2009

Playing The Role: Towards An Action Selection Architecture For Believable Behaviour In Non Player Characters and Interactive Agents

Gavan Acton
Western University

Playing The Role: Towards An Action Selection Architecture For Believable Behaviour In Non Player Characters and Interactive Agents

(Spine Title: Action Selection Architecture For Non Player Characters)
(Thesis Format: Monograph)

by

Gavan Acton

Graduate Program

in

Computer Science

A thesis in partial fullfilment
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

Non Player Characters(NPC) in many types of video games must make believable choices in order to create player immersion and enjoyment. This thesis proposes, implements and tests a novel NPC architecture making use of Role Theory, Appraisal Theory and Utility-Based decision making.

Keywords: Believable Decision Making, NPC, Agents, Role Theory, Appraisal Theory, Utility-Based Decision Making, Data-Driven, Emotions, Personality, Role-Based Architecture, Artificial Intelligence, Multi-Agent System, Emergence, AI Drama Manager, Video Games, Action Selection.

Contents

List of Figures

Chapter 1

Introduction

1.1 Overview

Perhaps the most interesting aspect of character-based video games is not the fidelity of their graphics, soundscapes or physics but rather how these games may have difficulty creating meaningful emotional experiences. Traditional mediums such as film, music, and literature's greatest and most powerful works engage us at the most human and basic level: the shared human experience. Whether reducing us to tears of sadness in Spielberg's Schindler's List [72] or making us laugh in Pixar's WALL-E [20], traditional mediums have and continue to tell important and powerful stories. Video games on the other hand have had trouble engaging to its audience at a deeper level of meaning [1] [12, 80]. Players rarely feel remorse as they kill wave upon wave of enemy soldiers. This exemplifies the fact that players often have difficulty feeling any sense of attachment to in-game characters [22]. Video games' inability to connect with players is two fold but based on one fundamental concept of humanity: choice. As video games are an interactive medium, players make choices each instant they play. However, games often rob players of any meaningful choice in order to ensure the coherence of a linear story. The effect is that players do not choose

[1] While several games have been known to evoke emotions (such as Final Fantasy VII [12, 80]), these crucial moments have been delivered by cut scenes, not as a result of player interaction with NPCs.

or shape the world they play in; rather their role is akin to train conductor moving forward to a predefined final destination. Secondly, and more importantly for this thesis, choice is also restricted in that the computer controlled Non Player Characters(NPCs) are not able to make meaningful choices. They often cannot get angry if struck, do not remember past interactions, and they are often unable to react in believable ways which have not been predefined by the designers. As a result, in many games, the interactions that a player can have with NPCs are extremely limited. Video games fail to provide meaningful emotional experiences as NPCs continue to remind the player that what they are experiencing is not real.

With increasing graphical fidelity [15] and powerful platforms available to game designers, Artificial Intelligence(AI) is increasingly becoming an important part of video games [15, 54, 69, 75–77]. Players expect AI that is dynamic, able to react to unexpected events and behave believably [19, 75–77]. As somewhat of a consequence to this, NPC AI and the creation of interactive agents, is an important and active area of research [3, 26–29, 34, 54, 82].

The state-of-the-art in the creation of believable decision making has reached out beyond computer science using models from psychology(personality, emotions, appraisal theory) and sociology(social appraisal variables, relationships and role theory). However, there currently exist no known Non Player Character(NPC) architectures that integrate all of these important models together to produce decisions that are believable and whose architecture is dynamic, flexible, reusable and efficient enough for the use in real-time video games. Therefore, the focus of this research is to design an architecture based on role theory that incorporates emotional models, personality models and reasoning about consequence into utility-based decision making processes.

To this end, this research makes specific contributions in creating a foundation to which believable descision making meeting the requirements of Loyall can be achieved

[41]. This work extends the state-of-the-art in NPC architecture and their decisions by: integrating Role Theory, Appraisal Theory and Utility-Based decision making into a single Data-Driven Agent; The personalization of Roles through their integration with Emotions and Personality Traits and the Active Emotional Memory; The personalization of Utility-Based planning and action selection taking into consideration the agent's relationship to its environment via it's Roles, Emotions, Personality traits and Consequences to actions; Allowing for the plug and play of Roles used by a game designers, AI drama manager or Agents to dynamically create interesting and believable behaviour. These contributions will consequently advance the state-of-the-art in NPC architectures and believable decision making.

1.2 Motivation

Believable decision making of NPCs has been noted as an important quality of any video game. Some of the reasons include increased player immersion [6, 69, 77], increased likelihood for interesting game situations [3, 60], creating a realistic environment where players feel less humiliation when losing [77] and an increased sense of empathy towards NPCs [3,19] as well as an increase in the ways to interact with NPCs. As a result, creating better games is dependent on having NPCs that make believable decisions.

1.3 Background

Believability is a central requirement of today's NPCs and as Livinstone writes "the requirement for modern computer games is not unbeatable AI, but believable AI" [40]. Interestingly, believability is not realism, but is rather the façade, an illusion that allows us to suspend our disbelief [40,45]. Believability is also largely subjective as one's culture and

perspective define it [45]. As a result, designers need to be cognizant that they must not emulate human processes, rather simulate them such that they appear believable.

Loyall defines the believability of an NPC to be based on two groups of requirements. The first group of requirements takes the works of the Arts that define that believable characters' need to express personality and emotion in their every action [41]. They should also be self-motivated, able to undergo change and finally, have social relationships [41]. The second group of requirements is based on the fact that unlike the Arts, interactive characters do not have humans controlling their every action. As a result, processing based requirements are what Loyall defines as the Illusion of Life [41]. These requirements include the Appearance of Goals, Concurrent Pursuit of Goals and Parallel Action, Reactive and Responsive, Situated, Resource Bounded, Exist in a Social Context, Broadly Capable and finally Well Integrated [41]. These requirements reflect a high requirement that humans have when judging NPCs. Consequently, any architecture for believable decisions must take all of these factors into consideration.

In an effort to meet the requirements of believable agents, many architectures include personality models into their decision making processes. A popular personality model is the Five Factor Model(FFM) also known as OCEAN [36]. The FFM is a set of distinct traits based on factor analysis [36]. Generally, an architecture will represent each trait as a single scalar value that can affect a number of agent processes including Goal Selection, Action Selection, Planning, Event Appraisal, and Emotional Responses [1,19,34,36,70]. Personality has been implemented using rule-based processes [5] and Utility-Based processes [19, 34]. A recent personality model provides an explanation of underlying motivations that are biological in nature [66]. Riess proposes 16 basic desires that motivate action as well as suggest the method in which they are modulated through action [66]. This perspective appears to be well suited for Utility-Based implementations; however none are currently known to exist.

Emotions have also been synthesized in believable agents to meet the requirements of believability by affecting a number of processes including Goal Selection, Action Selection, and Attention. Many models of emotion have been synthesized including Eckman's Six Universal Expression [8,49,70,85], the 22 Emotions in the OCC Model [3,26,27,34,56] and Sloman's [71] tertiary emotions. Much like personality, these are varied in their definition and generally represented by a single scalar value. Many features of emotion have been modeled including their elicitation due to situational meaning, concern and change [3, 26, 27, 32, 79]. Concern for example has been modeled by the elicitation of emotion due to inter-goal threats in an agent's active plans [3, 26, 27]. As a result, events that force the agent to re-plan will generate emotion based on how the agent was able to cope with the change. Habituation has also been partially modeled where emotions decay over time [3, 9, 70]. More complicated systems performed calculations for emotions in response to consequences. For example, Marsella and Gratch and Aylett [3, 26, 27] performed utility calculations that considered the agents actions against how others would react. While extremely computationally expensive, this technique could be modified for application to video games.

As humans are social beings, NPCs that wish to be believable must be able to reason in regards to social relationships, culture, value and norms [18, 27, 29, 45, 70]. Some relevant research in this area is the use of social cognitive variables of Dominance and Agreeableness [33]. Bailey and Katchabaw implemented Isbister's social variables through a rule-based emergent social system that allowed for the varied interaction among its agents based on their social variables, emotional state and personality factors. Another method used to describe social relationships and social structures is Role Theory [29, 57]. Here a role describes one's relationship to another person(Mother, Boss, Friend) or group(position) [57] and any person may have multiple roles. Roles further describe a relationship in terms of associated Desires(goals), Beliefs(Values, Norms and Worldview) and Intentions (BDI)

[29,57]. This provides significant advantages for designers as roles can be reused, and they are intuitively understood by non experts [29, 57]. Guyevuilleme implemented roles as a rule-based system but also extended it with a few utility calculations. Most importantly, he tied a numerical value to each role asserting its importance to the characters. This provides a simplistic but very effective method of defining the importance the goals held within the agent. Agents' values are represented as rules. An example value could be "wearing a tie at dinner". Interestingly, he suggests that these cannot be represented as utilities. However, Panzarasa suggests that values should be optional [57] which in-turn suggests that their adherence could be turned into a utility calculation. For example, all monks "shall not kill". However, given certain circumstances, personality traits or emotions, this may not always be the case. Consequently, social information plays an important role in the creation of believable characters.

Looking at complete architectures, decisions need to reflect the NPC's emotional state, personality traits as well as a precise knowledge and understanding of social relations as well as a care to the consequences of actions. Unfortunately, no known architecture integrates all of the components to create believable decisions for video games. While Marsella and Gratch, and Aylett [3, 26, 27] do excellent work in modeling emotions using appraisal theory, their proposed frameworks fail to consider a basic personality model in their decision making. Consequently, their planning fails to convey personality as well as suffers from low re-usability. Their approach to social knowledge also falls short in clearly defining the relationships. Also, Marsella and Gratch, and Aylett's implementation of action consequences is useful, but their approach appears to be too computationally expensive for the application in real-time video games. Johns integrated a utility function that combines both a personality model and emotions is a step in the right direction as it is efficient and allows for the characters' actions to reflect their personality and emotions. However, his approach does not take into consideration the consequences of

actions and ignores critical social knowledge. And so while Johns' framework is highly reusable and flexible, it would fail many tests for believability. Finally, Guyevuilleme's role-based architecture provides an intuitive and highly reusable structure for social knowledge. But it does not consider personality models and does not describe how roles can be used to aid the understanding the consequence of actions. This illustrates that the state the art in architectures for believable decision making still has much room for improvement.

1.4 Problem Statement

There is a clear requirement for an NPC architecture that generates actions that meet Loyall's requirements of believability [19, 75–77]. In addition, for the use in video games, the architecture for decision making needs to be highly flexible, re-usable and have good performance.

1.5 Proposed Solution

In looking at the state-of-the-art, it appears that there is significant room to improve upon the existing architectures. Unlike other implementations, every decision in our architecture will consider the agents' emotional state, personality traits, relevant social information as well as a care of consequence. Furthermore, our architecture will maximize flexibility, reuse with performance through the use of Roles.

To accomplish this, we propose a BDI role-based architecture that uses a Utility-Based decision making system. Both Goal Selection and Planning will be Utility-Based processes that will create believable actions by making extensive use of personality and emotional models, as well as social reasoning of consequence. A sense-think-act cycle will be implemented where events are sensed and impact current emotional state of the character. The

character's internal state will define which goals are currently being pursued. Furthermore, planning and the actions selected to achieve goals will use a utility function that is a reflection of the character's emotional state, personality variables as well as consequences to the actions. This work will extend the work of [29] by creating flexible roles as well as tying them to emotional memories that affect the agent's current state when active. As a result, this research has many critical contributions to the state-of-the-art in design of believable NPCs for video games.

We propose a role-based architecture as seen in [29] that has significant advantages in both design and flexibility. As discussed roles are an intuitive concept for designers which make them well understood. Roles also provide an excellent way to group a number of concepts that describe behaviour. Essentially roles are self sufficient in their description of behaviour, and with the right architecture in place, simply adding a role to an agent should result in complex and believable behaviour. Furthermore, roles are highly reusable as once defined, can be applied to any number of agents. Another advantage of a role-based architecture is that roles can be given a numerical assignment that captures at a high level the importance the relationship to the agent. This in-turn provides a simple yet effective mechanism to aid the goal selection process. Finally, as many roles can be applied to an agent, this can create a state of *role overload* where it becomes impossible for the agent to meet the goals for each role. This translates into believable decision making, through internal conflict and believable emotional states that can be useful for dilemma-based AI Drama Manager and in creating believable behaviour. However as noted above, roles on their own do not provide enough information to adequately create believable decision making.

We propose to extend the role-based architecture by associating additional information specific to the agent. This information should include the optional adherences to goals, values and norms that allow designers to identify the goals that are important to

the individual agent. This is in contrast to Guyevuilleme's implementation that assumes mandatory adherence to all goals, values and norms. This could be efficiently accomplished by associating a single scalar with each of the goals, norms and values. Furthermore, additional information will be associated with the role to be used in consequence planning. As actions can be identified to have a negative impact on the role's goals, this provides the agent with the information required to identify the utility of an action in respect to other agents' goals. Finally, as mentioned above, calculating the utility of an action with respect to other agents can be prohibitively expensive. Roles provide a mechanism to which generalizations can be made to these calculations. For example, all enemies within an area will be harmed by actions that are in conflict with their roles goals. As they share these goals, a utility calculation for the action can be performed once for the role and then generalized to each member agent within the group. This creates a more efficient approximation of consequence than seen in [3, 26, 27]. As a result, adding and associating information to roles should result in increased believability and efficiencies.

