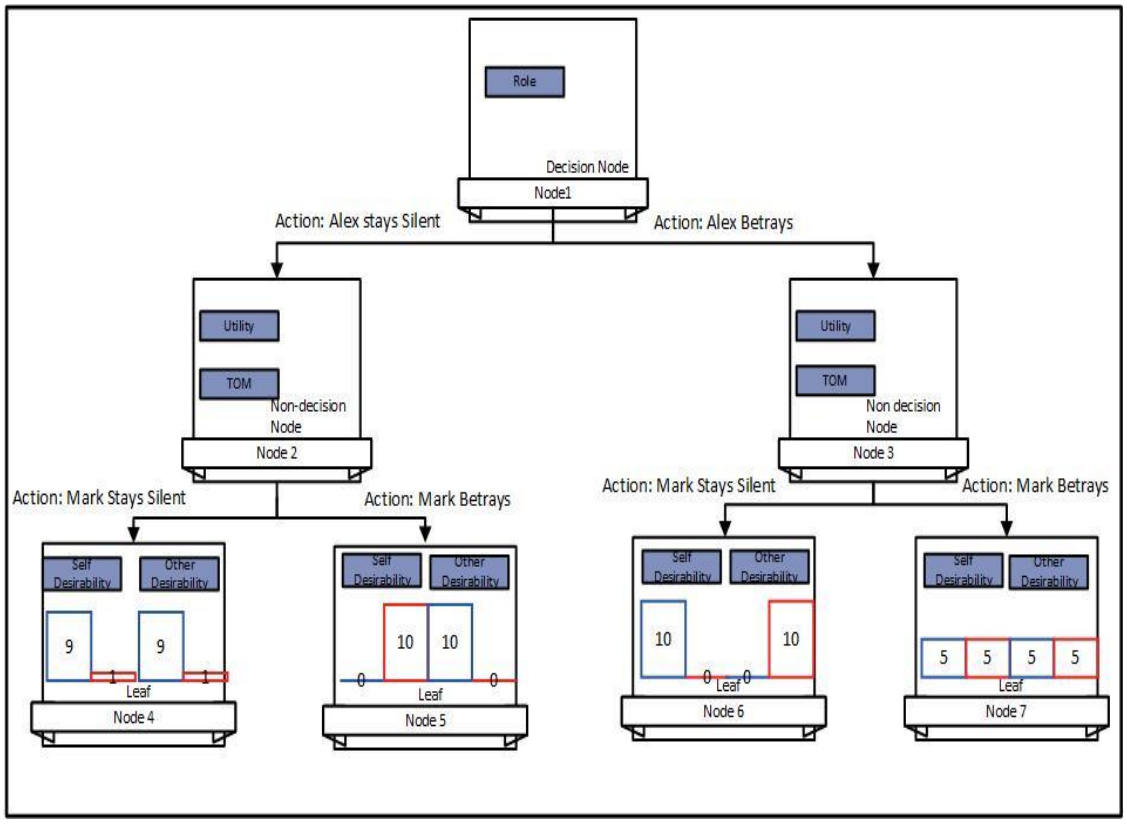

Figure 4.2: Decision tree for prisoner's dilemma

The general agent structure and data flow follows the description in Chapter 3. The role and TOM of each agent can be written and modified via the Authoring System. The planning tree evaluation starts from the bottom by leaf utility calculation, as described in the planning section of Chapter 3. This evaluation uses the agent's active roles weights, reward system, and possibly context free traits that have been implemented based on the specifications in Chapter 3. The second step after calculating leaf utility is to process non-decision nodes.

The Agent uses TOM to predict other agents' actions; this prediction can have a significant impact on the planner's final decision. TOM uses the opponent profile to assign probabilities to each available action in a non-decision node that need to be resolved by other actors' decisions in the runtime. With a functional description of the system in place, we will first describe the prisoner's dilemma and then discuss how our

model traverses the planning tree. Finally, we represent our model's potential to create new situations and socially-aware agents that can make decisions

Finally we represent our model potential to create new situation and socially aware agents to make decision.

4.2 Prisoner's Dilemma

The prisoner dilemma was originally framed by Merrill Flood and Melvin Dresher. It is a two player, perfect knowledge, and a non-zero sum game that both players can win or lose to varying degrees. It uses a hypothetical situation of two suspects, who are accomplices in a crime, being arrested by the police. The police have insufficient evidence to convict either, or both, of two suspects of committing a major crime that carries, with the conviction, a 10-year prison sentence. The police separate the prisoners and visit each of them individually to offer the same deal. If one testifies against the other and the said other remains silent, the betrayer (i.e., testifier) goes free and the silent accomplice receives the full 10-year sentence. If both remain silent, each is sentenced to only 1-year in jail for a minor charge. If each betrays the other, they both receive a 5-year sentence. Each prisoner must choose to betray the other or to remain silent; each subject has no way of discovering the other subject's decision before the end of investigation. The question is, given these parameters, how should each prisoner act?

		prisoner A	
		Stay silent	**Betray**
prisoner B	**Stay silent**	Each serves 1 year	A: Goes Free, B: 10 years
	Betray	A: 10 years,B: Goes Free	Each Serves 5 years

Table 1: The Prisoner's dilemma pay off table

A naive interpretation might reason that:

- If I stay silent I *could* go to jail for 10 years
- If I tell on my partner, I *could* go free

However, this assertion does not cover all possible outcomes; specifically, it leaves out the fact that one truly does not know what the other subject will do, and it does not take into account that the other subject's choice has a significant impact on the results.

Let us evaluate the situation from Alex's (one of the prisoner's) point of view. Alex is a mathematician familiar with decision theory fundamentals. He assumes that the potential action of the other prisoner (let's call him Mark) is unknown and unpredictable; consequently, from Alex's perspective, Mark is equally likely to stay silent or to betray. By following the above logic, Alex's choices imply the following potential outcomes:

- ✓ If Alex stays silent:
  - 50% chance that he serve 1 year (if Mark stays silent as well)
  - 50% chance that he serve 10 years (if Mark rats him out)
- ✓ If Alex betrays Mark:
  - 50 % chance that he go free (if Mark stays silent)
  - 50 % chance that he serve 5 years (if Mark betrays him as well)

His purely mathematical solution by applying percentages to the outcomes will be as followed:

Stay silent:

$$(50\% \times 1 \text{ year}) + (50\% \times 10 \text{ years}) = 5.5 \text{ years}$$

Betray the other subject:

$$(50\% \times 0 \text{ year}) + (50\% \times 5 \text{ years}) = 2.5 \text{ years}$$

When he treats both of Mark's choices as equally possible, betrayal seems the best strategy. If Alex betrays his partner, the two possible outcomes are zero and 5 years, an average of 2 ½ years. Furthermore, betraying Mark allows Alex to take the possibility of 10 years of incarceration out of play; instead, he is looking at a maximum of 5years. If Alex accepts all premises as valid, including that there is an equal chance of Mark betraying him or staying silent, betraying Mark has the best maximum, minimum, and average. From a game theory standpoint, betrayal is a strictly dominant strategy.

However, if Alex looks further, he finds a problem with replacing the inability to predict Mark's decision with a 50/50 chance. In doing so, he treats his accomplice as if Mark will simply flip a coin to make his choice. Assigning an equal chance to Mark's possible decisions does not take in to account the fact that: a) Mark is evaluating the situation just like Alex; b) what Alex might know about Mark; and c) other psychological and sociological elements involved in Mark's decision. The betrayal strategy is in Alex's best interests insofar as it guarantees the lowest sentence and saves him from the worst case of 10 years in prison. What if Mark follows the same logic and betrays him? In this case, they are both going to be sentenced to five years. If Alex accepts the fact that Mark is a reasonable and wise human being, he may have to reanalyze his equation since the favorable average of 2 ½ years may just be a mathematical deception.

If Alex and Mark can read each other's minds, the dilemma is resolved. More realistically, we can consider the possibility that Mark and Alex can use their knowledge of each other to predict the other's decision. If Alex believes that Mark can see the hidden solution of both staying silent, he may stay silent as well. What if Alex knows that Mark is in fact guilty and he is not willing to tell a lie, or what if they have a previous history of conflict where Mark has betrayed him and now Alex is looking for revenge? Furthermore, even if Alex can read Mark's mind, he may still have some other motivation to choose an action that doesn't necessary lead to what decision theory defines as the most desirable outcome.

By definition, a dominant strategy in game theory occurs when one strategy is better than another strategy for one player, regardless of their opponents' strategies and characteristics [53]. In the case of the prisoner's dilemma, betrayal is a dominant strategy because it always minimizes punishment under the assumption that all opponent actions are equally probable; as a matter of fact, two self-interested prisoners who give equal chances to their opponent's decisions would betray each other. To move toward a more believable model of the situation, let us reconsider Alex's method (from game theory) of applying equal probability to each of Mark's decisions without considering Mark's characteristics or thought processes.

Fundamentally, the prisoner's dilemma is about each subject's expectations about their opponents, and how their expected choices affect each other. Table [2] shows the Canonical PD payoff matrix for situations like the Prisoner's dilemma. Suppose that the two players are represented by the colors, red and blue, and that each player can choose to either "Cooperate" or "Defect". In this table T, R, S, P is each player's payoff based on the player and his opponent's decision. If both players cooperate they, both receive the reward, R, for cooperation; if blue defects while red cooperates, then blue receives T while red receives S and vice versa. If both players defect they both receive P. The condition (T>R>P>S) must be held in prisoner's dilemma where T, R, P, and S are 0, 1, 5, and 10 years in prison, respectively. The payoff relationship, $R > P$, implies that mutual cooperation is superior to mutual defection. A decision theoretic agent will always choose to defect because T>R and P>S [51]. The nuanced challenges posed by the prisoner's dilemma have sparked interest in its application to economics, politics, and biology among other fields [51]. We chose the prisoner's dilemma to assess our model because of these interesting dynamics, and because it is a well-researched, well-understood problem.

	Cooperate	Betray
Cooperate	R,R	S,T
Betray	T,S	P,P

Table 2: Canonical PD payoff matrix

If the agent can plan and behave believably according to its attributes, environment, and prior knowledge in the prisoner's dilemma, then it stands to reason that it can apply the same processes in other similar cases. We are using the prisoner's dilemma as an example problem to demonstrate (a) the validity of our architecture when applied to reasonable human characteristics; and (b) the flexibility of our architecture to model a wide range of personalities and social relationships.

In summary, naively applying classic decision theory in ignorance of both a model of the accomplice and the personality of the decision-maker is neither logical nor believable.

First of all, accounting for the accomplice's decision-making process makes the agent's decision-making process better informed and more logical. Second of all, accounting for factors peculiar to the decision-maker's personality and context (i.e., personality traits and active roles with respect to the accomplice) allows for a more believable decision-making simulation.

4.3 Preparing for the scenarios

In previous sections, we described the prisoner's dilemma and our model implementation specification. In this section, we take a look at how our architecture model prisoner dilemma. Figure 4.3 shows a schematic view of the agent in prisoner's dilemma. During the planning process, each available action is assigned a probability based on a comparison between the planning agent's potentially-imperfect TOM profiles of its accomplice, along with the action tags associated with each potential choice of action.

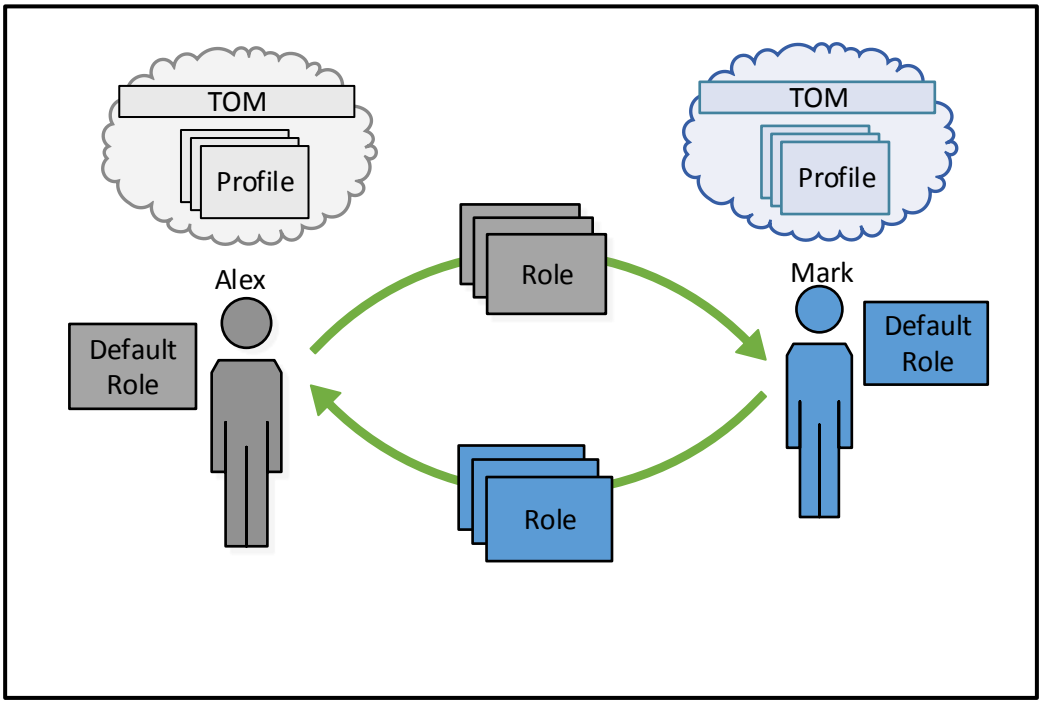

Figure 4.3: Schematic view of agent interaction.

In the computation process for a decision node, the agent has a set of available actions. It evaluates each action individually, based on its active roles by given formulas in Chapter

3 that takes into account action tags, agent roles and associated trait in each role as well as their importance. The weighing process enables the agent to compare actions in the context of their active role and how they enforce the agent to behave in a certain way. Before introducing more varieties into the prisoner's dilemma in our scenarios, we will first explain the planning tree algorithm with the traditional prisoner's dilemma.

There are three main personality traits that we can consider for the traditional version of the prisoner's dilemma to start authoring default roles with:

- Cooperation: In the traditional version of the prisoner's dilemma that has been discussed in Section 4.2, the cooperation attribute, and the fact that it can change the agent's tendency to stay silent or betray, has not been taken into account. In the decision theory solution for the traditional prisoner's dilemma, the agent chooses to stay silent (cooperate) or betray (not cooperate) only based on a probable utility gain. In our discussion in Section 4.2, we described the importance of psychological factors in agent decision making. The level of cooperation between agents is one of these psychological factors that decision theory does not consider in solving the traditional prisoner's dilemma. In the decision theory solution, the agent is neutral with regards to staying silent or betraying the other agent. We will discuss the effect of level of cooperation on the agent's decision shortly, in the evaluation of scenario 1. For now, from the personality perspective, we can consider that a higher level of cooperation increases the chances that the agent stays silent, whereas a lower level of cooperation makes betrayal a desirable option.
- Care of consequence: Care of consequence (COC) is a context-free personality trait that has been discussed in Section 3.5.2.3. As we have mentioned in chapter 3, the agent applies context-free personality traits during the planning process. COC will be used in leaf utility assignment by applying Formula (3) to moderate the degree to which an agent favors utility for itself over utility for others. In the decision theory solution for the traditional prisoner's dilemma, COC is zero; therefore, the agent does not consider and care for the utility of its opponent. COC aids the agent to be more believable by: a) considering the other agent's utility; and b) the fact that the other agent is also looking to maximize their utilities

- Utility Consideration: Utility consideration (UC) is another context-free personality trait that determines a margin in which the agent is willing to dismiss its principles in an exchange for better pay-off. A high level of UC suggests utility-driven agents behave similarly to as in the decision theory approach, i.e. that they will always choose the highest utility. On the other hand, a low UC prevents the agent from deviating from the dictates of its other personality traits, even if small deviations can change their utility significantly. Both a very high and a very low UC make the agent's decision-making predictable because it becomes either utility-driven or it follows rigidly-embedded personality traits. In a more complicated problem, with a denser planning tree, it can change the agent's behavior in various interesting ways. Our planning structure is unique in facilitating this factor to make action selection less predictable in a non-arbitrary fashion.

  It is very important to note that utility consideration is different from care of consequence. Care of consequence affects consideration of facts in leaf nodes; it determines the degree to which other agents' attitudes towards facts are considered when the leaf node is visited by the planning agent (i.e., proportion of personal- and other-desirability). Utility consideration affects computed values at decision nodes. It determines the degree to which behavioural personality traits "override" the computed utility of descendent nodes; (i.e., proportion of behavioural personality and computed utility from descendent nodes). In the decision theory approach, the agent is always looking for the highest possible utility. Looking for the highest possible utility makes the agent always choose the action which brings them the highest possible utility without considering action based on their personality traits.

In our model, the agent uses its traits, utility, and probability of other actors' expected behavior to analyze each state. In the following paragraphs, we show how each of the mentioned traits can affect the planning agent's decision-making process.

4.3.1 Leaf utility assignment

Self-Desirability for each leaf from Alex's perspective (one of our agents) can be calculated based on Formula (14), where $y_{p,A}$ and $y_{f,A}$ are the number of years that Alex may be sentenced to prison or go free in the sequence of actions in the runtime which leads him there. $v_{p,A}$ and $v_{f,A}$ are values of each year of staying in prison or being free, from Alex's perspective. For example in L1 in Figure 4.4 where Alex and Mark both stay silent, $y_{p,A}$ is 1 and $y_{f,A}$ is 9.

Formula (14):