In addition to a strengthened social reasoning system, the agent's architecture must maximize believability, flexibility and re-usability in the use of emotions and personality models. Frijda discussing emotional modeling and Ortney discussing personality, suggest the best way to achieve this is through the use of a fundamental trait models [24, 55]. A significant advantage of using these models is that they allow for efficient integration into utility functions as seen in [34]. Here actions may be associated with having a utility for the current emotional state and personality factors. Reiss' theory of 16 basic desires seems especially well suited in these regards as goals, and the consequences of actions can be associated with each desire. No systems are known to use Reiss' theory in their utility calculations and so, our implementation will be the first to synthesize his model. Furthermore, Eckman's Six Universal Emotions have been noted to be well suited for these purposes and have the advantage of being the basis to many facial animation packages

[8, 49, 70, 85]. As these models represent emotions and personality as simple scalars, it would be possible for AI Drama Manager to populate a world with unique characters simply by changing these basic parameters. However, even Eckman's Emotional Model does not fully encapsulate many of the complex emotions found in humans. As a result, we are the first to propose the use of role specific emotions. A role of Wife for example, could have role specific emotions such as jealousy or love. This could be an efficient method to allow agents to maintain a greater breath of emotions that affect their decision making processes. The integration of the personality and emotional models into roles will primarily be used to affect the decision making in Utility-Based goal selection and planning.

A final contribution is to extend the cognitive structure of agents with emotional memories. Important events that either aid or threaten a role's goals should be translated into emotional memories for the agent. These emotional memories may be associated with roles such that they may affect the interactions with other agents. While we have seen the cognitive state be affected by relationship variables in [26,27], the extension into roles provides significant advantages. This can play an important role in establishing an emotional state of mind for the character that in-turn will shape the plans created for the associated role. For example, when a goal is active for a particular role, the associated emotional memories should become active. This in-turn will affect how planning occurs. The result of emotional memories can provide an significant improvement in the believability for agents.

1.6 Outline

Chapter 2 discusses the related work within this area. Within this chapter we examine the concepts of Believability 2.1 and its requirements 2.2, Traditional AI techniques 2.3 and finally Architectures for Believable Agents 2.4. Chapter 3 discusses the design of this work is presented including the architecture 3.2 and its processes 3.3. Chapter 4 discusses the

implementation while Chapter 5 the Evaluation and a discussion of the findings. Finally Chapter 6 outlines areas of future work along with closing remarks.

Chapter 2

Related Work

2.1 Understanding Believability

Believability is a term that most computer scientists may find unfamiliar from a scientific perspective. However as a concept, believability is increasingly important to understand beyond an intuitive definition. In video games, believability provides us with the guiding principles that make it possible to outline the most fundamental concepts of Game AI. More so, the literature in the design of interactive agents, points to the importance of believability in the video games [15, 54, 69, 75–77]. Traditionally, Game AI has been responsible for providing a challenge to the player but this is no longer the case. As Livinstone writes,"the requirement for modern computer games is not unbeatable AI, but believable AI" [40]. In the following section, the important concepts of believability within video games are discussed before moving onto a set of requirements for the believability of NPCs in video games.

When we discuss believability, really we are talking about Coleridge's willing suspension of disbelief [61]. Believability is not realism. When observing a cartoon character such as Mickey Mouse, we see that his purpose is not to fool us that he is a mouse. But rather

his purpose is to strongly express personality and so that we buy into his existence [45]. For NPCs, it is this appearance, the façade that counts. As a result, believability is more important than the truth [40].

Believability is also largely subjective. Each audience member applies his own unique personal and cultural experience to scrutinize a character [45]. As a result, what is believable to one person may not be to another. MacNamme demonstrates this in a recent study where virtual agents where shown interacting in a bar to an audience with a diverse cultural background [42]. The behaviour of the agents was either non-interruptible (where an agent would complete and action before going on to another) or interruptible (where an agent could begin performing new actions before having completed a task). During his study, an interruptible character would go to the bathroom, chat with friends and finally return to his seat at the bar. In contrast, non-interruptible characters would go to the bathroom and immediately return to their drinking. Interestingly, Italians found the interruptible and more gregarious characters to be the most believable, while an Irish subject found the determined drinking behaviour being most believable [40]. And so, believability appears to be largely subjective to the audience's personal and cultural experience.

Having demonstrated the key concepts underlying believability, a formalized set of requirements is now examined.

2.2 Requirements for Believability

Many authors have attempted to define requirements for believability in a wide variety of areas including narrative, agents, robots [23, 41, 45]. This section describes a set of requirements for believable agents(NPCs) based on the work of Loyall [41]. The first of these describe requirements that have been taken from the Arts.

- *Personality*: The first and possibly the most important requirement of NPCs is

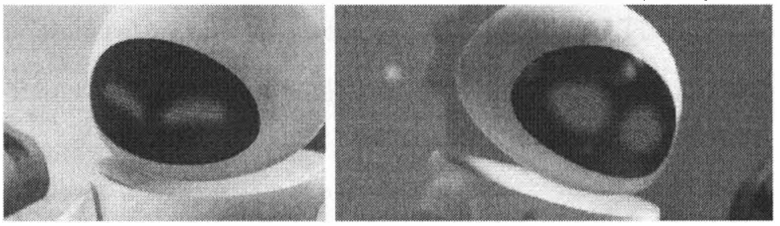
personality [9,23,39,41,45]. Personality allows us to define the specifics of a character, not the general. It is these specifics that we can use to characterize an agent. For example, in the memoirs of Disney artists Thomas and Johnston, they articulate the importance of personality in believability. "Any approach that attempts to create believable agents must allow (even require) personality based variation to this depth, all across the wide-ranging areas of the agent."(through Loyall) [41]. And so, if designers wish their NPCs to be believable, they must express their personality such that they are truly identifiable.

Personality should be imbued into a character such that it influences their every action, emotion and thought [45]. More so, personality should bring together all the character's powers into a single form of expression [45]. This allows for both the verbal and non verbal cues to communicate to the audience both the thoughts and intentions of the character.

It has also been recommended that characters have exaggerated personalities such that their behaviour clearly communicates to the audience the characters' personalities [33].

- *Emotion*: The second requirement for believability of NPCs is emotion [9,23,39,41, 45]. Believable characters must have emotional reactions to the internal events such a realization of guilt or to external events such a surprising action from another [41]. Also, emotions must be expressed in personality specific ways [45]. For example, if a character is surprised, the character should display this surprise in a way that is coherent with their personality. If a character is angry, they should be able to communicate this. In Figure 2.1 , we see Eve, a robot from Pixar's 2008 WALL-E film. Here her emotions are clearly expressed through her eyes and body language that helps to create her believability. More so, it is not enough to simply express

Figure 2.1: WALL-E's Eva in various emotional states (Happy and Sad)



emotion, believable characters must understand and react to emotions of others [41]. It is expected that when another is crying or angry, our actions may differ when in their presence. As a result, emotions play an important role in the believability of an agent.

- *Self Motivated*: Loyall defines this self motivation from the works of Disney's Thomas and Johnston who write "Prior to 1930, none of the characters showed any real thought process . . . the only thinking done was in reaction to something that had happened. Mickey would see [something], react, realize that he had to get a counter idea in a hurry, look around and see his answer, quickly convert it into something that fit his predicament, then pull the gag by using it successfully." [41]. It is important that characters do not simply react to the outside world, but also perform actions based on internal motivation [45]. Other authors have described a similar requirement as "self impelled actions" [39]. Self motivation appears to be an important requirement of believability.

- *Change*: An important characteristic for the believability of NPCs is change [41,45]. Change may come because of an internal event such a realization or external one such as an attack from an enemy. A change in a character may be demonstrated by a change in their personality, their relationships with others or the actions they choose. However, change cannot be arbitrary. It must be reflective of the character's

personality [45]. Change is also an important requirement to video game narrative, as it is fundamental to any narrative experience [3, 45]. Themes of personal growth dominate many of the stories told, and NPCs in games must demonstrate change based on events they experience.

- *Social Relationships*: Another important for an agent's believability appears to be the ability to engage in social relationships [23, 29, 41, 45]. This has been true for character based arts where interaction between characters affects the relationship between them. Thomas and Johnston(through [41]) remark that the development of relationships among characters was the one of their arts greatest advancement. Indeed the power of relationships can be viewed as a central pillar of the human existence as well as a requirement for believability.

- *Consistency of Expression*: When an agent needs to express himself, he needs to apply a consistency to his expressions [23, 41, 45]. His voice, gaze and movement should all reflect his current state of emotions, personality as well as the context. For example, an angry person should be agitated, use a louder voice and wilder body movements. If the combination of these actions are unrelated or unfocused(happy face with angry voice), the audience becomes confused and believability is lost.

- *The Illusion of Life*: Interestingly, as traditional mediums rely on humans(actors, writers, animators) to make decisions for their characters, many requirements were overlooked by early studies of believability in the arts [41]. And so, as agents must rely on a logic system to ensure believability, the following requirements outline important characteristics that ensure an illusion of life.

  - *Appearance of Goals*: The first component of the illusion of life is an appearance of goals [9,39,41,45]: All living beings have goals of varying complexity. A simple

goal may be finding food to eat, whereas a more complex goal could be "being honest". The audience must be able to understand why the agent is acting in a certain fashion and what they are trying to achieve. And so, an appearance of goals implies self motivation that is important to the NPCs believability.

- *Concurrent Pursuit of Goals and Parallel Action*: In everyday life, we often interweave multiple actions. We are expected to walk, talk and chew gum, at the same time. Agents must also have this ability to perform multiple actions in parallel or they may come across as odd and not believable [41,45]. The agents should also have the ability to interleave their actions to achieve multiple goals. For example, a human agent who wishes to buy supper and pick up his daughter should be able to accomplish both in the same trip.
- *Reactive and Responsive*: Agents must also be able to react to their environment [41,45]. If confronted with a life threatening situation, an agent should be able to react in a believable fashion by trying to remove himself. Not only must a reaction occur, but also the response must happen at a reasonable speed [41,45]. Typical human reaction depends on the stimuli and the task [81] . Simple reaction time tests like catching a falling ruler have response after 150-200 Milliseconds [81]. More complex recognition of random objects in space has media reaction time of 445 Millisecond [81]. Complex reasoning may take substantially longer, and it is important that these reactions be properly simulated. Believable reaction times should be respected in the action/reaction process of agents.
- *Situated*: An agent is situated if he is able to constantly evaluate his current situation and to change plans if required [41, 45]. For example, if an agent on the battlefield decides to help another, he should constantly re-evaluate the current situation ensuring his actions are realistic and achievable. As games

often deliver a story, it is important that agent's playing a role within a story should be situated in the narrative experience [3]. This should help in the believability of the narrative as agents make decisions that make sense in the context.

- *Resource Bounded*: Another important requirement for the illusion of life is that agents must have a limit to their abilities [41, 45]. For example, they must have capabilities in line with their powers. A human agent should have the ability to jump but it should not be so high that it breaks believability whereas a super human mutant would not have the same restriction. As a result, agents should be resource-bound.
- *Exist in a Social Context*: Among human capabilities is the adeptness to understand the cultural and social conventions of the world we live in [41, 45]. We understand that standing a certain distance away from another is considered proper and that certain physical actions have meanings. Agents should be aware of these conventions and able to act and react to them.
- *Broadly Capable*: Agents should seem to have abilities akin to the intended representation [41, 45]. If an agent portrays a human, they must appear to think, act, sense, display emotion, talk and exists in a dynamic world. If any of these capabilities are not present or are not believable, the agent will fail to be believable.
- *Well Integrated(Capabilities and Behaviours)*: The final requirement for an agent's believability is that their capabilities should be seamlessly integrated [41, 45]. There should be no observable boundaries in its behaviour. This seamless behaviour is seen to be believable.

In summary, the believability of NPCs in video games is dependent on many requirements which have been derived from the Arts(Personality, emotion, self motivation, change and social relationships) and Computer Science(Illusion of Life) [41,45]. In order to achieve believability, many of the requirement must be successfully executed in parallel. Failure to do so may result in a break in the audience's suspension of disbelief and the agent's believability.

Having listed the requirements for believable agents, we will now examine the history of AI in games focusing on how NPCs make decisions.

2.3 Artificial Intelligence In Games

Game Artificial Intelligence(AI) is a misnomer in the world academe as it shares little in common with Classical AI. At the core, video game AI is the method to decide the behaviour of any aspect of a video game. Generally, game AI has been responsible for, but not confined to, creating NPCs that provide a challenge and creating the structure for narrative [63, 82] while Classic AI is focused on generalized problem solving. In the next section, the goals, challenges of Game AI, and traditional techniques for decision making in games are discussed.

2.3.1 What is Game AI?

First and foremost, Game AI must operate in real time, simulating movement on the screen by quickly replacing one image with another. For smooth and believable movement, higher rates providing smoother action is desirable. As a result, the processing, decision making, and rendering of video games is highly time-sensitive [63, 82]. This has typically resulted in AI that is computationally inexpensive [15].