$$SD_A = (y_{p,A} \times v_{p,A}) + (y_{f,A} \times v_{f,A})$$

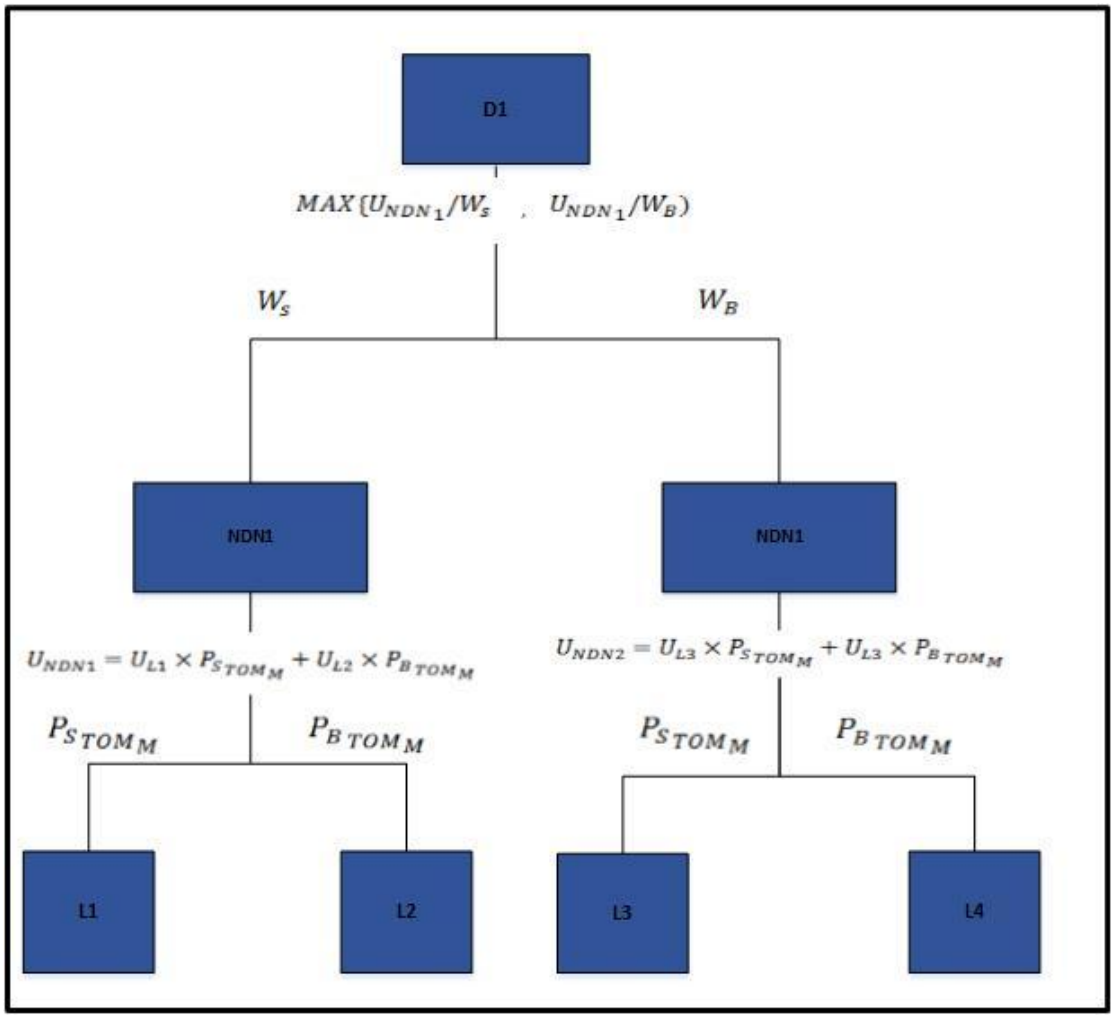

Figure 4.4: Alex planning tree in prisoner's dilemma

Other desirability's for each leaf from Alex's perspective can be calculated based on Formula (15) ; this describes Alex's evaluation of Mark's situation, where $y_{p,m}$ and $y_{f,m}$ are the number of years that Mark may be sentenced to prison or go free, respectively. As we have described in chapter 3, Alex uses his own reward system to evaluate the desirability of the situation from Mark's perspective; in this formula, this quality has been represented by using $v_{f,A}$ and$v_{p,A}$.

Formula (15):

$$OD_A = ((y_{p,m} \times v_{p,A}) + (y_{f,m} \times v_{f,A}))$$

Alex then calculates the desirability of each leaf by applying SD_A and OD_A to Formula (16). This formula uses Alex's care of consequence trait. In decision theory, COC is always 0, making the agent to not take into account the other agent's situation. Humans take into account the consequences of their actions on other peoples' situations even when they do not have any social ties with them. Our model proposes the COC personality trait as an easy solution to providing this quality in agent decision-making.

Formula (16):

$$U_{L_i} = (1 - COC) \times (SD_A) + (COC) \times (OD_A)$$

4.3.2 Non Decision Node

Alex uses Formula (17) with the level of cooperation in his TOM profile of Mark to calculate the probability of Mark's being cooperative or staying silent. In the prisoner's dilemma, m, or the number of available actions in a non-decision node, is 2. Formula (17) is based on procedure that has been explained in the non-decision node of the planning tree in Section 3.7.2.2. In the prisoner's dilemma, it calculates$P_{S_{TOM_M}}$, or the probability of Mark staying silent according to his profile in Alex's TOM (TOM_M), as discussed in Section 3.7.2.2. The Formula (17) checks through all available traits that could be matched between available actions (being silent or betraying) and Mark's profile traits (T_i). It does the same for $P_{B_{TOM_M}}$, or the probability of being betrayed by Mark. Because the prisoner's dilemma has a symmetric tree, $P_{S_{TOM_M}}$ and $P_{B_{TOM_M}}$ in node 2 and 3 in Figure 4.4 will be the same; however, our planning approach can be applied to asymmetric trees as well.

Formula (17):

$$P_{S_{TOM_M}} = \left(\frac{\sum_{i=1}^{n} 1 - abs\left(T_i\left(M_{TOM_A}\right) - T_i(\alpha)\right)}{\sum_{j=1}^{m}\sum_{i=1}^{n} 1 - abs\left(T_i\left(M_{TOM_A}\right) - T_i(\alpha)\right)} \right)$$

Alex then calculates the utility of non-decision nodes by applying the probability of each branch to its associated leaf utility U_{L_K}Formula (18):

Formula (18):

$$U_{NDN} = \sum_{k=1}^{2} \left(\frac{\sum_{i=1}^{n} 1 - abs\left(T_i\left(M_{TOM_A}\right) - T_i(\alpha)\right)}{\sum_{j=1}^{m}\sum_{i=1}^{n} 1 - abs\left(T_i\left(M_{TOM_A}\right) - T_i(\alpha)\right)} \right) \times U_{L_K}$$

4.3.3 Decision Node

In the decision node, Alex uses his personality traits to find the weight for staying silent(W_s) or betrayal (W_B) by applying his personality traits to Formula (13) . In the absence of the utility consideration trait, Alex chooses an action with the lowest weight that will take him to the highest utility Formula (19). W_i is action weight-based on Alex's roles and U_{NDN_i} is the utility of its decedent node.

After calculating the desirability of all actions available to them, the agent finds the ground ratio by dividing the utility associated to the most desirable action by its weight, as discussed in Section 3.7.2.3. In general, the agent follows its personality; deviation from the standard is only worthwhile if it causes a significant shift in the agent's utility. The ground ratio is the deviation of the utility that the agent can gain by following the closest action to its personality. All other actions have a higher weight but, if one of these less desirable actions based on the agent's personality could bring it a significantly higher utility, the agent will follow that one. The agent uses the ground ratio to find out if the option of gaining a significantly higher utility by performing a less desirable action is available or not.

Formula (19):

$$MAX(U_{NDN_i}/W_i)$$

4.4 Scenarios

In this section, we apply our model to test the agent's decision making in a different variation of the prisoner's dilemma. The Authoring System enables us to create interesting new scenarios by only assigning multiple roles to Alex and Mark (our agents). Scenario 0 demonstrates simulation of decision theory with our agent default role. Scenario 1 deals with the main plot of the prisoner's dilemma; Alex and Mark, who have not met each other before, use their stranger profile TOM and default roles for making their individual decisions. Scenario 2 demonstrates our model's ability to make agent planning be situated in the social context by introducing a friend role. Scenario 3 demonstrates the quality of asymmetric social roles when Alex treats Mark as an enemy but Mark considers him as a friend. Finally, scenario 4 simulates an interesting love triangle that ends with a scarification of one edge to protect his ex-girlfriend and betrayal of his opponent. In the process of evolving from the traditional scenario to more complicated ones, the Authoring System changes agent internal data, such as active roles and TOM profile. By changing how the agent perceives the world and what its priorities are, the system can simulate more complicated cases. Note that the flexibility in our design facilitates assigning new and multiple roles to the agent without making any change in the agent planning process

4.4.1 Scenario 0

In Scenario 0, we present our model's success in modeling the decision theory approach with our proposed planning tree. The assignments for variables in the planning tree are as follows:

- Cooperation: The agent is neutral towards being cooperative or not. That means that staying silent(W_s) or betraying the other agent (W_B) have equal weight. This makes the agent choose one option based only on which non-decision node has a higher utility.
- UC: The decision theory approach only considers probable utility in decision making without taking into account the agent's personality. The agent UC is zero.

- COC: In the decision theory approach, the agent never considers the other agent's utility in leaf nodes, making the agent COC trait equal to zero.
- $P_{i_{TOM_M}}$: The decision theory approach does not suggest any practical solutions to predicting the other agent's decision.

Considering these variables, the agent who follows the decision theory approach makes a decision based on comparing the results of the below equations from each of non-decision nodes:

$$U_{NDN_1} = \left(P_{S_{TOM_M}}\right) \times ((1 \times v_L) + \left(9 \times v_p\right)) + (1 - P_{S_{TOM_M}}) \times ((0 \times v_L) + \left(10 \times v_p\right))$$

$$U_{NDN_1} = \left(P_{S_{TOM_M}}\right) \times ((10 \times v_L) + \left(0 \times v_p\right)) + (1 - P_{S_{TOM_M}}) \times ((5 \times v_L) + \left(5 \times v_p\right))$$

Due to the fact that v_L is positive and v_p is negative, even with $P_{S_{TOM_M}}$ equal to 1, the agent betrays its opponent. This decision is due to the variable assignment in decision theory and the inflexibility that these values impose on the agent decision. The description above shows our planning process' success in simulating agents that can act within the decision theory approach's predictions. In the following section, we describe our model's flexibility in allowing for the creation of new situations through assigning new roles to the agent. A new role can create a new personality or social context for the agent. The planning tree uses the agent's role's preferences and unique attributes to choose the best action.

4.4.2 Scenario 1

The plot in the first scenario is same as in the traditional prisoner's dilemma: Alex and Mark, who have not met each other before, are arrested, and each one has been offered the same deal. Validity of the TOM module in this scenario will be demonstrated in a simple situation by walking agents through the planning process and explaining how each variable affects the decision-making process. Authoring System partial templates for the

agent default role and the TOM module for the agent have been shown in Figure 4.5. The only constraint in assigning a value to "years in prison" and "years of freedom" has been discussed in Section 4.2 for the general case of the traditional prisoner's dilemma. The pay-off arrangement in the traditional prisoner's dilemma is such that the mutual cooperation receives higher payoff in comparison to mutual betrayal. However, in our model, by adding more roles to the agent, we demonstrate how the classic prisoner dilemma can evolve into more interesting cases. For example: the agent may choose to betray only to punish its opponent (Scenario 3 and Mark in Scenario 4) or the agent may have other interests that makes them choose to stay silent (Scenario 4). In the following paragraphs, we discuss how each variable in the role or TOM can affect agent decision making.

Default Role	
Trait Name	**Trait value**
Care of consequence	
Cooperation	
Utility consideration	
Active Pursue Goals	
years in prison	
years of freedom	

TOM	
Stranger Profile	
Trait	Value
Cooperation	

Figure 4.5: Authoring Tool partial templates for the first scenario

As we have mentioned before, in our role-based architecture, each agent has at least one default role that formalizes the agent general behavior when the action does not involve any of the other active roles' target. Due to a lack of any previous social ties in the traditional version of the prisoner's dilemma, Alex and Mark both use their default roles

in the first plot. Because Alex and Mark do not know each other, they use the 'Stranger' profile in TOM to predict the other agent's actions. Consider a case where Alex has an average level of Cooperation, which means he has no preferences over staying silent or betraying Mark ($W_s = W_B$). This means in applying Formula (19) to choose an action in N1, U_{NDN_i} is the determining factor. If Alex's *stranger* profile does not detect any preference for Mark to either betray or cooperate (i.e., the profile reflects a neutral degree of Cooperation, the same as in the decision theory approach), Alex assumes that each of the two actions has a 50% probability ($P_{S_{TOM_M}} = P_{B_{TOM_M}}$). With equal probability for both of Mark's actions, Alex's only determining factor in making the decision is a comparison between U_{L_i}. Interestingly, with any degree of care of consequence less than 0.5, Alex betrays Mark; otherwise, he stays silent. The reason for this fluctuation in his decision making can be seen in Formula (16). Care of consequence being equal to 0.5 means that Alex cares for Mark's situation as much as he cares for himself. Therefore, among leaf nodes, L1 has a significant advantage.