Game AI is also concerned with creating embodied characters that operate in a complex game world [1] [45]. AI must be self aware of its body and the complex environment it operates in. This is a stark contrast to traditional AI that is motivated in creating solutions to a specific problems of the mind, excluding the inclusion of the body. Game AI should have "a broad range of shallow sensory, decision and action capabilities rather then a single, narrow, deeply modeled capability" [45]. This is reflective of the broad set of requirements outlined for believability.

Another difference is the type of solution found in Classical AI and Game AI. While Classical AI tries to create general solutions, Game AI focuses on creating believable be-

[1] In [45], Mateus discusses what he calls behaviour AI. This is essentially Game AI for all intents and purposes

haviour for the agent given a specific environment [45]. Here the Game AI is only appropriate given the environment the same way a fish is only effective while in the water. Removed from its environment, it will not succeed. As a result, game AI has generally focused on bundling behaviours that are then used in contextually appropriate times. Classical AI tends to use a more generalized problem solving whereas game AI is the specific.

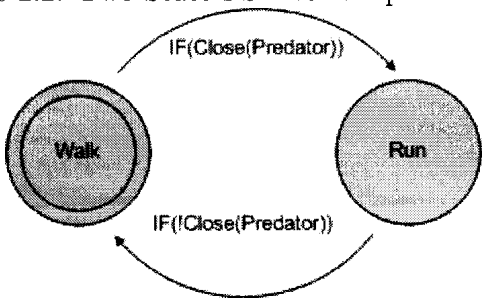
Gold's comment is very appropriate when he wrote that "AI is the big lie of the games business" [25]. Game AI does not try to simulate the real world but is rather the smoke and mirrors that create an illusion of intelligence. Game AI is considered to be successful when it is believable and appears alive through its actions. Classical AI in contrast is successful when it accurately solves a particular problem.

Game AI is found everywhere in games and describes many of the behaviours of characters on and off the screen [63]. On screen, AI has been used for the decision making of AI characters, their movement, and reaction to world events as well as executing strategies in order to provide a challenge to the player. Off-screen, AI has been used to create dynamic music and adjust the pace and difficulty of the game [4, 31]. And so, fundamentally, AI is involved in every decision about the behaviour in-game. Having outlined what Game AI is, we will now look at how Game AI has traditionally made decisions.

2.3.2 Traditional Techniques

As discussed earlier, game AI is concerned with making decisions that affect behaviour. Through the evolution of video games, many techniques have been implemented with increasing complexity in design and computational requirements. Traditionally, game AI have used many techniques including Finite State Machines, Rule based Systems, Scripting, State Space Searching, Expert Systems, Fuzzy Logic, Emergent Behaviour, Genetic Algorithms, Artificial Neural Networks and Flocking [63]. In this section we will discuss many of these traditional techniques to provide the reader with a basic understanding of

Figure 2.2: Two State FSM for simple behaviour



how believability and decisions have been created in video games.

2.3.2.1 Finite State Machines

Perhaps the most influential and ubiquitous technique for decision making in video game AI is the use of the Finite State Machine(FSM) or Finite State automaton [63,82]. A FSM is an abstract model for computation which is composed of:

- A finite set of states
- A start state
- A current state which is initially the start
- A set of one or more accepting states
- A set of transition functions that map a given state and input condition to another state

Finite state machines allow a designer to easily modify behaviour by associating each state with some explicit processing. For example, Figure 2.2 demonstrates a simple two state FSM that creates a running or grazing behaviour based on the proximity of a predator.

Some of the advantages of FSM include understandability, low computational cost and predictability [63].

Some of the disadvantages of FSM include the following: They become difficult for designers to understand and maintain as the number of states increases [82]; and all states must be known ahead of time. This restricts the number of behaviours and possible interactions with the AI.

In the end, FSMs are extremely useful for bundling behaviours into logical functioning units with little computational cost [63,82]. As a result, many video games use some form of FSM to create NPC behaviour. However, given that FSMs must have a logical transition for every situation they encounter, creating truly believable characters will be impossible as the designer would have to know ahead of time every possible interaction and produce a state with an associated action. At least for simple behaviour, FSM provide a basic level of believability.

For a more in depth look at FSM please consult and their use in video game AI one should consult [63].

2.3.2.2 Scripting

A prevalent type of AI in video games is the use of predetermined scripts that are precompiled in a non-native language such as lua or perl and interpreted into engine native commands [15, 63]. Scripting is a special case of a rule-based system that checks every frame if a given script should be executed. Processes that use scripting may be called rule-based or reactive. If conditions in the world are true, the commands defined by the script will be executed in the game world. An example script would be held in a text file that is precompiled. For example, the script below executes in the condition that the predator is close.

```
script Run (NPC npc, Condition PredatorClose )
{
    // handle transition from walk to run
```

```
    IF(npc.action == WALK)
    {
        npc.action = RUN;
    }
}
```

The result, much like a finite state machine is that when rules fire when certain conditions are true, resulting in a change of behaviour. The difference from FSM is that scripting allows game designers to work on simple scripts that do not require the expertise of a programmer to create behaviours [11, 15] . Another advantage of scripts is that they allow for somewhat reduced complexity as scripts are compiled into machine instructions during pre-compilation [47].

As a script bundles behaviours that are predefined, scripting has the same issues has FSM in regards to creating believable decisions. That is, for each possible interaction and event, a unique script must be created to ensure that the character behaves properly. As discussed with FSM, this is unfeasible for a human designer to think of every possibility much less create a script for each event. While scripting is an effective way to deal with static and simple NPCs, it will not be the way truly believable characters are created in the future.

2.3.3 State Space Searches

Another important method for decision making in agents has been through mapping the problem space into a graph problem and solving it at run time. Under the heading State Space Search(SSS), many systems have successfully applied graph searching to a variety of video games [63]. Most prominently SSS's have been widely used in the directed search for NPC movement within a world using A* algorithm [30, 62] and more recently intelligent

action selection through Goal Action Oriented Planning [54] and Utility-Based Planning and Decision Making [63]. Even rule-based Systems can be thought of a having a SSS component in the inference engine and modification of working memory.

In state space searches, an abstract graph is created where there is [63]:

- A set of nodes that represent in-game states.
- A set of labeled edges which represent the transition from one state to another

As a result, the graph is represented as a network of nodes that are tied together having a representation of the search space. Here the graph represents world locations where each labeled edge has a distance.

A search through the graph then occurs by selecting a node as a start state and then creating a list of nodes that ends at the goal node. With this in mind, we will now describe three types of systems that use SSS to create believable behaviour.

2.3.3.1 A*

Traditionally, a graph can be searched through a brute force method or the search may be heuristically guided by a utility function [30, 62]. A* is an example of such a depth first heuristic search that has been particularly useful in planning the movement of NPCs in the world [30, 62]. A* operates by a selecting the next node in its search by applying a utility function as seen below to the labeled edge whose fitness is maximized. Here, a utility function gives us an idea of how well one edge compares to another; for example, assuming an NPC is in a graph at a start node and wants to arrive at the goal node in the minimal time. Our utility function would be looking for not only looking for the least cost edge but also the one that brings us closer to the goal. This is defined in the equation below.

$$\text{utility} = \text{distance(node, goalNode);}$$

At each step, the A* algorithm is guided by the utility of a particular node until the goal node is found. The beauty of this algorithm is that its use of a utility function allows it to take an approximation of how useful a particular edge in the graph is. However, this is entirely based on the quality of the utility function. If the utility incorrectly models the graph, then it will expend a large of computational effort searching through the graph needlessly.

The advantages of A* are its ability to efficiently find solutions to graph problems which can then be translated in to the game world [30,62]. Most movement by NPC in games is decided by an A*. More than that, it is possible to create a dynamic utility function that can allow a NPC to consider many possibilities in while making a decision. For example, the utility function at one node could consider important characteristics of the terrain based on the NPCs equipment while the information at another node could be ignored. This in-turn could create believability as NPCs appear to be much more situated as their actions seem more intelligent and thoughtful.

Of course the disadvantage of A* is that it must be performed at run time hence being more expensive computationally. Another issue is that modeling the utility function can be non-trivial as an incorrect model will result in a reduction in performance over other simpler algorithms. This places an increased demand on programmers who need to tweak and modify the heuristic function to achieve believability.

For more information on A* and other heuristic searches be found in [30,62].

2.3.3.2 Rule-Based Systems

Rule-Based Systems(RBS) have a long tradition in classical AI through automated reasoning and theorem proving [63]. More recently they have been applied to video games and present a very interesting and powerful framework for believable AI [37,38]. RBS are composed of [63]

Figure 2.3: Rules for a RBS with behaviour for predators

```
1. IF(PredatorClose) Then IncreaseFear(0.1)
2. IF(NoPredators \&\& Fear \textgreater 0) Then DecreaseFear(0.2)
3. IF(Fear \textgreater 0.5) Then SCARED
4. If (Scared \&\& Have Gun) Then ATTACK
5. IF(Scared) Then RUN
```

- A set of facts describing the world
- A working memory where the facts about the world are represented and kept up to date.
- A set of rules composed of preconditions and post-conditions. These can be thought of as "If X then Y" statements which allow for automated reasoning. The result of a rule may be the execution of an action or a change to the working memory.
- An Inference Engine which uses the facts and rules to make inferences and the world.

A simple example of an RBS could be the following. Assume in working memory there exists two facts (Predator Close, Have Gun) and the rules in Figure 2.3.

Now the inference engine finds any rules that match the current working memory. Some RBS may execute one rule at a time while others such as SOAR may execute rules in parallel [37,38]. In the example above, if a predator is near the fear value in the agent will increase. Also, if the fear reaches a certain level then a new fact "SCARED" will added to working memory. Once this is done, during the next iteration the rules 4 and 5 will execute. As multiple rules are fired, then a conflict resolution engine will decide which rule to execute. This could be in the form of a rule that always takes the most complicated rule, or a utility function could evaluate both actions [26,27,37,38].

RBS have many strengths that have translated them into being used in video games. The first of strength is that rules can be easily understood by non experts [63]. The

simplicity of rules allows for almost anyone to create rules. Another strength is that it is easily modified as new rules are seamlessly integrated to created new behaviour [63]. Simply adding a new rule to the rule base extends the functionality of the system without having to make any other changes. The possibly greatest strength of RBS is that they can create complex behaviour through very simple rules. This allows for increasingly believable characters.

RBS are not without their disadvantages. The first of these is that the creation of rules may require an expert of the system for the proper functioning [63]. Also, as rules define which behaviours are possible, only a finite set of possible actions can be created unless combined with other AI techniques such as fuzzy logic. Another disadvantage of RBS is that the inference engine and resolution can be computationally expensive in comparison to simpler methods such as FSM. Of course, as new rules are added to the rule base, it may sometime be difficult to establish their effect in a chain of complex rules that modify working memory [63]. This higher complexity in design may impede the creation of believable agents. Traditionally, RBS systems have also been rather weak in dealing with uncertainty as designers must encoded this information by hand into the rules [26].

The complex behaviour that RBS can create through simple rules is the compelling reason that they have been used in many games. With an increased computational cost, smarter, more believable AI have been created.

2.3.3.3 Goal Action Oriented Programming

Where A* is a generalized search algorithm, Goal Action Oriented Programming(GAOP) is a programming paradigm that creates believable actions thought the searching of graph of actions to achieve a given goal. This means that in GAOP [54]:

- Each node in the graph is a state

Figure 2.4: Rules for Goal Action Oriented Planning

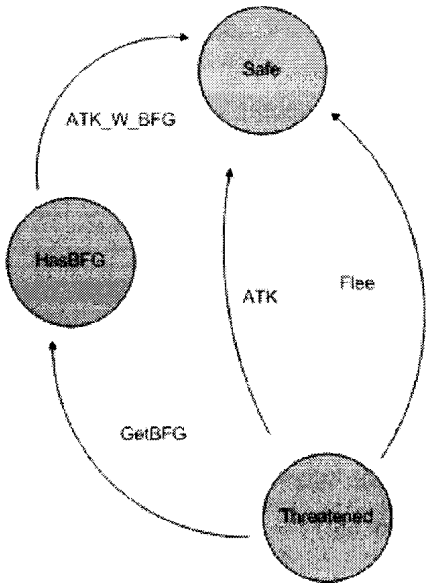
```
Action: AttackWithPipe ( AWP )
pre conditions: HasPipe
postConditions: OtherIsHurt

Action: Attack (ATK)
preconditions: none
postconditions: OtherIsHurt

Action: Flee
preconditions: CanRun, Health  0.5
postconditions: Safe

Action: Hug
preconditions: Safe, LikesOther
postconditions: OtherIsHappier
```

- Each edge defines an action whose actions transitions from node state to another.
- A goal node set which reflects the desired effect of an action. This may reflect a world state, such as having health above a numeric value.

A planning mechanism known as the planner then creates a sequence of actions at runtime to achieve a desired goal [54]. For example, Figure 2.4 lists a series of actions with pre and post conditions. When assembled as a graph as see in Figure 2.5 , one can see that by stringing a series of actions together, the shown effect can be achieved. Here, to achieve a goal of safety, the planner would create a plan of action. Such a plan could be 1. GetPipe 2. AttackWithPipe

Generally, the planner will perform a regressive search from the goal to the available actions as this is more computationally efficient [54]. Although not discussed, the planning system can be upgraded with a heuristic to guide the search when it is possible to evaluate

Figure 2.5: Visualization of GAOP Rules



the desirability of actions. This may aid in the believability of dynamic selection of actions. This will be further discussed in the following section.

Some advantages of GAOP systems are they are easily modified [54]. Simply adding new actions with pre and post conditions allows for more complex plans to be created. Also, once a basic states and actions are created in the game engine, designers can created their own pre and post conditions that allow changes to the operation and behaviour of the NPC [54]. Much like scripting, this allows designers to have greater flexibility without having to have programmers implement the complex behaviours.

One disadvantage of GAOP systems is that they must perform their plans at runtime that is computationally expensive. Another drawback of using GAOP systems is that dealing with uncertainty in plans still remains a difficult task [27]. Furthermore, GAOP on its own does not provide a system for conflict resolution and consequently has no method to differentiate which plan is the best among other competing solutions.

The GAOP system has been successfully implemented in many state-of-the-art video games such as FEAR that has been widely applauded for the believability of its AI [54].

Figure 2.6: Utility Base Planning

1. `ATK(.2)`
2. `ATK_W_PIPE(.5 * SkillWith(pipe))`
3. `GET_PIPE(.01 + EffortToGet(pipe))`
4. `Flee(.6)`

The flexibility and simplicity of the GAOP system allows for easy development of dynamic believable behaviours.

2.3.3.4 Utility-Based Planning And Decision Making

One main weakness of GAOP and other systems that generate plans is the ability to decide which plan is the best among competing solutions. To solve this, a common approach is to introduce for each action a utility function that quantifies a guess to how useful the action is [6]. The result is for each plan, a numerical assignment of 'worth' is found by adding up the expected utility of each of its actions that in-turn is used to rank plans. An example, is the setting for GAOP mentioned above. Assuming that a utility function has been described for each action and the final plan is subject to a heuristic that the highest combined utility is the best.

Looking at the above actions in Figure 2.6, the shortest plan with the highest utility would be to flee. However, if the pipe is near and the agent's skill is high with a pipe, that utility may be the highest.

Using the expected utility has many advantages as it allows for behaviour that is dynamic and situated. Also, as utility functions can be expanded to incorporate many aspects of the agent's current situation, believable decision making can be achieved. [3, 27].

Unfortunately, creating the utility function for actions may be difficult in complex settings where relevant information must either be derived using a logic system, pre-computed

and modeled by hand [3,27]. This may place significant effort on the designers to correctly model their intended behaviour.

And so, while planning with utility is advantageous in creating dynamic and informed decisions, it is computationally expensive with several caveats.

2.3.4 Summary of Techniques for Game AI

In closing this section, we have outlined a number of traditional techniques for making decisions for NPCs in video games. Finite State Machines are the most simple of methods but are mostly effective when there is limited processing and the number of states is known beforehand. As a result, FSM suffer from limited believability, as they become intractable from a design, development and maintenance perspective for complicated NPCs. Scripting with predefined scripts is slightly more computationally expensive than FSMs but they are significantly easier to understand and develop as behaviours become bundled in a logical fashion. Finally, within the State Space Searching Systems, we discussed A*, rule-based Systems, Goal Action Oriented Programming and Utility-Based planning and decision making that all have many strengths based on their ability to dynamically make decisions based on the current world state. They also have a high level of flexibility for the designers. State Space Searching methods are computationally more expensive than other hard coded methods but the payoff is in the believability of the agents who appear to be better situated and more contextually sensitive to the other offline techniques as mentioned above.

The following chapter discusses the concept of agent-based design as well as the injection of psychology, sociology to increase the believability of decision making in the state-of-the-art NPCs.

2.4 Architectures for Believable Agents

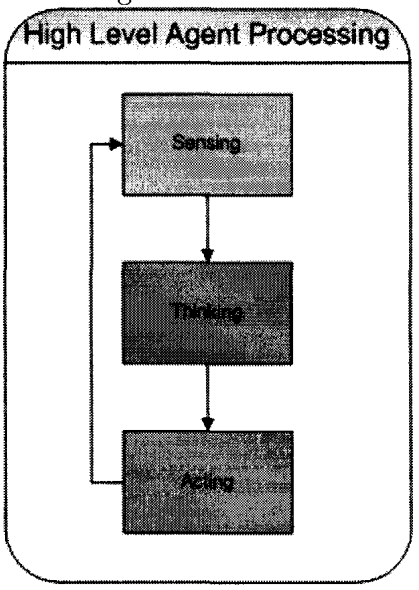
One methodology for creating believable NPCs in video games is to use an Agent-Oriented AI (AOAI) [39,63]. Traditionally, NPCs in video games are scripted and whose behaviours changes based on an observer process that modifies its states or calls for changes when it deems them appropriate. However, this method often breaks immersion as the observer has inside information about world states, giving NPCs the appearance of having cheated [6,63]. In response, many games and researchers are attempting to increase the believability by making decisions through the eyes of the NPC. In this paradigm, each agent is responsible for making decisions, based entirely on its own assessment of its local environment [63].

Agent-Oriented AI tends to process the environment in a sense-think-act cycle as can be seen in Figure 2.7 [9,32]. The sensing process involves processing of internal and external stimulus caused by events that are broadcast in the game world. Events may come in the form of actions caused by other players or the game world and are then filtered based on their relevance to the agent. Events deemed relevant are then passed onto the thinking process.

During the thinking process, processing occurs in order to formulate which action to perform next. Finally, a response action is selected from a set of possible actions. The action is passed onto the acting process that executes the action and broadcasts the event into the game world.

The most common framework among AOAI is BDI, where agents are formed of Beliefs, Desires and Intentions [65]. Here beliefs are facts about the world, desires are world states(goals) that the agent would like to make true and intentions are the formalized plans to achieve any desires [65]. This approach increases believability as agents must be situated in order to make their own choices, as well as have observable intentions. The side effect is an increased computational cost as agents must maintain facts about the world

Figure 2.7: Agent Sense-Think-Act cycle



they operate in [65].

In implementation, BDI agents use a mixture of the AI techniques to perform their decision making. For example, it is common for BDI agents to use explicitly defined world states and a planning system much like a GAOP to dynamically solve for goals. However, the way in which goals are selected could be based on rule-based reactions to the environment as seen in Bates' Lyotard [9], or based on expected utility of achieving the goal as seen in [3, 29]. It is helpful to think about BDI agents having a set of processes that is implemented using a variety of decision making techniques.

For example, COG-AFF is an affective architecture built of three process layers including reactive, deliberate and a meta-management layer [71]. Each layer has the capability of interrupting or modifying the actions of other layers. The reactive layer uses rules (or scripts) to match observed events to both actions. These are used for a variety of tasks including modeling emotional reaction to an event. The deliberate layer uses a rule-based system to create plans, affect working memory and consider possible outcomes. Finally, a meta-management layer is responsible for self-observation and self reflection.

Another affective architecture is the COGNITIVA architecture that is composed of three layers being Social, Deliberate and Reactive [32]. Events are perceived by each layer which in-turn may affect other layers by modifying shared values or mediating plans. For example, the reactive layer is responsible for immediate reactions to events such as one that is pain-inducing, based on rules. For example, if pain reaches above a threshold, the reactive layer attempts to remove the agent's body from the source. However, the higher layers may affect this threshold by lowering it or raising it, based on the importance of its goals. As a result, the layers of the COGNITIVA architecture interact with one another in shaping the actions that are selected.

An example of a rule-based reactive system is Bates' Tok [9]. This architecture is composed of two main modules named Hap and Em that affect one another to create believable behaviour. Hap is a goal-directed reactive action engine. It continuously chooses the agent's next actions based on its perception, current goals, internal state and behavioural features. Em is ToK's emotional module that uses predefined emotional reactions to particular event types and may affect HAP, by changing the importance of goals. As a result, Tok may create believable behaviour through the use of rules and scripts to change the state of the agent as well as predetermined actions.

The architecture first proposed by of Bailey and extended by You provides dynamic and data-driven approach [5,87]. Here agents are composed of roles, emotions and personality models combined with scripts to create believable reactions and emergent behaviour. Included in this architecture is the ability to integrate many emotional and personality models creating flexibility in design while being computationally efficient. However, the main weakness of this approach is the scripted nature of the agent processing does not allow the dynamic additions of roles, learning or planning with a care of consequence. Consequently, agent's using the architecture of Bailey or You may not believably react to events.

A more complicated approach to create believability in the architecture of Marsela and Grathces' EMA [26,27]. This architecture is based on Lazarus' Appraisal Theory that incorporates an explicit representation of beliefs, desires, intention, plans and probabilities. Events result in causal interpretations that in-turn are evaluated by EMA to affect working memory. EMA may also affect decision making by using decision theoretic planning that assigns utility to individual actions based on a number of features such as beliefs, desires, perceived probabilities and emotional state as well as a the utility to others. However, as this approach is more interested in simulating the emotional and cognitive processes, it appears to be too computationally expensive for today's video games.

Indeed, many of the architectures described are complex, layered and designed specifically for believability [3, 29] while others attempt to replicate human cognitive processes [13, 26, 27, 83, 84].

Having discussed the general architectures of agents, in the following section will discuss the use of psychological models to affect the decision making processes of NPCs.

2.4.1 Psychology Foundations

In the search for methods to create believable agents, most state-of-the-art techniques have turned to psychological models of humans to provides a foundation for NPCs. This section will outline some important models of personality, emotion and how they have been used within agent based architectures.

2.4.1.1 Personality

Personality in agents is defined as the agent's pattern of behaviours or interactions with its environment [68]. Personality can be seen as a reflection of how events are appraised, the goals we choose and we go about achieving them [9, 19, 27]. For example, if a pen has gone missing from a person's desk, the owner could choose to blame someone else for having

stolen it, or blame themselves for having been careless. Here, personality changes the way events are appraised. Personality may also be seen as a reflection of moral beliefs and personal standards [34]. Many models of personality exist within the realm of psychology. This section discusses the important models of personality that have relevance in creating fundamental traits for NPCs [16,55].

A recent theory explains personality as the motivations founded in the biological basis of both humans and animals. According to Reiss, 16 Basic Desires motivate action [66]. These desires can be seen in Figure 2.8 . Reiss argues that a person can have reason to behave without necessarily being aware of it. Furthermore, instead a simple characterization of personality, Reiss provides in greater detail the structure and modulation of these motivation through action. Here Reiss assumes the following about basic desires [66]:

- A person is made up a unique valence of the 16 basic desires which individuals prioritize differently. Generally, unusually strong or weak desires are used to characterize an individual. For example, someone who is "Power Hungry" will have normal traits except for power.

- Each basic desire is regarded as a continuum of potential motivation anchored by opposite values. An individual aims for a set point along this continuum. This would result in someone who is "Power Hungry" to have a set point at the upper end of the power continuum.

- Actions performed whose result satisfies the desire or reaches the set point results in a form of joy. This feeling is temporary and the desire will reassert itself and needs to be satisfied again.

- Motivation is a result of the difference between the set point and the current amount of intrinsic satisfier felt. The longer desire remains unsatiated, the stronger the motivation to perform actions whose results satisfies the desire.

Figure 2.8: Reiss's 16 Motives

Motive name	Motive	Animal behavior	Intrinsic feeling
Power	Desire to influence (including leadership; related to mastery)	Dominant animal eats more food	Efficacy
Curiosity	Desire for knowledge	Animal learns to find food more efficiently and learns to avoid prey	Wonder
Independence	Desire to be autonomous	Motivates animal to leave nest, searching for food over larger area	Freedom
Status	Desire for social standing (including desire for attention)	Attention in nest leads to better feedings	Self-importance
Social contact	Desire for peer companionship (desire to play)	Safety in numbers for animals in wild	Fun
Vengeance	Desire to get even (including desire to compete, to win)	Animal fights when threatened	Vindication
Honor	Desire to obey a traditional moral code	Animal runs back to herd when stared at by prey	Loyalty
Idealism	Desire to improve society (including altruism, justice)	Unclear: Do animals show true altruism?	Compassion
Physical exercise	Desire to exercise muscles	Strong animals eat more and are less vulnerable to prey	Vitality
Romance	Desire for sex (including courting)	Reproduction essential for species survival	Lust
Family	Desire to raise own children	Protection of young facilitates survival	Love
Order	Desire to organize (including desire for ritual)	Cleanliness rituals promote health	Stability
Eating	Desire to eat	Nutrition essential for survival	Satiation (avoidance of hunger)
Acceptance	Desire for approval	Unclear: animal self-concept?	Self-confidence
Tranquility	Desire to avoid anxiety, fear	Animal runs away from danger	Safe, relaxed
Saving	Desire to collect, value of frugality	Animal hoards food and other materials	Ownership

As example of Reiss' theory at work, consider the following scenario involving a character named Jake. If Jake feels lonely (where the social contact's set point is far from the current level). The longer that social contact is absent, the greater the motivation for social contact will be felt. At some point the motivation will take over and he will decide to take action and perform actions that satisfy the desire. If, after some time, Jake's desire for social contact has been saturated(social contact is above the threshold), he will take actions to seclude him by going home. In this example, we see how actions are a result of a personality model that is driven by internal drives.