If we keep Alex's 'Stranger' profile the same, by decreasing Alex's care of consequence or decreasing his cooperation level, he betrays Mark. A higher degree of cooperation from Mark results in $P_{S_{TOM_M}}$ being greater than$P_{S_{TOM_M}}$; consequently, N1 (in Figure 4.4) receives a higher weight in the utility calculation of NDN1 (same as N3 and NDN2). By considering cooperation from Mark, Alex is basically choosing between 1-year in prison or going free. If Alex has a low degree of care of consequence, he will betray. On the other hand, by considering the negative value of a 10-year prison sentence for Mark, Alex may stay silent. The description of the process explains that the close value of $P_{S_{TOM_M}}$ and $P_{S_{TOM_M}}$ does not convince Alex to stay silent. With the close value of $P_{S_{TOM_M}}$ and $P_{S_{TOM_M}}$ based on Alex's TOM, he would stay silent only if he is cooperative or has a high level of COC.

This simple example illustrates how modifying roles and traits through the Authoring System offers the potential to define more interesting characters. For example, increasing or decreasing Alex's cooperation—his willingness to work in collaboration with Mark—

will influence Alex's decision, just as his impression of Mark's cooperation will also exert an influence.

With TOM modules in place, we can enhance our agents further by fine-tuning their personality through their default roles. A high level of cooperation in Alex causes him to cooperate with Mark even if his TOM profile of Mark has a relatively low cooperation value. This means, in comparison to the neutral role, now Alex may stay silent despite expecting a lower level of cooperation in his stranger profile. Intuitively, a lower level of cooperation in Alex's personality makes him prone to betray unless a very high level of cooperation is present in his stranger profile (close to 1).

We only test Utility Consideration in the first scenario. If we assign it to zero, this causes Alex to not deviate from his personality in any circumstances. A lower level of the cooperation trait makes him betray, whereas a higher level of cooperation leads him to stay silent without considering either of the TOM prediction or utility assignment. On the other hand, the maximum value of utility consideration makes Alex act in the path that leads to highest utility; he takes into account TOM and still use care of consequence in the leaf utility assignment. If TOM predicts cooperation and care of consequence as not being close to 1, Alex will stay silent; otherwise, he will betray as well.

The utility consideration trait has a very high potential to create interesting behavior in agents. However, in the next few scenarios, we will not consider it anymore because we already know how it affects agent decision making. All agents in the next few scenarios use the best ratio approach to choose their action. In a denser tree with various branches, utility consideration has a high potential to make agents less predictable .The first scenario clearly demonstrates our success in implementing TOM and using it in agent decision making. Furthermore, the decision tree traversal algorithm uses all of the factors to determine the final decision.

Figure 4.6 represents a diagram with two sets of data. In each data set the agent COC is constant and the agent has been tested with different degree of cooperation and expected cooperation from its opponent. Blue and red dots in the diagram recorded the threshold that the agent switches from staying silent (cooperative action) to betray (non-

cooperative). With the same COC all the points above blue /red line means the gent would stay silent whereas points above the colored line means the agent would betray. By decreasing COC, moving from the blue line to the red one, we can see that the same Cooperation level demands a higher expected cooperation from the opponent to makes Alex stay silent; Note that the purple point on the red line has a higher level of TOM in comparison to purple point on the blue line. Comparing the cooperation level of two yellow points one on the blue line the other one on the red one, also suggests that with the same level of expected cooperation, Alex with lower level of COC ,yellow dot on the red line, should be more cooperative to stay silent in comparison to the yellow dot on the blue line.

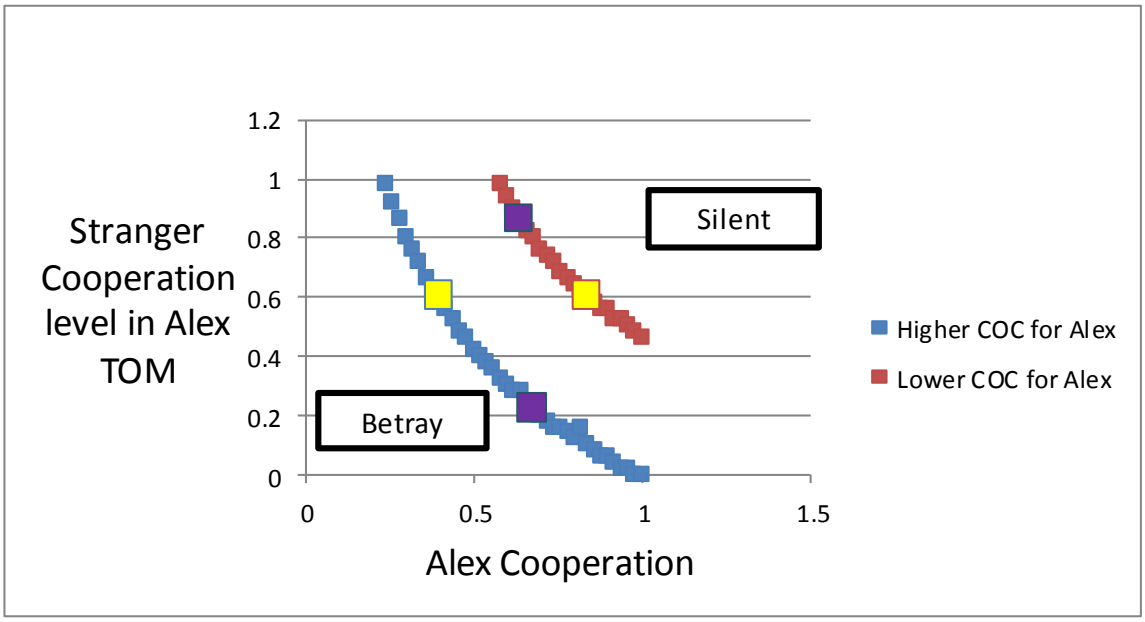

Figure 4.6: COC impact in Scenario 1

The diagram in Figure 4.7 represents the agent threshold for the lowest expected TOM and cooperation level to stay silent by possessing same level of COC. In this diagram the agent COC has been kept the same and line represents the agent threshold for changing its decision from staying silent to betray its opponent. Comparison between red and purple dots illustrates how a more cooperative agent may stay silent even with a low level

of expected TOM from its opponent; whereas a less cooperative agent (purple point) has a higher threshold for expected cooperation from its opponent to stay silent.

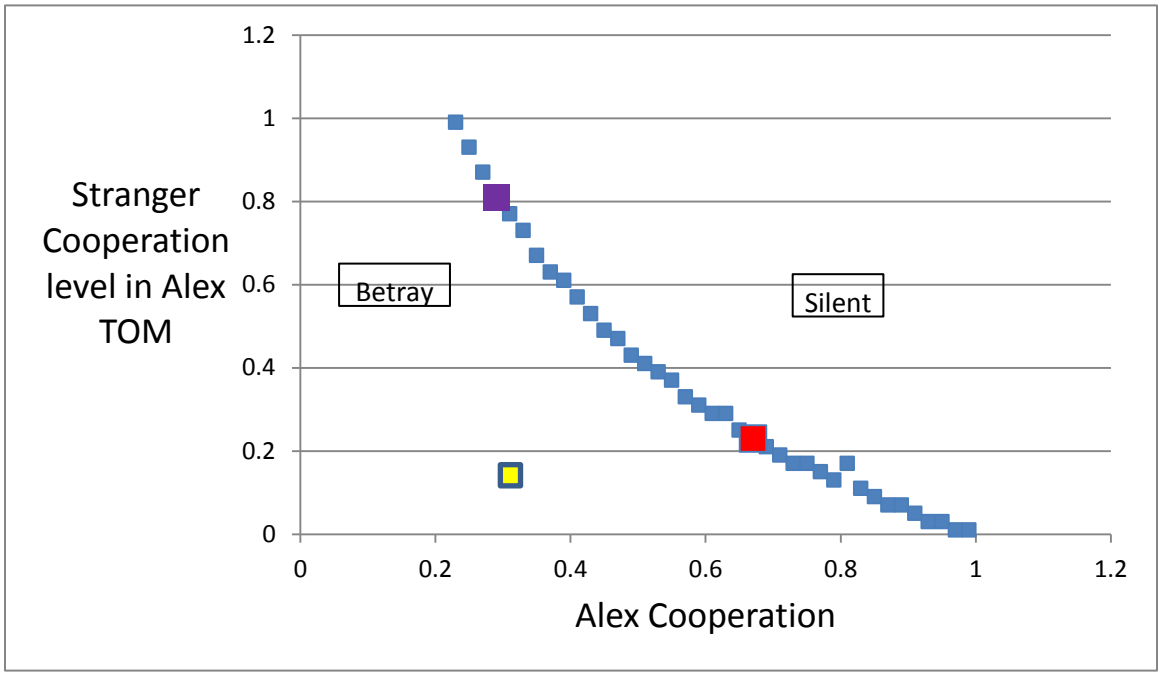

Figure 4.7: TOM versus Action tendency in Scenario1

4.4.3 Scenario 2

In the second scenario, we introduce the friend role to the implementation; each agent has a friend role towards the other one which enables the testing of more traits. The friend role can affect decision making whenever an action involves its target. Alex can have two different values for his cooperation trait: one for the default role, and the other in his friend role. He can be generally defined as a non-cooperative person, although he may be willing to cooperate with his own friends. Figure 4.8 represents a partial template for roles and variables involved in the second scenario. In the new scenario, although the agent still has a stranger profile, he will not use it due to the fact that a more accurate model of Mark has been provided.

Defult Role	
Trait Name	Trait value
Selfishness	
Cooperation	
Honesty	
Utility consideration	
Active Pursue Goals	
years in prison	
years of freedom	
Role importance	
value	

Friend Role	
Trait Name	Trait value
Cooperation	
Loyalty	
Interest Basde Goal	
Fact and its state	value
Betrayed(Mark,Alex)=True	
Betrayed(Mark,Alex)=False	
Betrayed(Alex,mark)=False	
Betrayed(Alex,Mark)=True	
Target	
Mark	
Importance	
value	
Affinity	
value	

TOM	
Stranger Profile	
Trait	Value
Cooperation	
Mark Profile	
Trait	Value
Cooperation	
Loyaty	

Figure 4.8: Authoring Tool partial roles and TOM profiles for Alex in the second scenario

The friend role changes leaf evaluation as well. When Alex is calculating leaf utility because of his friendship with Mark, he cares more about his probable state. Our system provides this consideration by using their friendship affinity level ($A(T(F_i))$) in Formula (6) from Chapter 3. Friendship enforces a couple of interest-based goals for Mark and Alex: now, Alex has a negative value for betraying Mark, or being betrayed by him. Following the same reasoning, he has a positive reward for not betraying or not being betrayed. Recently added goals and affinity levels change leaf utility order. In the current scenario, even with a low level of care of consequence, L1 still has a higher utility in comparison to others because neither Alex nor Mark has been betrayed. On D_1branches, due to a higher level of cooperation in Alex's role towards Mark, W_s is smaller than W_B. Not surprisingly, Alex also expects more cooperation from Mark, as a friend, meaning that $P_{S_{TOM_M}}$ is greater than $P_{B_{TOM_M}}$. Consequently, when considering all factors

involved in the scenario, there is no wonder that both Alex and Mark stay silent.

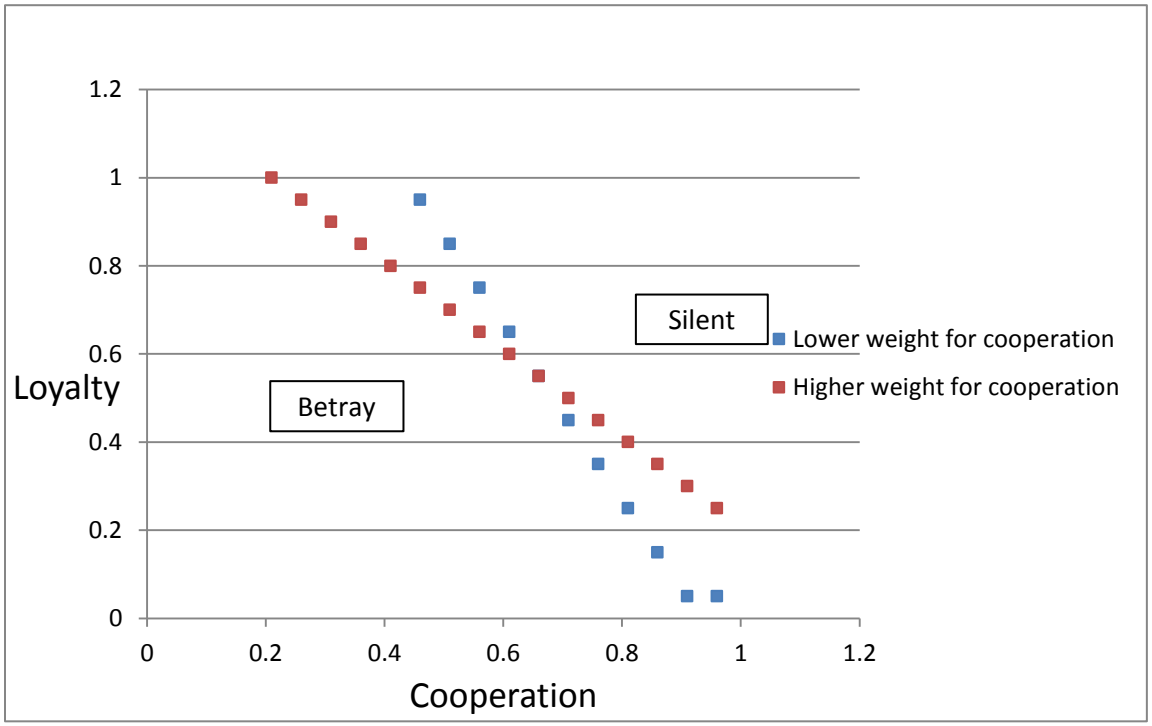

Figure 4.9: Trait and role importance changes the agent decision in scenario 2.

In the diagram in Figure 4.9 blue dots represent the case that agent has equal weight for cooperation and loyalty based on its role. Red dots represent higher weight for cooperation trait in comparison to cooperation during the test. Any dots above each colored line mean that the agent would stay silent, whereas points under the line make the agent to betray its opponent. The friend role determines high level of cooperation and loyalty whereas enemy role makes their values small. As we can see by adding loyalty trait, now the agent can even cooperate with low level of cooperation, and vice versa.

We can add even more complexity to the traditional prisoner's dilemma by introducing a scenario wherein two agents are friends and one of them has actually committed the crime, and the other one knows about it. This new factor does not change the NDN1 and NDN2 utility in Figure 4.4. However, W_s and W_B are different when it is taken into account that Mark is actually guilty of committing the crime and there is an honesty trait in Alex's default role. This new modification in the scenario creates a new dilemma for the agent: if Alex tells the truth that means betraying a friend, which is in contradiction

with his cooperation level in a friendship despite having been approved by the honesty trait contained within his default role.

There are couples of factors that help us analyze Alex's decision: the proportional importance of the friend role and default role; trait weights for honesty and cooperation; and, finally, how much he cares about not betraying Mark. For a loyal friend with a high importance for friendship, it would presumably be hard to betray a friend. The point in introducing the honesty factor is that there is now the possibility of a friend betraying their peers, *even* with their possessing a high level of cooperation. Assigning more weight to the honesty trait and default role in comparison to those of cooperation and friendship results in Alex testifying against Mark. This is interesting because this creates a conflict within Alex's thought processes. People often find themselves in conflicting situations. The fact that our proposed model facilitates the creation of conflicts, as well as different approaches that an individual character can take to deal with these conflicts, makes our agents more interesting and believable. Different approaches that the agent takes to deal with conflicting scenarios represent: a) more depth in the agent's decision making and underlying psychological reasoning; and b) more depth to their personalities.

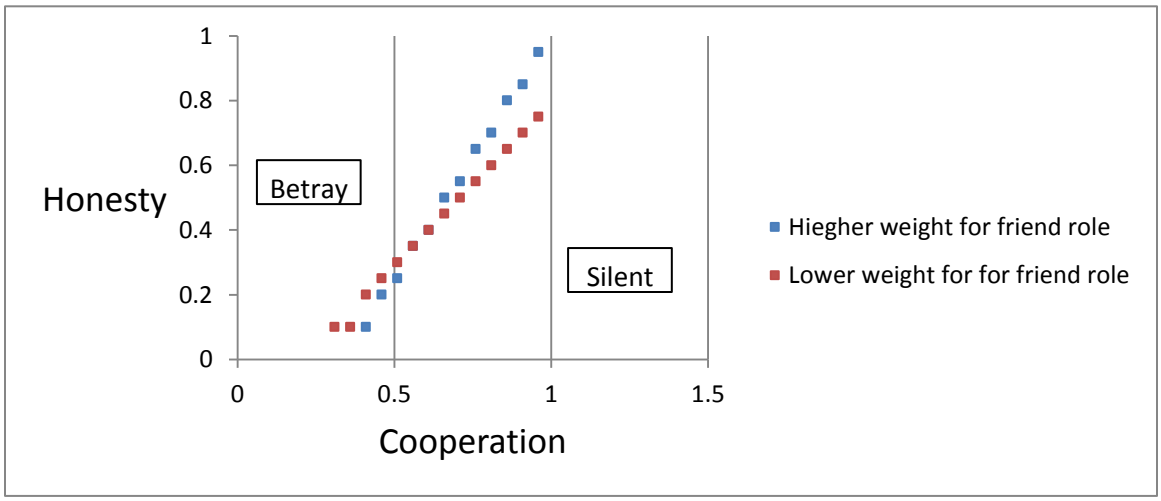

Figure 4.10: Conflict of interest caused by role overload in Scenario 2

The diagram in Figure 4.10 demonstrates when Alex needs to choose between adhering to honesty trait in its default role or cooperation trait in his friend role, when he knows about his friend's crime. Blue dots represent higher weight of friend role in comparison to the default role. The Blue and red line illustrate the agent threshold to stay silent by keeping default and friend role weight constant against each other and changing the honesty and cooperation degree one associated with Alex's default role and the other with his friend role. Any point above each colored line, means Alex betray Mark, his friend, in favor of his default role and honesty trait. In this diagram we can see that if Alex has a higher level of honesty he needs to have a higher level of cooperation to stay silent.

The second scenario clearly explains the encapsulation of personality traits in a role framework. Furthermore, it demonstrates how the role context can affect agent behavior, as in, for example, Alex's dealing with a friend priority in action evaluation with traits that have been defined in friendship (Kind of confusing explanation of the example). The agent is capable of handling multiple roles and, even more interestingly, is capable of using more than one role to evaluate one and the same action from different perspectives. Lastly, it was a good example to illustrate how conflicting goals or traits from multiple roles can create dilemmas in a role-based architecture.

4.4.4 Scenario 3

In the third scenario, Mark keeps his friendship with Alex; however Alex secretly hates Mark. We switch Alex's role towards Mark to 'Enemy', illustrated by a low level of cooperation and a negative affinity towards Mark. There is also a new trait that can be used in this plot, namely Alex's loyalty towards Mark. The new setting enables us to introduce this trait where Alex treats Mark as an enemy; however, Alex knows he will probably stay silent ($P_{S_{TOM_M}} > P_{B_{TOM_M}}$). Therefore, by using Formula (18) to calculate NND1 utility, the utility of L1 has a higher weight in comparison to L2, as well as to L3 in NND2. Alex's interest-based goals are also different from the last case: now, he does not care about being betrayed by Mark. This means no negative valence will be imposed on L2 and L3.