While an interesting personality model that gives insight into human and animal decision making being motivated by desires, it only provides the basics into their working and does not consider how social norms, emotions, other how outside variables may effect the desires. For example, assuming one is lonely, an emotion such as sadness may impede actions that would satisfy this desire. A beautiful woman present in the room may decrease the intrinsic variable for Romance, hence increasing the desire for romantic actions. As a result, Reiss's theory of Basic Desires could provide a basic outline for action motivating agents, but much work needs to be done in integrating it into a model for cognition and action. Furthermore, as of the time of writing, no known implementations of the model exist.

A very important personality model is the Five Factor Model(FFM) often known as OCEAN [34,36]. This model characterizes personality among five distinct dimensions being Openness, Contentiousness, Extroversion, Agreeableness and Neuroticism: [36]

The FFM is a set of distinct traits based on factor analysis [36]. However, the FFM model of personality does not outline how traits affect actions; rather it is a characterization of personality. Interestingly, the personality of a loved one can be described in much greater detail as the quirks, and idiosyncrasies become important descriptors. And so, it is within this context that we realize that the FFM fails to fully describe one's personality. As a result, it needs additional descriptors in order for gain the robust description of personality as required for believable agents.

Despite its faults, the FFM has been implemented in many computational models. Generally, each dimension of the model is represented as a scalar value(0..1). These scalar values may be then used to affect a number of agents' processes including Goal Selection, Action Selection, Planning, Event Appraisal, Emotional Responses [1,19,34,36,70]. For example, personality scalars may be used to bias calculations in the agent's mental processes. Planning and Action selection may be biased by using a utility function that considers the

personality traits for certain actions [19, 70]. Greeting another agent, for example, will involve the extroversion trait. Higher extroversion will result in selecting grander, louder expressions [1, 19, 70]. Neuroticism may bias the interpretation of an event by amplifying the emotional reaction of the agent [19, 70]. As a result, the FFM can be seen to affect the way a character behaves. Some implementations may change personality over time as a reflection of the types of actions they choose [70].

The following section examines the use of emotions in agent-based architectures.

2.4.1.2 Emotions

Since Aristotle, emotion has been discussed as a central component of human reasoning. Emotion clouds reasoning and affects our every decision. More so, compelling arguments have been made suggesting that emotions are a central requirement for intelligence [17,59, 71]. Minksy captures this question well [48].

> "The question is not whether intelligent machines can have emotions, but whether machines can be intelligent without any emotions?"

The importance of Emotions was also noted by Loyall in Section 2.2, who defined them as a requirement for believable NPCs. More so, NPCs must use both verbal and non verbal cues(gaze, pose, actions) to alert the audience of the agent's inner state [24,41]. As a result, it seems intuitive that a cohesive emotion model would be used to generate these cues [8,24]. Emotions have also been noted as an important factor guiding social decision making [18, 35]. Another important characteristic of emotion is that it "interrupts and redirects attention (usually with accompanying arousal" [1, 32]. This attention is important as it can aid with decision making and goal selection. Interestingly, the quest for a biologically distinct set of emotions has not proven to be fruitful [26, 71]. Much like personality, many models exist with a wide array of difference. In developing believable NPCs, a central

context dependent as the action tendencies are a result of the appraisal of an event [24,70]. Guilt for example, is a complex emotion as it is the feeling of displeasure that urges us to either relentlessly undo our own actions or in contrast, remain paralyzed. The action tendency from guilt is dependent on how the event is interpreted. Emotions can be seen as either primary or secondary on the basis of the action tendencies it produces. And so, emotions vary from basic to complex with action tendencies depending on the interpretation of an event.

Having examined the concept of emotions, we will now describe how they have been modeled within agent-based architectures.

2.4.1.2.1 Implementation of General Emotional Models In most of the papers surveyed, individual emotions are represented as single scalar value [3, 9, 26, 27, 49, 70, 85]. Joy for example could be represented as a continuous value between zero and one. Here, a character is joyful is the value of the scalar is near one and not joyful if near zero. Another option is to create composite emotions that represent the combination of two emotions on a continuous scale across a negative positive scale [55]. Here a Joy-Sadness scale could be represented as range from negative one to positive one. The problem with composite emotions is that not all emotions have an opposite. For example, complex emotions such as jealousy do not have an opposite. Interestingly, implementations of emotions use either a single values or a composite scheme. None use a mixture of both schemes. Scalar emotions allow for the a simple update routine to modify its value by adding or subtracting to its current state [9, 26, 27, 49] by using a logarithmic scale [3]. Another approach to represent emotion is to represent a range as a discrete set of values ranging from low to high [78]. Emotions have traditionally been represented in simple forms.

As psychologists have yet to present a single cohesive all-encompassing model for emotions, NPC designers tend to choose the number of emotions to model based on the char-

acters and situations they are trying to create and as a result, no singular unified group of emotions is present within the literature. Among the popular emotional models are Eckman's Six Universal Expression [8, 49, 70, 85] and the OCC Model [3, 26, 27, 34, 56].

Eckman's model allows NPCs to increase their believability by using real time facial expression software that is often based on this universal model. Furthermore, this ensures developers do not need to create an arbitrary mapping onto Eckman's that suffers in fidelity and computational cost. One strength of this model is that the reduced number of emotions allows for a reduced complexity in design and computational resources. Eckmans Emotions include Anger, Disgust, Fear, Happiness, Sadness and Surprise.

Another popular cognitive model that incorporates emotion is the OCC that has a total of 22 emotions [1, 3, 9, 26, 27, 56, 79]. Interestingly, the OCC model of emotions is often criticized for its complexity and overlapping emotions [8]. In implementation, the number of emotions is highly reduced [27]. In light of this, one of the creators of the OCC model, suggest a reduction in number of emotions to a simpler scheme using only five composite emotions [55]

Other general emotional models discussed in the NPC literature are Sloman's [71] tertiary emotions and other arbitrary models design specifically for the task [58, 74].

Much like personality, emotions have been used to affect a number of processes within agents including Goal Selection, Action Selection, and Attention. The action tendencies which emotions create are generally implemented by using a utility function during planning [34, 70, 85] or from a rule-based perspective which selects pre-computed plans(behaviours) based on the emotional thresholds in the agent [3, 9]. Utility-Based planning with emotion considers the emotional level of particular emotions in deciding if the action should be used. Rule-Based systems use thresholds or pattern matching to decide which actions to use [3]. Emotions are often used in Goal Selection as it facilitates which goal should be in focus [32, 79]. For example, if an agent is highly fearful that a

goal will not be achieved due to an event in the environment, the high emotional response allows for the agent's internal reasoning to place this goal as its top priority [3,27].

We will now investigate the use of appraisal theory and its implementation.

2.4.1.2.2 Appraisal Theory and Its Implementation As discussed above, many important implementations use Lazarus' appraisal theory [26,27]. Lazarus' core contribution Appraisal Theory defines the processes of emotion. Appraisal theorists argue emotions are evoked from two continuous tightly coupled processes being appraisal and coping [26,27].

Much like in Frijda's work [24], appraisal occurs as a result of an event being interpreted based on a person's beliefs, desires and intentions. This is a reflexive and automatic response to the event that does not require deliberation and a result, and, as a result, generally does not require sophisticated reasoning and can be implemented as a simple mapping [27]. However, if the complicated reasoning is present for an event, the appraisal stage may utilize it. The result of the appraisal stage is that a set of appraisal variables are evaluated which gives a distinct characterization of the event. Appraisal variables should minimally include concepts such as relevance, desirability, causal attribution, likelihood, urgency, ego involvement and coping potential [26]. These appraisal variables are then used during the coping stage to formulated responses and/or emotions.

The coping stage is an ongoing process that is characterized by how one responds to an event and uses one current state plus the appraisal variables. Consider an event such as a man entering your room in the middle of night. If your appraisal of the situation is that the man is a robber(hence high relevance, urgency, likelihood and low desirability), your ability to cope with the situation forms your emotional response. If you are prepared for such events(have gun at your pillow), your emotional response will be much different then if you have not made such preparations. Your ability to maintain or change the

situation in respect your goals has a significant impact on the emotions that result from an event. Furthermore, realizations you may make at sometime in the future may also have an impact on your emotions. For example, imagine you suddenly realized that the man is not a robber, but rather your son who said he would check in after his night out. Your deliberation on the event results in a change your interpretation and as a result, your emotional reaction. Deliberation may be implemented as continuous execution of rules within a rule-based system [27].

Coping can be characterized into two broad classes being *problem-focused* coping and *emotion-focused* coping. Problem focused coping generally results from strong emotions and forces a person towards strategies and actions that deal with the situation that caused the emotion [26, 27]. Marsella and Gratch implemented this as a planning problem using causal reasoning. Here, an inference engine attempts to deal with an event by creating plan to overturn the situation. If the plan's utility is high enough, the plan will then be executed. However, when the utility of the best plan is low, the agent turns to emotional-focused coping. Emotion-focused coping is alteration of one's interpretation of an event [26, 27]. This could result in the reduction to the importance of a goal or changing the blame of an event to someone else. As a result, the situation is reinterpreted which will evoke emotions in the process. In closing, coping is as powerful mechanism as it allows for events to have a variety of impacts on the state of the agent.

In implementation, coping strategies maintain desirable or overturn undesirable in-focus emotion instances [27]. Many strategies are implemented such as taking action, planning, getting help, procrastination, positive re-interpretation, acceptance, denial, mental disengagement, and shifting blame [26, 27]. These strategies are selected based on hard coded rules given the internal state of the agent. That is, particular events may evoke predefined coping strategies in parallel. In these cases, the preferences for particular types of strategies decide which strategy will be executed. Unfortunately, the implementation of Lazarus' ap-

praisal theory has tended extremely computationally expensive as the processing has been done to mimic human processes, not create believability [26, 27].

2.4.1.2.3 Integrating Emotions and Personality with actions As both emotions and personality play an important role in the decision making of agents, few implementations explicitly express utility of an action as a combination of emotion and personality [5, 34, 70]. Even an intuitive examination of action tendencies leads one to think that personality and emotions play a direct role in deciding which action is selected. For example, if you are calm, the likelihood of yelling at someone who knocked over your coffee is based on your personality traits more then your emotional state. However, if you are angry toward this person, the likelihood of your yelling increases. As a result, it could be described that specific emotions amplify specific personality traits that in-turn affect the types of decisions chosen. Other approaches have decreased and increased personality factors based on emotional reactions [70]. Romano for example, used negatively appraised events to reduce a FFM personality factor such as extroversion [70]. Indeed, recent research hints to a biological link between action, personality and emotions [14]. However, there is little other research that provides a clear link between any two models of personality and emotions [34]. As a result, the implementation of Johns' utility calculations combining the OCC model of emotion and the FFM is arbitrarily derived [34]. The advantage of this system is that it allows for efficient planning based on both personality and emotions.

Emotional Models have been implemented many ways with many features of human emotion being simulated by either using rule-based processes or Utility-Based approaches. As a result, the believability of agents increases. What follows is another important aspect of believability: Understanding Social Knowledge.

2.4.2 Sociology

Humans are social beings, and our social relationships, culture, values, and norms all play an important part in defining who we are. As a result, believable agents must be socially situated, and able to reason in social terms [18, 27, 29, 45, 70]. Agents armed with social knowledge will have greater believability as they interact with players and NPCs [29, 70]. More so, emotion and personality do not describe important aspects of human behaviour including social constraints, moral standards and routine behaviours [29]. Like personality and emotions, social models are still largely undefined by the community [29]. Furthermore, social agents are particularly difficult to create as they must reproduce behaviours that are complex, reflecting a number of intricacies of human behaviour [29]. This has led to a number of creative experiments attempting to create believable characters. In this section, we will examine how social knowledge has been thought of, structured and embedded into socially situated agents.

An important social element within NPCs is social relationship. While they may be simply defined as a link between two people, social relationships are clearly identified with additional information including social variables, emotions, and roles. Isbister notes that relationships are in part defined by two primary social variables being agreeableness and dominance [33]. Dominance is tied to the concept of social power. Dominance helps define the type of relationship as friendly, unfriendly, and neutral [33]. Bailey and Katchabaw implemented Isbister's social variables through the through a rule-based emergent social system that allowed for the varied interaction among its agents based on their social variables, emotional state and personality factors.

Social relationships have also been bolstered with quantitative social emotions in order to give feeling to the agent's relationships. The OCC cognitive models define relationships with variables such as: [70]

- The degree to which the agent likes to other
- The degree to which the agent dislikes the other
- The degree to which the agent has formed a cognitive unit with the other
- The degree to which the agent is familiar with the other.

Several systems have implemented the OCC model of relationships. Some use this existing information in the causal attribution of blame for an event [27], as well as defining when another is friend or foe. These social emotions may update using a rule-based approach during the appraisal of the event. If an agent is perceived to have caused a harmful event, a rule will convert emotional reaction such as fear to an increased dislike for the character [26].