Negative affinity towards Mark changes the leafs' utility order. Once again, adjacent leafs from the betray action have a higher payoff (L3 and L4); with a low degree of loyalty, Alex will definitely betray Mark. With a higher level of loyalty, it will be a battle between loyalty and cooperation weight. On Mark's side, staying silent is still the candidate action; unfortunately, there is a very narrow chance that Alex will not betray him. This means Alex goes free while Mark spends 10 years in prison.

This scenario shows the application of asymmetric social relationships in our system. In our system, roles can be easily created, modified, and applied to the agent. Each role can be customized for the agent and desirable scenario.

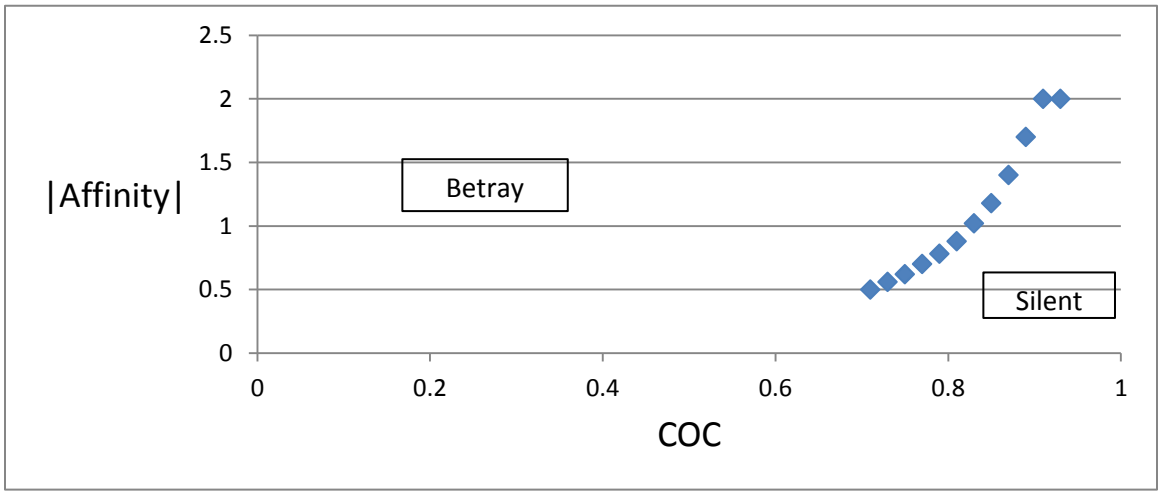

Figure 4.11: COC versus Animosity in Scenario3.

The diagram in Figure 4.11 illustrates relationship between COC, animosity towards the enemy, and their affect in the agent final decision. By keeping Alex's cooperation and expected cooperation from Mark the same, we played around with COC and affinity. In this Scenario, Alex doesn't have an Interest goal to betray Mark instead of that he is using his own belief system to found out about Mark state utility. Due to the fact that Alex affinity towards Mark is negative, any undesirable state for Mark carries a higher utility for Alex. On the other hand, Alex knows that with a high probability Mark will stay silent. With Low COC Alex betrays Mark but as COC increases he only betrays Alex if he really hates him.

4.4.5 Scenario 4

The fourth scenario modifies the third one with the introduction of two more new roles. In this plot, both Alex and Mark are aware of a mutual hatred between them, as the two "edges" of a love triangle. Alice, who was previously in a relationship with Alex, is now with Mark. This history motivates us to create considerate ex-boyfriend and jealous lover roles.

Scenario 3 can be the simple version of the fourth one, where Alex does not care about Alice anymore but he wants to take revenge on Mark. We can assume that, although Alex loved Alice in the past, he does not care about her anymore and he feels animosity towards her partner (Mark) now. This situation gives us same result as Scenario 3. In this case, Mark may know about the whole situation and still treat Alex as a friend. This would cost him 10 year in prison due to Alex's betrayal (Scenario 3).

In the second case, we can assume that Alex, despite his animosity towards Mark, still cares about Alice's feelings. He knows that by betraying Mark, Mark would either be sentenced to 5 or 10 years of prison; if he stays silent it will be either 0 years or 1 year, which brings less grief to Alice. This time, Alex considers the fact that more years in prison for Mark means a longer period of grief and sadness for Alice, whom he still has a positive affinity towards. This can be represented in our model by an interest-based fact that represents a negative value for Alice's grief. These considerations, once again, shift the leafs' utility ranking to have a higher utility for L1 and L2 and, consequently, a higher utility for NDN1 in comparison to NDN2. Although in Alex's decision node, $W_s < W_B$, the propagated utility from leafs and their presentation in U_{NDN_1} and U_{NDN_2} makes the decision a battle between how much Alex cares about Alice's not suffering grief in contrast to his own default role's goal, namely to spend as few years in prison as possible. Surprisingly, Alex may stay silent despite a low level of cooperation.

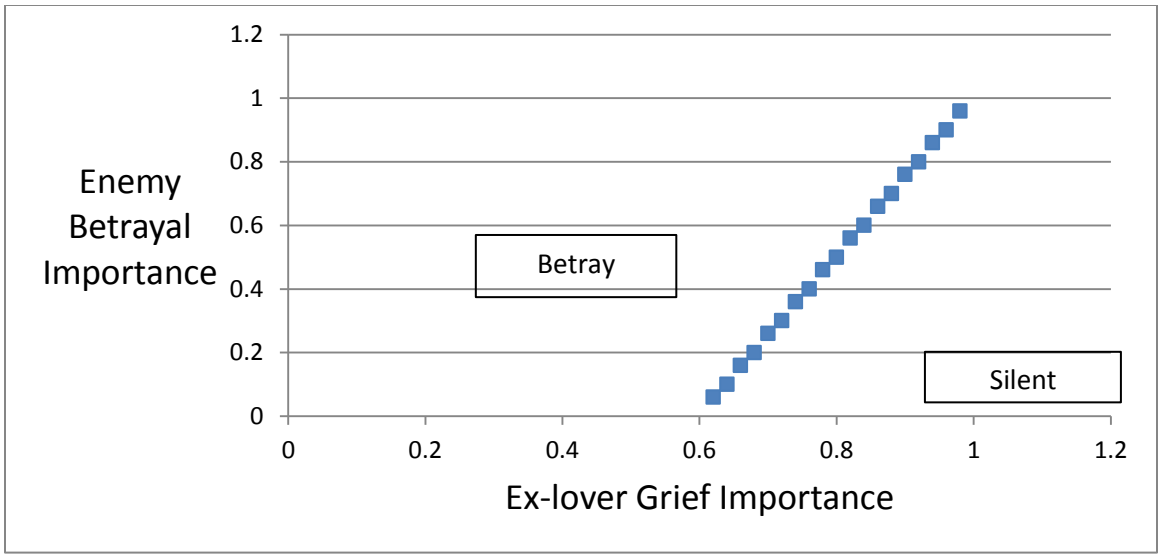

Figure 4.12: Conflict of interest in Scenario4

The diagram in Figure 4.12 demonstrates the battle in Alex mind to betray Mark or stay silent. Matter of betraying the other side of triangle is a battle between the degrees that Alex cares for his Ex-lover grief and how much he wants to see his enemy (Mark) misery. The blue line demonstrates the agent threshold for staying silent by weighting these two variables against each other. Any point above the blue line represents the agent betrayal. Betrayal determines the agent higher value towards seeing his enemy misery despite the fact that he is aware of ex-lover grief.

On Mark's side, his 'Jealous' role towards Alex makes him develop a very negative utility against any state that provides his opponent fewer years in prison than he has to endure. He has a low level of cooperation; he also gives a high probability to Alex's betraying him. His negative animosity is an additional factor to lead him to betray Alex. This scenario ended up in favor of jealous Mark going free, and Alex spending 10 years in prison.

4.5 Summary and Discussion

Scenario 0 to 4 examined consider some possible variations to the traditional prisoner's dilemma. The Authoring Tool facilitates the addition of more varieties by allowing for the design of new roles, traits, and facts. Although we only tested the proposed model on

one scenario with different variations, our planning algorithm and agent architecture are general enough to perform within any given scenario and with any role specifications.

The TOM module and its emergent application in the planning tree is one of the main contributions in our design. In scenario one, the assumption that the other person is a stranger enables the designer to test the agent's decision-making with different stranger profiles. If Alex is not biased towards being cooperative, he does not have any tendency towards being silent or betraying. The matter of staying silent or betraying will be determined by Alex's stranger profile and also his care consequence that has been determined by the care of consequence trait. A low level of care of consequence allows Alex to makes his decision only based on his own utility without considering the other peoples'. With a low level of care of consequence, the agent's decision is asymmetric to the decision theory approach. However, by tuning the care of consequences to have a higher level, the agent will follow the other agent's decision. As long as care of consequences does not cause the agent to blindly look for his optimal utility, he will follow his TOM.

From a psychological stand point, care of consequences enables the agent to make a more believable decision by considering the existence of other agents and these other agents' preferences. Although our current prototype does not support recursive TOM, the fact that the agent considers other agents outside his social network enables him to make more situated decisions. One of the objections to the decision theory's solution for the prisoner's dilemma is its failure to consider that the other prisoner is also seeking the highest payoff. Our TOM, care of consequence trait, and their application in the planning tree, enables the agent to perform more believable behaviors by considering other agents.

The first scenario provides a straightforward test bed to check the functionality of TOM. Our TOM module affects the agent's decision-making when the agent is not biased towards cooperation or does not have a very low level of care of consequence. This is parallel to human behavior: as long as one personality trait is not dominant, all of the other factors will also play a role in decision making.

In Scenario 2, our role components and their application in the planning tree ensures that the agent is acting according the role context by:

- Considering role target in state evaluation: Unlike in Scenario 1, the other prisoner involved in Scenario 2 is a friend who the agent has a role and affinity towards. This changes the state utility calculation. The consequences of the agent's action will be evaluated regarding to what may happened to his friend.
- Behavioral Personality Trait: the agent considers specific personality traits from its friend role toward the other agent. This makes the agent follow a consistent pattern of behavior that has been defined by the active role's personality trait.
- Interest Goal: Defining interested-based goals to avoid betraying a friend is another mechanism that makes the agent considers their social context.

In the second part of Scenario 2, we add another variation, where the agent has to make a decision between adhering to his cooperation level or his level of honesty in the default role. This variation represents our agent model's success to use all active roles in the agent's decision-making process. This variation, and the agent's decision being based on the relative importance of the cooperation in default role and honesty in friend role, is a good demonstration of the role's architecture potential to create a dilemma-based scenario. Dilemmas could be created for the agent by providing contradicting components in different roles and assigning them to the same agent.

The third and fourth scenarios demonstrate the flexibility of our model to change the agent's decision by changing its role. In scenario 3, Alex has a negative affinity towards Mark but he knows that Mark will cooperate. Mark's profile in Alex's TOM is the same as in Scenario 2, but Alex's negative affinity towards Mark, low level of cooperation, and lack of interest-based goals, causes him to betray Mark.

Scenario 4 represents a battle between action weight and utility. Although Alex is not cooperative, he stays silent. The agent makes this decision in favor of preventing to cause his ex-girlfriend sadness. This scenario clearly demonstrates the role-based architecture to make the agent make unpredictable but, at the same time, allowing the agent to perform well-reasoned decisions. If Alex purely makes his decision based on his

cooperation level, this decision would have stayed the same. However, in our planning algorithm, personality traits are only one factor that causes him to perform a different action in a different situation with the same set of available actions.

Being in a social context can affect the agent's decision and behavior. In this sense, our role model and its integration with the planning tree enables the agent to consider their social context in all planning steps. Being in a certain social context affects: a) the agent's action tendency (For example: Alex, with the same default role, chooses a different action in Scenario 1 and 2.); and b) The agent's perspective of the word state; our architecture facilitates this by introducing interest-based goals (For example: Alex's leaf utility assignment in scenario 2 and 3 are significantly different.); and c) The agent's final decision, even with same action tendency (For example: in scenario 4, Alex has a low level of cooperation but the utility calculation makes him cooperate.).

In conclusion, the role architecture and its components, such as goals, traits, and the reward system, add layers in the agent's decision-making. This layered behavior situates our agent in their social context. More importantly, because the agent's decisions involve many factors, it is hard to predict. By comparing the results from all of the scenarios, we can assert that the agent's decision is not based on one single attribute or quality. The fact that the agent is cooperative does not make him act cooperatively in all given situations. Predictability, as discussed in chapter 2, is one of the main problems with current NPCs. The agent's planning consists of several atomic elements that each play a role in the decision making process.

5 Conclusions

In conclusion, the design presented in this work addresses the problem of believability in human-like agents that were introduced in Chapter 2. One of the main problems in classic AI is its rigid structure. Making changes in the behavior of agents created with classical AI requires modifying scripts, and as the amount of code increases, keeping track of changes becomes problematic.

5.1 Summary

Four test scenarios in Chapter 4 thoroughly represented our model's potential to add more variety to the agent without undergoing a change in the structure of the agent model. The agent was situated in the social context with role components such as belief, personality traits and goals. By changing these components the designer can create a new role and expect a different behavior. Our role-based architecture is expressive; assigning more than one role to the agent can easily enrich the character with multiple facets based on how the character relates to others. The planning process is novel in considering the agent roles' components in all steps. During the planning process the agent weighs both the context and its own preferences in pursuit of its goals. The agent utility assignment has the ability to perceive others' utility through the agent reward system. Planning not only considers other agents' utility but also takes into account the effect of other actors' decisions. The TOM module enables the agent to take into account the decisions of other agents. This is another factor that enables the agent to be more socially aware by: a) considering world facts based on its role, b) considering other actors decision. Due to the fact that the planning tree focuses on the agent's active role and evaluates action and states according to all related roles' attributes, the agent's decision is not easily predictable. The proposed TOM module and its application are general enough that it can be used in any scenario for any kind of personality. The current implementation can provide the agent with one facet of contextualized rational decision-making. The agent can consider multiple facts, roles and traits that can all determine or change the planning evaluation during the decision making process to some degree.

5.2 Contributions

Our proposed model in particular has made the following contributions:

- The Role based architecture ensures that the agent performs coherent behavior in one specific role. The agent roles define how it perceives events and world facts, which affects its decision making.
- One of the main factors in the agent decision making is considering other agents in its planning by means of the TOM module.
- Our planning architecture takes into account all of the agent's roles' components. The planning module guarantees that in all steps the agent evaluates states or actions based on its active roles.
- Being able to cope with nondeterministic situations makes the agent more believable. The role based architecture enables the user to author their desired agent with customized qualities.

5.3 Future Work

This architecture provides many avenues for future work, including:

- TOM: Expanding the TOM module so that it can learn during runtime using machine learning algorithms. Our TOM module is unique in that it uses the profile of other agents in order to predict their actions. However TOM module only performs one level of recursion, so we leave the higher levels of implementation of theory of mind to future works. Higher levels of theory of mind can construct the complete planning trees of other agents and therefore come up with a more accurate prediction.
- Integrating Emotion: Integrating emotion with the planning tree is another interesting area that can be investigated further. The agent's Emotional State can affect its decision making; this can be reflected by mapping Emotion Traits to specific behavior or traits. The agent in who is enraged can use this personality trait instead of its active roles personality traits. For example, an agent that is angry has a

higher tendency to act as an aggressive person even though its personality is not aggressive.

- Memory: Memory module can also be optimized for agents that need to perform in real-time. Our current Memory structure can be optimized by putting limits on the number of Memory Cells. Introducing heuristics that discard events not only based on their chronological order but their importance from the agent's perspective is another interesting area of research that can be further developed.
- Integration in a Real Game: Deploying the agent model as NPC in a real game through observing its integration in the game environment with complicated scenarios and more agents involved has been left for future work as well. This implementation enable user testing that has been mentioned earlier.
- Other Module: Implementing the complete emotion, planning tree generation, learning aspect of TOM and intention recognition has been left for future work as well.

6 Refrences

[1] Charness, Gary, and Matthew Rabin. "Understanding social preferences with simple tests." The Quarterly Journal of Economics 117, no. 3 (2002): 817-869.

[2] Silverman, Barry G., Michael Johns, Jason Cornwell, and Kevin O'Brien. "Human behavior models for agents in simulators and games: part I: enabling science with PMFserv." Presence: Teleoperators & Virtual Environments 15, no. 2 (2006): 139-162.

[3] Bacon, Timothy J., Philip Jones, Randall B. Garrett, and Andreas Tolk. "Integration of psycho-social models and methods in NATO's approach to operations; a review of NATO research and technology organization (RTO) systems analysis studies (SAS-074)." In Winter Simulation Conference, pp. 2852-2859. Winter Simulation Conference, 2009.

[4] Schmorrow, Dylan, Gary L. Klein, Robert Foster, John Boiney, Sean Biggerstaff, Paul R. Garvey, Matt Koehler, and Barry Costa. Applied Use of Socio-Cultural Behavior Modeling and Simulation: An Emerging Challenge for C2. Office Of The Deputy Under Secretary Of Defense (Science And Technology) Rosslyn VA, 2009.

[5] Durupinar, Funda, Jan Allbeck, Nuria Pelechano, and Norman Badler. "Creating crowd variation with the OCEAN personality model." In Proceedings of the 7th international joint conference on Autonomous agents and multiagent systems-Volume 3, pp. 1217-1220. International Foundation for Autonomous Agents and Multiagent Systems, 2008.

[6] Mateas, Michael, and Andrew Stern. "Façade: An experiment in building a fully-realized interactive drama." In Game Developers Conference, Game Design track, vol. 2, p. 82. 2003.

[8] Si, Mei, Stacy C. Marsella, and David V. Pynadath. "Thespian: An architecture for interactive pedagogical drama." In Proceeding of the 2005 conference on Artificial Intelligence in Education: Supporting Learning through Intelligent and Socially Informed Technology, pp. 595-602. 2005.

[9] El-Nasr, Magy Seif. "Interaction, narrative, and drama: Creating an adaptive interactive narrative using performance arts theories." Interaction Studies 8, no. 2 (2007): 209-240.

[10] Barber, Heather, and Daniel Kudenko. "A user model for the generation of dilemma-based interactive narratives." Proceedings of the AIIDE 7 (2007): 13-18.

[11] Mateas, Michael, and Andrew Stern. "Structuring content in the Façade interactive drama architecture." Proceedings of artificial intelligence and interactive digital entertainment 3 (2005): 93-98.

[12] McCoy, Joshua, Michael Mateas, and Noah Wardrip–Fruin. "Comme il Faut: A System for Simulating Social Games Between Autonomous Characters." (2009)

[13] McCoy, Josh, Mike Treanor, Ben Samuel, Brandon Tearse, Michael Mateas, and Noah Wardrip-Fruin. "Comme il Faut 2: a fully realized model for socially-oriented gameplay." In Proceedings of the Intelligent Narrative Technologies III Workshop, p. 10. ACM, 2010.

[14] Loyall, A. Bryan. "Believable agents: building interactive personalities." PhD diss., Stanford University, 1997.

[15] Sally, David. "Social maneuvers and theory of mind." Marq. L. Rev. 87 (2003): 893.

[16] Paiva, Ana. "Empathy in Social Agents." International Journal of Virtual Reality10, no. 1 (2011): 1

[17] Dias, Joao, Samuel Mascarenhas, and Ana Paiva. "Fatima modular: Towards an agent architecture with a generic appraisal framework." In Proceedings of the International Workshop on Standards for Emotion Modeling. Leiden, Netherlands 2011.

[18] Aylett, Ruth, and Sandy Louchart. "If I were you: double appraisal in affective agents." In Proceedings of the 7th international joint conference on Autonomous agents

and multiagent systems-Volume 3, pp. 1233-1236, International Foundation for Autonomous Agents and Multiagent Systems, 2008.