2.4.2.1 Roles

Perhaps the most exciting view of society and social relationships is through the concept of Roles. Based on Role Theory, a role defines the relationship of person to another person, group or even object [10]. Roles are often thought of a concept used in theatre, where an actor plays an assigned part [10]. In everyday life, a role is intuitively defined. The role of mother for example, defines a relationship between a woman and to a child or children. However, the role does more then just formalize the relationship, it also brings together desires(goals), intentions, beliefs [57]. Applying this to the creation of believable characters, roles provide a formalized description of what is expected from an agent [29,57]. This provides significant advantages for designers as roles can be reused, and are intuitively understood by non experts [29,57]. Let us quickly outline what the various components of a role from a BDI agent perspective:

- *Desires(Goals)*: Are associated to a role and represent a goal or state that should be met. The role of a mother for example will have goals such as keeping children

fed, ensuring their safety as well as nurturing their intellectual growth. The role of a military officer may have a goal to be at work for a certain hour of the day.

- *Beliefs*: Are mental attitudes characterizing how the world is viewed through the role [57]. These may be composed of values, norms and a world view [29]. Values indicate actions and ideals that must be adhered to [29]. For example, a monk may believe that the act of killing is not permissible on any grounds. This would require monks to find other ways to achieving their goals. However, a soldier would not have such objections. Interestingly, Panzarasa argues that some values should be optional [57]. This means that not all monks completely adhere to their own rules, which in-turn allows for greater flexibility for personality to be expressed. Social Norms may represent rules that should be adhered to. Wearing a tie for example when at work is an example of norm for the role of a businessman. Another form of belief is a Worldview that defines certain relationships among roles. This "typefecation" may produce generalizations onto the world such as "He is a man, then he is strong" [29]. Role-based beliefs encompass a specific characterization of how a person thinks when in the role.

- *Intentions*: Are future directed states the player wishes to bring about [57]. Within a BDI framework these may be defined as plans.

As roles tie together many aspects of social knowledge into a comprehensive set, they appear to be an optimal way to organize an agent's social behaviour. In addition, from a designer's perspective, roles are very re-usable as they can be defined independently of an individual agent [29]. Roles are an important concept to design social knowledge that in-turn aids in the creation of believable characters.

Interestingly, many roles can be applied to an individual. A woman may be a mother, a worker, and friend, all at once. Consequently, it may not be possible to always meet the

requirements of all the goals for each role. This is described as a state of *role overload* [57]. It is within this state, that roles become truly interesting from the perspective a story. Story telling has often used dilemma to evoke emotion from the audience [7]. We can empathize with characters having to make decisions between two mutually exclusive goals. For example, Romeo in Shakespeare's play is torn between defending his family honour and his love for Juliet as he tries to stop Tybalt and Mercutio from killing one another. As a result, role overload could play an important part of interactive storytelling where NPCs struggle with difficult but believable decisions.

Role theory may also help with the development of emergent narratives. Recent research in multi-agent systems suggest that using roles as a central part of the agent architecture can enable increased communications and situatedness, dynamic problem solving, predictable, stable and reliable behaviour all within an emergent environment [51–53, 73, 86]. This conjunction of roles and story has not been explored in any literature surveyed.

While [29], defines roles between two agents, roles can be extended to be applied onto groups. For example, the role "Enemy" need not apply to each individual enemy. Instead, the role can represent a class of individuals in the form of a group. In-turn, all perspectives taken by the agent onto the group members will be unified. A role can also help define group memberships by establishing the positions within a group [51]. Roles can contain the information that defines how one relates to others in a group. For example, someone taking the role of General knows that each group member within the group is supposed to obey to him in the form of values. Also, the role can contain concepts such as group how structures are composed. As a result, roles can define a wide variety of group dynamics that in-turn can be applied onto individual agents to provide social knowledge about the group.

As for the implementations of roles, [29] provides the only implementation and discussion of the structure and decision making in a role-based system for NPCs. His agents

include several modules built around the concept of roles. Here an agent may have multiple roles, each having an associated importance value. Goal Selection occurs using a utility function that ranks each role-based on the concern for a goal multiplied by the importance of the goal. This provides a simple yet effective mechanism for simulating attention and goal selection. Furthermore, this provides flexibility for designers as they can create a social structure defining the importance of social relationships.

2.4.3 Story Concerns

Interestingly, in creating smarter more dynamic believable agents, we have introduced a strong contradiction within the video game system [2]. Story, the video game narrative, has traditionally been well structured with a predefined beginning, middle and end. However, injecting fully autonomous believable agents into this environment destroys the traditional story as their group behaviour is more akin to an emergent system, where the actions of each agent become a symptom of the actions of other agents and the initial system state. As a result, many researchers have attempted to *rein in* their agents through a variety of techniques. In other words, dynamic believable agents are capable of doing things that were not anticipated when the story was written, potentially causing problems, inconsistencies, and paradoxes in the story.

The most popular technique among those surveyed is through the use of an AI drama manager(DM) or director whose responsibility is to observe and manage the interaction among agents. Generally DMs can be categorized by the method they attempt to guide the story. These methods include treating the story as a optimization or planning problem. The solution to the problem involves identifying the current plot state and a desired state and then identifying which actions could occur next [43, 44, 46, 67, 69]. Most DMs have a host of abilities to affect the decisions made by the agents ranging from intrusive action selection to proscriptive action denial [43, 69, 78], or to a motivational approach such as

adding goals to an agent [78]. These approaches have allowed for players to increasingly shape the world they play in while maintaining the coherence of story.

2.4.4 Summary of Agent Architectures

Today's NPC Architecture is built around an Agent-Oriented Design where all decisions are made through the eyes the agent. In order to improve the believability of agent decision making, many models include an agent's personality traits, emotional state as well as a precise knowledge and understanding of social relations. This in-turn has resulted in increasingly believable agents at the cost of computational expense.

As video game narrative is an important aspect of video games, designers of NPC architectures need to keep this in mind as they create decision making processes.

2.5 Summary of Related Works

With the problems of photo realistic real time rendering largely solved [15] and increasingly powerful platforms available to game designers, good AI is increasingly becoming an important part of video games [15, 54, 69, 75–77]. Players expect AI that is dynamic, able to react to unexpected events and behave believably [19, 75–77]. As somewhat of a consequence to this, NPC AI is an important and active area of research [3,26–29,34,54,82].

Creating believable decisions is however a challenging endeavours. Human judgment is both relative to their own culture but also based on the complexities of human behaviour and social interactions. As a result, designers who wish to create believable behaviour need to respect the requirements set out by Loyall [41].

Traditional techniques involve modeling behaviour off-line and then using simplistic finite state machines, and scripts to select the appropriate behaviour at run time [63, 82]. Newer techniques have traded an increased computational expense for dynamism, flexibility [3, 26, 27, 29, 34, 54, 63, 82] and reuse [63, 82]. These approaches require behaviours to be decided at run time through state search techniques. Among these techniques are A*, Rule-Based Systems, Goal Action Oriented Programming and Utility-Based Planning. However, these techniques fail to properly capture important aspects of what makes characters believable.

The state-of-the-art in the creation of believable decision making has reached out beyond computer science using models from Psychology(personality, emotions, appraisal theory) and Sociology(social appraisal variables, relationships and role theory) into both traditional and search based techniques. Personality models such as the FFM have been implemented within a number of agent processes including Goal Selection, Action Selection, Planning, Event Appraisal, and Emotional Responses [1, 19, 34, 36, 70]. This has been accomplished through role-based processes [5] as well as Utility-Based processes [19, 34].

Another important model discussed was Reiss' Theory that explains underlying human motivation and how action may modulate desires. A number of emotional models have also been used including the Eckman's Universal Emotions and the OCC model. These have been used for a number or processes including Goal Selection, Action Selection and Attention. Social Knowledge has also been used to aid the decision making process of agents. Here, social variables and roles have been used to characterize the behaviour of agents. Roles provide an excellent reusable framework that has many desirable aspects for organizing behaviour within video games.

Chapter 3

Design

In order to create believable agent behaviour, maximize flexibility and re-use for game designers, our design is a multifaceted approach incorporating Psychology, Sociology and Appraisal theory into a highly reusable data driven architecture. Explicitly, we define a novel role-based architecture that incorporates an agent's emotional state, personality factors, social knowledge and perceived consequences to a influence the agent's Utility-Based cognitive processes and behaviours. This unique agent architecture should produce believable agent behaviour while being both flexible and highly re-usable. This design contributes to the state-of-the-art by:

- *Integrating Role Theory, Appraisal Theory and Utility-Based decision making into a single Data Driven Agent.* With the agent's roles defining a complex relationship to the environment, the integration of Appraisal Theory provides the processing required to believably interpret and be affected by external events, choose goals, and consider the consequences to possible actions. The integration with Utility-Based decision making creates an agent that can both differentiate between any two goals or actions, but also provides the mechanism in which an AI drama manager could reject a certain action while choosing another. As a result, the architecture

integrating Role Theory, Appraisal Theory and Utility-Based decision making should allow for both believable acting and reacting.

- *Personalized roles.* Typical definition of a role includes information related to Desires, Beliefs and Intentions [29, 57] . This architecture extends the state-of-the-art by *personalizing* roles through their integration with emotions and personality traits. This novel approach allows an agent's internal state to define the importance of a role and its intrinsic desires, but also shapes intentions through Utility-Based planning/action selection and appraisal of consequence. As a result, the believability of an agent's actions is defined through the way roles are affected by their current personality traits and emotional state. Finally, the personalization of roles allows for re-usability of roles as once defined, can be applied to many agents.

- *Automatic Role Integration.* This is a novel and important aspect of the architecture as roles should be written once and then be re-applied to many agents who personalize them through their emotional and personality characteristics. Creating an architecture that supports the *plug and play* of roles is extremely useful in minimizing the work of game designers. Furthermore, automatic role integration allows an AI drama director to shape a story by adding/removing roles to agents. As a result, automatic role integration is an important contribution of this agent architecture.

To achieve these contributions, the architecture is designed with the separation of data from processing. This creates an agent whose roles, personality and emotions define its data and with Appraisal Theory and Utility-Based decision making forming it's processing. Figure 3.1 illustrates a high level architecture of the agent and its interaction with the world or an optional AI drama manager. As illustrated, the agent's processing features five stages that use the agent's data to produce a list of actions. In the case where a drama manager exists, an action negotiation may take place. Finally, an individual action

is selected and executed within the world that would then be broadcast to other agents including the agent itself.

The high level conceptual processing (Section 3.1) is now outlined in order to provide the reader with a sense what is required from the roles, emotions and personality described in (Section 3.2). Finally, (Section 3.3) provides a detailed description of each stage of the agent's mental processing highlighting its strengths and novel integration of Role Theory, Appraisal Theory and Utility-Based decision making.

3.1 High Level Process

The conceptual processing of the agent architecture as shown in Figure 3.2 is composed of five main stages being the Relevance Filter, Appraisal, Coping, Goal Selection and Planning/Action Selection. Excluding the Relevance Filter that is described by Bailey [5], the inspiration for the later processes can be found in the theoretical works of Lazarus on Appraisal theory, and its translation into a computation model by Marsella and Gratch [26,27]. However, beyond the conceptual processing of each stage, this architecture differs greatly in that the core elements of architecture are centered around roles, where as Marsella and Gratch has no such features. The conceptual processing is then composed of:

- *Events*: Are observable changes that occur in the game world and are interpreted by an agent. Events are tagged with relevant information such as where the event occurred; who caused the event; what was the agent's internal state (emotions, personality traits); what action was taken and what were the consequences. Events should be broadcast to agents given constraints such as the distance from the event.