[19] Aylett, Ruth, Natalie Vannini, Elisabeth Andre, Ana Paiva, Sibylle Enz, and Lynne Hall. "But that was in another country: agents and intercultural empathy." In Proceedings of The 8th International Conference on Autonomous Agents and Multiagent Systems-Volume 1, pp. 329-336. International Foundation for Autonomous Agents and Multiagent Systems, 2009

[20] Aylett, Ruth, Marco Vala, Pedro Sequeira, and Ana Paiva. "FearNot!–an emergent narrative approach to virtual dramas for anti-bullying education." In Virtual Storytelling. Using Virtual Reality Technologies for Storytelling, pp. 202-205. Springer Berlin Heidelberg, 2007.

[21] Weallans, Allan, Sandy Louchart, and Ruth Aylett. "Beyond Double Appraisal in Emergent Drama.", tecfalabs.unige.ch, School of Mathematical and Computer Sciences Heriot-Watt University, Riccarton, Edinburgh, EH14 4AS, UK

[22] Marsella, Stacy C., and Jonathan Gratch. "EMA: A process model of appraisal dynamics." Cognitive Systems Research 10, no. 1 (2009): 70-90.

[23] Pynadath, David V., and Stacy C. Marsella. "Minimal mental models" In Proceedings Of The National Conference On Artificial Intelligence, vol. 22, no. 2, p. 1038. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2007.

[24] Pynadath, David V., Mei Si, and Stacy C. Marsella. “Modeling Theory of Mind and Cognitive Appraisal with Decision-Theoretic Agents”. University of Southern California Los Angeles Inst. For Creative Technologies, 2011.

[25] Pynadath, David V., and Stacy C. Marsella. "PsychSim: modeling theory of mind with decision-theoretic agents." In International Joint Conference on Artificial Intelligence, vol. 19, p. 1181. Lawrence Erlbaum Associates LTD, 2005.

[26] Marsella, Stacy C., David V. Pynadath, and Stephen J. Read. "PsychSim: Agent-based modeling of social interactions and influence." In ICCM, pp. 243–248, 2004.

[27] Klatt, Jennifer, Stacy Marsella, and Nicole C. Krämer. "Negotiations in the context of AIDS prevention: an agent-based model using theory of mind." In Intelligent Virtual Agents, pp. 209-215. Springer Berlin Heidelberg, 2011.

[28] Mac Namee, Brian, and Pádraig Cunningham. "A Proposal for an Agent Architecture for Proactive Persistent Non Player Characters.", tara.tcd.ie (2001).

[29] Prada, Rui, Samuel Ma, and Maria Augusta Nunes. "Personality in Social Group Dynamics." In Computational Science and Engineering, 2009. CSE'09. International Conference on, vol. 4, pp. 607-612. IEEE, 2009.

[30] Durupinar, Funda, Nuria Pelechano, Jan M. Allbeck, Ugur Gudukbay, and Norman I. Badler. "How the ocean personality model affects the perception of crowds." Computer Graphics and Applications, IEEE 31, no. 3 (2011): 22-31.

[31] Johns, Michael, and Barry G. Silverman. "How emotions and personality effect the utility of alternative decisions: a terrorist target selection case study." Center for Human Modeling and Simulation (2001): 10.

[32] Talman, Shavit. "The adaptive multi-personality agent." PhD diss., Department of Computer Science, Bar Ilan University, 2004.

[33] Ghasem-Aghaee, Nasser, Bardia Khalesi, Mohammad Kazemifard, and Tuncer I. Ören. "Anger and aggressive behavior in agent simulation." In Proceedings of the Summer Computer Simulation Conference, pp. 267-274. Society for Modeling & Simulation International, 2009.

[34] Santos, Ricardo, Goreti Marreiros, Carlos Ramos, José Neves, and José Bulas-Cruz. "Personality, emotion and mood simulation in decision making." LS Lopes, N. Lau, P. Mariano & LMR (Eds.), eds., New trends in Artificial Intelligence 8 (2009): 215-216.

[35] Reiss, Steven. "Multifaceted nature of intrinsic motivation: The theory of 16 basic desires." Review of General Psychology 8, no. 3 (2004): 179-193.

[36] Becker-Asano, Christian, and Ipke Wachsmuth. "Affect simulation with primary and secondary emotions." In Intelligent Virtual Agents, pp. 15-28. Springer Berlin Heidelberg, 2008.

[37] Ghasem-Aghaee, Nasser, and T. I. Oren. "Effects of cognitive complexity in agent simulation: Basics." Simulation Series 36, no. 4 (2004): 15

[38] de Melo, Celso M., Peter Carnevale, Stephen Read, Dimitrios Antos, and Jonathan Gratch. "Bayesian model of the social effects of emotion in decision-making in multiagent systems." In Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1, pp. 55-62. International Foundation for Autonomous Agents and Multiagent Systems, 2012.

[39] Becker-Asano, Christian, and Ipke Wachsmuth. "Affect simulation with primary and secondary emotions." In Intelligent Virtual Agents, pp. 15-28. Springer Berlin Heidelberg, 2008.

[40] de Melo, Celso M., Peter Carnevale, Stephen Read, Dimitrios Antos, and Jonathan Gratch. "Bayesian model of the social effects of emotion in decision-making in multiagent systems." In Proceedings of the 11th International Conference on Autonomous Agents and Multiagent Systems-Volume 1, pp. 55-62. International Foundation for Autonomous Agents and Multiagent Systems, 2012.

[41] Saunier, Julien, Hazaël Jones, and Domitile Lourdeaux. "I feel what you feel: Empathy and placebo mechanisms for autonomous virtual humans." In Intelligent Virtual Agents, pp. 323-329. Springer Berlin Heidelberg, 2009.

[42] Marsella, Stacy, Jonathan Gratch, and Paolo Petta. "Computational models of emotion." A Blueprint for Affective Computing-A Sourcebook and Manual (2010): 21-46.

[43] Dufwenberg, Martin, and Uri Gneezy. "Measuring beliefs in an experimental lost wallet game." Games and Economic Behavior 30, no. 2 (2000): 163-182.

[44] Rizzo, Paola. "Goal-based personalities and social behaviors in believable agents." Applied Artificial Intelligence 13, no. 3 (1999): 239-271.

[45] Rizzo, Paola. "Goal-based personalities and social behaviors in believable agents." Applied Artificial Intelligence 13, no. 3 (1999): 239-271.

[46] Mac Namee, Brian, Simon Dobbyn, Pádraig Cunningham, and Carol O'Sullivan. "Simulating virtual humans across diverse situations." In Intelligent Virtual Agents, pp. 159-163. Springer Berlin Heidelberg, 2003.

[47] Steunebrink, Bas R., Mehdi Dastani, and John-Jules Ch Meyer. "The OCC model revisited." In D. Reichardt, editor, Proceedings of the 4th Workshop on Emotion and Computing, 2009.

[48] Gratch, Jonathan, and Stacy Marsella. "Evaluating a computational model of emotion." Autonomous Agents and Multi-Agent Systems 11, no. 1 (2005): 23-43.

[49] Wehrle, Thomas. "Motivations behind modeling emotional agents: Whose emotion does your robot have." In Grounding emotions in adaptive systems. Zurich: 5th International conference of the society for adaptive behavior workshop notes (SAB'98), vol. 60. 1998.

[50] Gebhard, Patrick. "ALMA: a layered model of affect." In Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems, pp. 29-36. ACM, 2005.

[51] Bo, Xianyu. "Other-regarding preference and the evolutionary prisoner's dilemma on complex networks." Physica A: Statistical Mechanics and its Applications 389, no. 5 (2010): 1105-1114.

[52] Hagen, Edward H., and Peter Hammerstein. "Game theory and human evolution: A critique of some recent interpretations of experimental games." Theoretical population biology 69, no. 3 (2006): 339-348.

[53] Mark, Dave. Behavioral mathematics for game AI. Course Technology Cengage Learning, 2009.

[54] Russell, Stuart Jonathan, Peter Norvig, John F. Canny, Jitendra M. Malik, and Douglas D. Edwards. Artificial intelligence: a modern approach. Vol. 74. Englewood Cliffs: Prentice hall, 1995.

[55] Loyall, Joseph Bates A. Bryan, and Scott Reilly. "An Architecture for Action, Emotion, and Social Behavior." (1992).

[56] G. Acton. Playing the Role: An Action Selection Architecture For Believable Behaviour in Non Player Characters and Interactive Agents. Master's thesis, University of Western Ontario, 2009.

[57] J. YOU. Comprehensive Believable Non Player Characters Creation and Management Tools for Emergent Gameplay. Master's thesis, University of Western Ontario,2009 .

[58] Lisetti, Christine L. "Believable Agents, Engagement, and Health Interventions." In HCI International 2011–Posters' Extended Abstracts, pp. 425-432. Springer Berlin Heidelberg, 2011.

[59] Lisetti, Christine L., and Eric Wagner. "Mental health promotion with animated characters: Exploring issues and potential." In Proc. of the Stanford AAAI Spring Symposium on Emotion, Behavior and Personality, pp. 33-44. 2008.

[60] Ekman, Paul. "Basic emotions." Handbook of cognition and emotion 98 (1999): 45-60.

Curriculum Vitae

Name: Arvand Dorgoly

EDUCATION

University of Western Ontario Fall 2011- Present

London, On

- Masters in science in Computer Science

Shahid Beheshty University 2007-2011

Tehran, Iran

- Bachelor of science in Software engineering

SUMMARY OF QUALIFICATIONS

- Able to learn any new programming language quickly
- Good team member with leadership skills
- Knowledge of game theory, artificial intelligence, and behavioural economics
- Excellent problem solving and software debugging skills
- Experienced with database design

Technical Skill

- Have experience programming with C++, C#, Java, Prolog, Python, Matlab, Pascal.
- Familiar with hardware description languages: VHDL,Verilog
- Web/DB technologies: HTML, java scripts, MYSQL
- Familiar with different SDK's including Microsoft Visual Studio, Xcode, Eclipse.
- Familiar with Open CV and OpenGL

PROFESSIONAL EXPERIENCE

- Teaching Assistant, Computer Science, University of western Ontario
- Research Assistant, Computer Science, University of Western Ontario