Figure 3.1: Agent Architecture

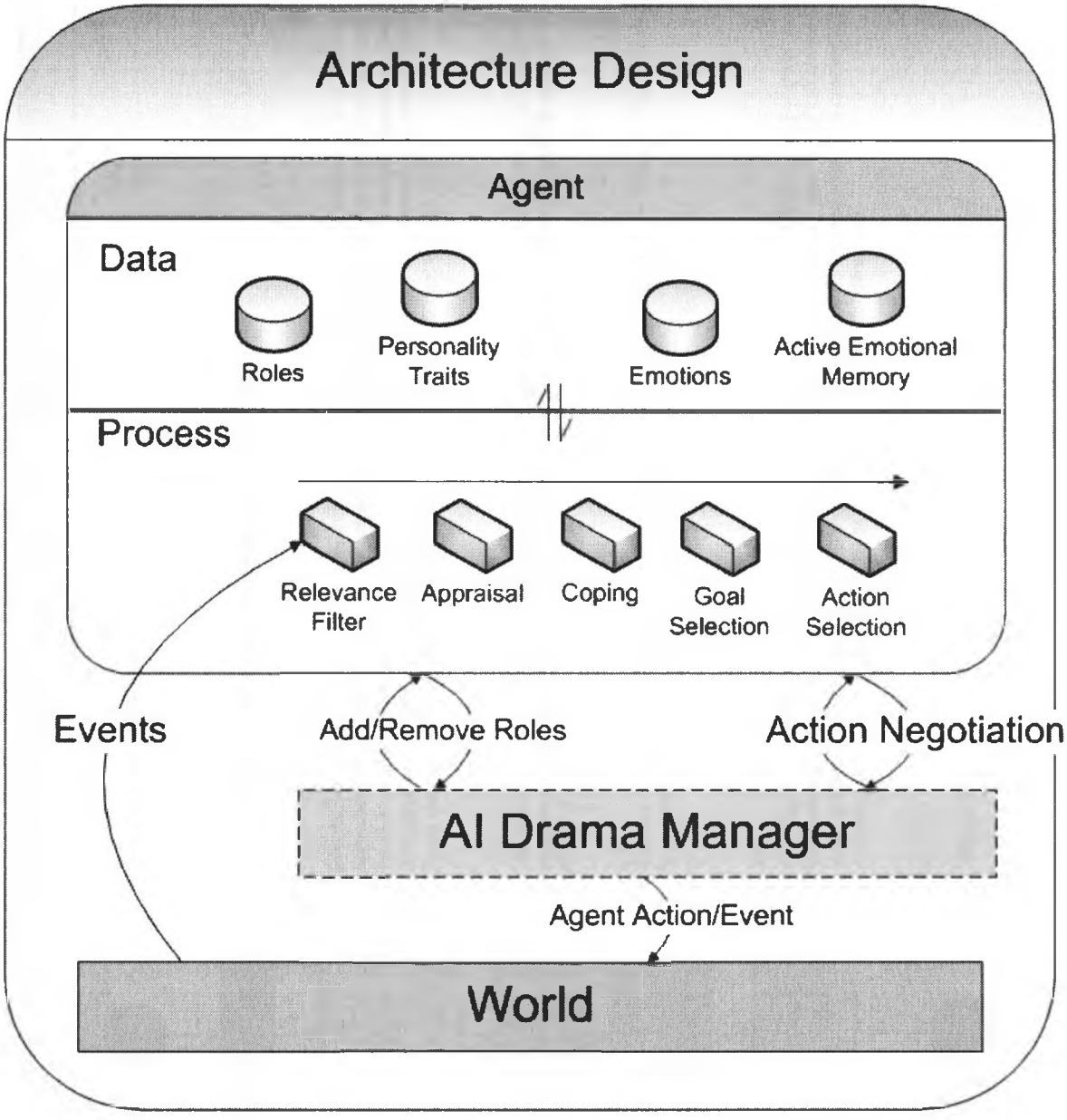


Figure 3.2: High LevelProcessing

- *Relevance Filter*: Once an agent has sensed the event, the relevance filter uses role information to discover if the event is of importance and needs to be further processed by the agent. This allows agents to discard world events that have no relevance to the agent. This stage aids in reducing the computational requirements of the agents. These interest events are then passed onto the appraisal stage. The Relevance Filter is further discussed in Section 3.3.1.

- *Appraisal*: Following the relevance filter, interest events are passed onto the appraisal where their emotional importance is identified in relation to role specific concepts such as Beliefs, Desires and Intentions. If an event is deemed relevant, the appraisal process will create an emotional memory combining both the event information and an initial emotional reaction. Finally, the appraisal stage should add an emotional memory into the *Active Emotional Memory* that defines the agent's mood. The Appraisal process is further discussed in Section 3.3.2.

- *Coping*: The agent, informed of world events, now considers how the emotional memories can be dealt with. In instances where a negative appraisal of an event is formed, Gratch suggests that this process should have specific coping strategies [26, 27]. These strategies may include problem-focused coping where the events may be overturned through action or emotion-focused where events are re-interpreted. Coping is discussed in detail in Section 3.3.3.

- *Goal Selection*: At each frame, the agent must decide which goal to pursue. In this architecture, the agent's internal state (emotion, personality) and possible consequences to actions drive the goal selection process through a utility-based process. This deliberation produces an *active goal* that reflects the agent's state and roles. The Goal Selection process will be further discussed in Section 3.3.4.1.

- *Planning/ Action Selection*: The final stage before an agent performs an action is

the planning process. Here, a plan described by a series of actions is selected that will fulfill the requirements of the active goal. However, the plan is biased to select actions that reflect the agent's internal state, its roles and possible consequences to the actions. The approach to the formation of plans is novel to the architecture through their integration of many of the agent's parameters into a single utility function. The result, plans are personalized given the agents state. The Planning and Action Selection process is discussed in greater detail in Section 3.3.4.2.

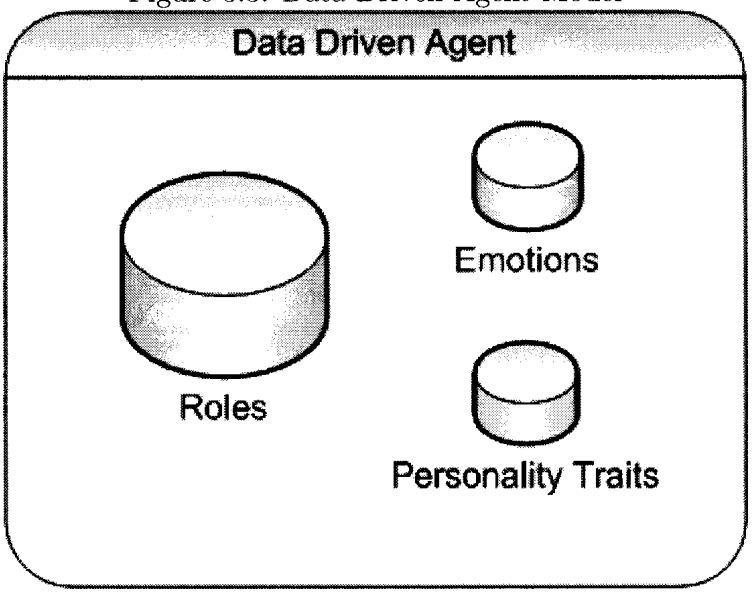
Finally, when an action is passed to the world, an Event is created and is tagged with relevant information that agents can use to understand the event.

Having highlighted the high level general processing, the role-based agent architecture will now be defined so that the integration of Role Theory, Appraisal Theory and Utility-Based Decision Making can be clearly discussed in Section 3.3.

3.2 Role-Based Agent Architecture

In order to create maximum efficiency and reuse, the agent model should maximize its use of the data driven paradigm [5]. Here, the agent's data(or parameters) are defined independently of its processing. This allows a designer, or an AI Drama Manager to create new behaviour simply by changing the agent's internal state. To achieve this, we concisely define our agents as being a list of three types components being the current Roles, Emotions and Personality Traits as seen in Figure 3.3 . Throughout the remainder of this thesis, the agent's *current state* should be understood as this list. There are significant advantages to this role-based architecture. One advantage to this approach minimizes the required memory use by only needing to store relevant data within the agent. Each role will define what emotions and traits it requires. Consequently, only relevant information needs to be updated within the agent. Furthermore, this approach allows for flexibility

Figure 3.3: Data Driven Agent Model



in agent behaviour as the emotions, personality and roles clearly define agent. Also, once roles,

In the following sections, the design of emotions (Section 3.2.1) will be discussed followed by the agent's personality (Section 3.2.2) and finally the design of the roles (Section 3.2.3).

3.2.1 Emotions

In this section, the components of the emotional subsystem are defined. This includes the individual emotional and traits their typefication into basic and role-based emotions. Finally the agent's mood is defined through a set of active emotional memories.

As discussed in the previous sections, emotions play an important role in directing the decision making of humans and are an important requirement of believability [17, 24, 41, 59, 71]. To meet this requirement, the agent architecture features a trait-based emotional subsystem that has many advantages. Trait-based approaches are simple in representation

and consequently are well suited for the integration into Utility-Based processes. Another advantage of a trait-based approach is that they are easily updated. Generally updating a single scalar value is all that is required. Trait-based emotions are excellent as they can be modeled once and instances can be reused in agents. For instance, once Anger is defined, it can be easily assigned to any agent that requires it. Using a trait-based approach is advantageous in that the architecture is not tied to a specific emotional model such as OCC's 22 emotions or Eckman's Six Universal Emotions. The designer is free to choose which emotions to use. Finally, and importantly for our architecture, a single scalar value representing an emotion allows for a very natural integration into Utility-Based processes.

The general structure of an emotion should minimally contain:

- *Range*: Defines the minimum and maximum value of the emotion.
- *Current Value*: The current state of the agent's emotion within the predefined range.
- *Activation Function*: is used to increase the current value of the emotion and characterizes how the emotion behaves under this process. This allows for an agent's emotional characteristics to be modeled. 'Quick to Anger' could be modeled using a single value multiplier or complex function that amplifies the change of the emotion. This definition gives designers an increased ability to model the agent's emotional state.
- *De-activation function*: The deactivation function is the opposite of the activation function as it defines how the current value is decreased.

The types of emotions for the role-based architecture will now be discussed.

3.2.1.1 Emotion Types

In order to accurately represent an agent emotional state within a role-based architecture, two types of emotions are defined, namely being *basic* and *role specific.*

Basic emotions are not specifically tied to any particular role but rather are part of the agent and used in every role to affect decision making. Happiness for example, is a basic emotion that could be used in many roles to influence the planning process. As a result, your interaction with friends or lovers will change depending on how happiness. Consequently, basic emotions should be shared across many roles and integrated into many Utility-Based cognitive processes. Basic emotions have been modeled in many systems through simple representation similar to the general emotion discussed in Section 2.4.1.2. Typically these basic emotions will use a standardized model such as Eckman's Universal emotions. As the architecture is data driven, it provides significant advantage in allowing the designer to choose any emotion they feel appropriate. Simply adding the emotion to an agent and specifying their integration into the process is all that is required.

Novel to this architecture is the definition of role specific emotions. Theses emotions may capture complex characteristics of the role by influencing the Utility-Based decisions with regards to a role. For example, role specific emotions for a Lover role, could include love, jealousy or anxiety. Role specific emotions can also be an *additive* to basic emotions. For example, if an agent's role to a friend includes a role specific anger emotion, the agent's total anger towards the friend can be seen as the addition of both the basic emotion and the role specific one. Role specific emotions have significant advantages over traditional emotional representations as they can be arbitrarily defined and used within the role. As a result, designers can clearly define how one agent relates to another on a number of complex but arbitrary emotional levels. Furthermore, as each role will be responsible for the maintenance of these emotions, their elicitation and influence can be clearly defined within the role. A final advantage of the role specific emotions is that they can be thought of

as a representation of the history between two agents. For example, through an interaction, an agent may become angered specifically with another individual. Keeping track of these emotions provides a type of basic learning where future interactions can be biased. The role specific emotions create a novel and effective mechanism to affect Utility-Based processes such as appraisal and planning in order to create believable agent behaviour.

3.2.1.2 Active Emotional Memory

A final feature of the emotional subsystem is the Active Emotional Memory (AEM) that clearly defines an agent's cognitive state or mood. The AEM is a set of individual *Emotional Memories* created during the appraisal process that represents important events and their corresponding emotional reactions. Emotional Memories will be further discussed in Section 3.3.2. An agent's mood can be seen as the aggregation of these emotional memories as seen in [26, 27]. However, this use of the AEM is different in that the emotional memories are tied to roles and may combine with role specific emotions, to influence many processes such as appraisal, coping, goal selection, and planning and action selection. For example, when generally angry, but dealing with a good friend(defined by a friend role), we are less likely act in an aggressive manner. However, if that person is our enemy and we are already angry, our behaviour may indeed be aggressive. Our definition and use of the AEM combined with roles supports an important type of mental processing not found in other approaches.

The AEM plays an important role in the agent's decision making processes as it provides the basis for the emotional state of the agent. For example, when in a good mood, it is unlikely that you will lash out at a friend. However, the opposite is true when you are in a foul mood. The AEM allows designers to capture this type of decision making in a simple and efficient manner.

The AEM should be composed of rules that define how long events will remain active

in memory, as well as the procedure for their removal. One possible implementation of the AEM could use recent research showing that one is unable to experience emotions much different from their current state [50]. In other words, if you are happy, it is impossible to instantly feel sad. Simply measuring the distance between two emotional states could define the integration of the emotional memory into the AEM. Other simpler approaches could include restricting the number of active memories or requiring a certain emotional valence.

3.2.2 Personality

In order to create believable decision making, the agent's personality must be reflected in their decisions as discussed in Section 2.2. The achieve this, the architecture uses a Personality subsystem that represents an agent's personality. Otrney suggests that a trait-based personality model is advantageous to model agents due to its simplicity, flexibility and re-usability [55]. As no personality model defined within the Psychological community is all encompassing and specific enough to perfectly define and individual, our architecture's personality model is abstracted from any particular predefined model. This is similar to the work of [87] in that *any* personality model can be used with the one restriction that personality trait must be representable as a scalar value. This value is required to ensure the ability to perform utility calculations and bias decision making processes such that the personality of the agent is evident. Novel to this architecture is the definition of two types of personality traits being *general* and *motivational.*

3.2.2.1 General Personality Traits

As most personality models including the FFM and Myers Briggs assign each trait an individual scalar value within a range, it follows that a General personality trait is composed of:

- *Range*: defines the minimum and maximum value of the personality trait.
- *Current Value*: the current state of the agent's personality trait within the predefined range.

As with emotions, this representation is both simple to define and is efficiently integrated into Utility-Based decisions. Furthermore, this allows for a number of traits to be modeled and reused.

3.2.2.2 Motivational Personality Traits

As discussed in Section 2.4.1.1, most personality models can are defined by a collection of single valued traits. Motivational Personality traits define the desirability of actions that satisfy the trait and require both a set point and a current value. For example, Reiss's model of 16 desires as discussed in Section 2.4.1.1, considers one's predisposition for certain actions to be based on a desire that is defined by the distance between the trait's set point and current value. As a result, Motivational Traits require an increased complexity. This architecture assumes that Motivational traits will be minimally composed of:

- *Range*: Defines the minimum and maximum value of the personality trait.
- *Current Value*: The current state of the agent's personality trait within the predefined range.
- *Set Point*: Defines the point within the range where the agent aims to be
- *Desire Function*: Defines how much desire is felt to accomplish actions that satisfy this desire. This function calculates the distance between the set point and current value to define a single scalar value for desire. For example, if the set point is greater than the desire then the desire function should return a positive value indicating the strength of the desire. When the current value is greater than the set point, the

desire then becomes negative and demonstrates avoidance for actions that satisfy this desire.

The inclusion of a motivational personality is novel to this architecture as no known implementations are known to exist. The main advantage is that motivational personality traits combine both an overall propensity for certain action, a current desire and also have a simple representation. Consequently, their representation can be easily integrated into Utility-Based decisions to bias outcomes.

3.2.3 Roles

A central aspect of the architecture is the use of roles to clearly define the relationship between the agent and its environment. To properly define this relationship, roles must include components that affect the processing and behaviour of the agent. Inspired by the work of Guyevuilleme, a role is defined with Beliefs, Desires(Goals) and Intentions(plans) as discussed in Section 2.4.2.1. Our architecture makes a number of significant extensions and modifications to aid believable decision making. First, each role is designed to be tightly integrated with the agent's internal processing using concepts of Appraisal Theory. To achieve this, each component of the role includes functions that are used during the various processes. For example, during the appraisal stage, each of the agent's values includes a function to appraise the new event. Secondly, roles are personalized by being tightly integrated with the state of specific emotional and personality traits. Consequently, roles clearly define the relationship and impulses of an agent over a wide variety of emotional and personality states. This in-turn creates believable decision making and has a critical secondary effect. Once defined can then be re-used in any number of agents simply by accessing their relevant emotional and personality traits. This ability to *plug and play* is both one of its greatest strengths, and a novel feature of the architecture.

Figure 3.4: Role Components

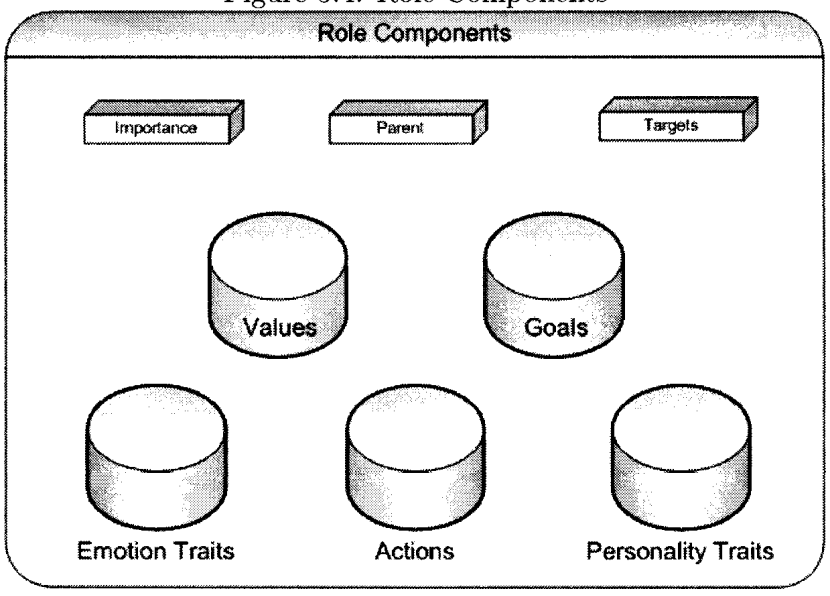


Figure 3.4 presents demonstrates the components of a role. A role is further defined as being composed of:

- *Name*: The name of the role. Friend, Saint, Ninja are examples of particular names.
- *Agent*: The agent who owns this role. This is assigned when the role is added to the agent and allows the role to access specific information such as the AEM.
- *Target*: Who or what is the target of the role. Typically this will be another agent, a group or even the agent itself. The target can be any other object within the world. A sword for example could have a particular role to the agent. The target is assigned to the agent during the role's instantiation.
- *Goals*: The role's goals are world states achievable through actions. Each role should contain actions that can achieve these world states. As discussed in Section 3.3.4.2, planning process uses these actions to achieve a goal. Goals should also be responsible

for generating goal directed emotions as seen in [26,27]. Goals are further defined in Section 3.2.3.1.

- *Beliefs*: Each role can also be composed of beliefs about the world that produce emotional reactions to events and generate information relating to the consequence of actions. Values, Norms or a World View make up the agent's belief system and play important roles in many processes including appraisal and planning. Beliefs are further defined in Section 3.2.3.2.

- *Emotions*: As discussed in Section 2.4.1.2, Emotions should play an important role in shaping an agent's course of action. As a result, the role's emotions need to define how the agent feels towards the role's target. These emotions play a critical role in defining how an agent will interact with the target. Roles should contain both types defined of emotions being basic or role specific. This allows for basic and role specific emotions to additive to one another allowing decisions to incorporate both the agent's mood, but also his relationship to the role's target.

- *Personality Traits*: As with emotions, personality traits play an important part in many processes of the role. As a result, the role needs have direct access to personality traits within the agent.

- *Actions*: The actions defined within the role are *role specific actions* used during the planning process to create a plan. Actions will use the emotions and personality traits to modify their utility during the planning process. Actions are further defined in Section 3.2.3.3.

- *Importance*: The importance of the role to the agent defines at a high level how critical the role is to the agent. This value plays a central role in scaling the strength of emotional memories as well biasing the planning process to select the most important

goal.

Having provided the reader with an overview of the definition of the role, the goals, actions and beliefs are further defined.

3.2.3.1 Goals

As defined above, goals are world states achievable through action. To create believability, goals must be selected based on the needs and current state of the agent. Furthermore, Goals must be tightly integrated into the appraisal, coping and the planning processes. For example, to create believable reactions, the appraisal stage needs to be aware of the agent's goals as events can either aid or cause a goal to be more difficult to achieve. This is similar to the works of [26, 27]. To meet these requirements, goals should minimally contain:

- *Name*: The name of the goal. This could be Attack, Threaten or Court.
- *Target*: The target defines to whom the goal applies. The parent role could supply this information.
- *Preconditions*: Are a list of world states that must be true in order for the goal to be achieved. For example, a precondition for goal of eating food should is that the agent should have food. As a result, if the agent does not have food, the planning process should attempt solve for this precondition by creating a plan.
- *Appraise Event Function*: Is a function that takes an event and produces a goal directed emotional reaction. This is similar to the works of [26, 27] whose agents respond to events that either aid or hinder the achievement of a goal.
- *State Utility Function*: Evaluates the current fitness of a goal with respect to the current state of the agent. Goals are selected when the emotional and personality

traits well matched with the designer's desired state. A high-level example would describe that a goal such as Attack in a friend role would have a high fitness when the agent is very angry with his friend. The state utility defines when these goals are appropriate given the agent's state.

- *Action Utility Function*: Is a function that defines if a particular action is well suited to achieve the goal. This provides a simple yet effective differentiate between one action and another during the planning stages.

This definition of a goal is superior to traditional definitions in that the utility of the selected goal is based on the agent's state and combined with the state of the role. Furthermore, the State Utility Function provides a simple yet effective mechanism in biasing a goal's importance allowing the agent's personality and emotions to play an critical part in the personalization of the role and creating believable decision making.

Having now defined the concept of a goal and its components, beliefs are now described.

3.2.3.2 Beliefs

A role's beliefs play an important role in each stage of the agent's processing. Beliefs define how one interprets events and biases the goal selection and planning processes. Consequently, beliefs play a central role in the assessment of consequence and evaluation of the utility of an action. As described in Section 2.4.2.1, an agent's beliefs may be composed of Values, Norms and a World View. Consequently, beliefs are defined to have:

- *Name*: Defines the name of the belief. A value belief could be BeLoyal
- *Type*: Defines the type as being a value, norm or worldview.
- *Targets*: Defines to whom the value applies. This could be individual or group taken from the role parent or assigned to individual agent.

- *Watch List of Consequences*: Defines a list of consequences of interest to a Belief. For example, a Ninja's *Be Stealthy* would be concerned with consequences to actions such as loudness. This list provides an efficient mechanism to identify what events are important to the agent.

- *State Utility Function*: Defines when utility of the Belief given the internal state of agent. This describes how important the belief is to the agent given his current state. This further allows for the personalization of the role.

- *Appraise Event Function*: Is a function that defines how events related to this belief are appraised. Important events will create emotional memories during the appraisal stage.

- *Action Utility Function*: Defines how useful a particular action is to the agent. This described utility will measure the consequence of the action against the belief's predisposition towards them. For example, a Ninja Role would have a Value such as *Be Stealthy* that would bias the utility of an action based on its loudness. Louder actions would less likely be used. As a result, the action utility function plays an important role in defining if the consequence of an action and it appropriateness during planning.

This belief structure has significant advantages over other definitions belief definitions by allowing partial adherence to each belief based on the agent's internal state. Much like with goals, a utility calculation defines how important a particular belief is to the agent given its state. The rationale for this is clear when using the role of a Monk as an example. A value of 'will not kill' is highly dependent on the monk's personality characteristics but also his emotional state. Game designers should have the ability to define monks that are more likely to behave badly or out of role during moments of great emotion or with agents who are ill suited for a particular role. The importance of the agents belief systems

given the agent's state becomes an important aspect of believability. A second important contribution of this work is the Action utility function defines the utility of an action based on that actions consequences. This provides a simple and effective mechanism to create believable action by biasing the utility of actions based on their consequences.

Having described Beliefs, we will now discuss a Role's Actions.

3.2.3.3 Actions

In order to best define how an agent will behave when achieving a goal, the actions selected should make use of the internal state of the agent. An action defines an event that a agent will perform in the world. These are used within the planning/action selection process including the calculation of consequence. Actions are composed of a number of variables and functions that aid the processes that use them. This list includes:

- *Name*: The name of the action. Examples could be, Walk, Talk, and Strike.
- *Target*: The target of the role defines to whom the action will be directed. The parent role will generally supply this information.
- *Pre-conditions*: A list of conditions that must be true before the action is executed. These conditions can include internal agent states, such as emotional thresholds or relational conditions defined in how the agent relates to the world. The conditions can include both agent and role specific data. For example, Talking to a friend may involve reaching him first. As a result, being close to the target is a precondition to talking with them.
- *Post-conditions*: Is a list of possible consequences to the action if the agent executes it. Included with the consequences should be probability variables or functions defining the likelihood of the event occurring. This information will be used during the planning/Action selection stage.

- *Consequences*: Are a list of relevant information associated with the action. For example, an Action such as talking or yelling would have a loudness consequence value associated with it. This information plays a critical role in evaluating the utility and consequences of the action by the goals and beliefs.

- *State Utility Function*: This function defines the utility of the action based on the agent's current state. This is similar to the state utility function for goals and beliefs. Here the utility function can use role specific emotions and information to assign its utility

- *Personality Effect Function*: When the action executes, this function causes a reduction in the associated desires within the motivational personality traits. This allows for the satisfaction of the desire component of a motivational personality trait.

As with goals and beliefs, the significant contribution of this definition is the integration of role specific information in calculating the utility of the action. This has many advantages as the actions selected become reflective of the agent's internal state in addition to the current state of the role. In consequence, the planning process becomes personalized and reflective of the agent's state. This action definition is novel to the architecture.

Having provided the reader with a detailed definition of the main components of an agent, creating believable actions through the processing of this data will now be discussed.

3.3 Agent's Cognitive Processes

In order to achieve believable action, the agent's cognitive processing uses the information contained in the roles to produce changes within the agent. The processing shares conceptual similarities with the works [26, 27] where appraisal and coping modify the agent's

internal state. Our approach differs in that it is designed to make use of highly reusable components (roles, emotions, personality) defined in an agent as well as its current state.

Section 3.3.1 discusses design of the Relevance Filter, and Section 3.3.2 discusses the Appraisal. This is followed by Section 3.3.3 and Goal selection in Section 3.3.4.1. Finally, Planning is discussed in Section 3.3.4.2.

3.3.1 Relevance Filter

As the number of agents in an emergent system increases so does its complexity. As a result, there exists a potential for significant complexity in discovering what events are important to an agent. In order to decrease this expense, the relevance filter acts as a gate only allowing the appraisal of the event to occur when it is of interest to the agent's roles. Specifically, as each role is added to the agent, the role's goals and beliefs are examined to create an interest *watch list* for the agent. This allows in part for the automatic integration of roles into the agent. This list minimally contains:

- *Target Watch*: The target defines who was involved in the event. For example, if your friend is involved in an event, any goals you have towards him need to be appraised to see if the event occurring furthers or impedes them and should generate an emotional reaction. The target can also be associated with any Beliefs directed towards a particular agent. As a result, any event that involves the target passes to the appraisal process.

- *Action Watch*: A second criteria for appraisal is what action has occurred. Beliefs for example, may be interested in the actions occurring in the world. Police officers want to know when a theft has occurred or if a gun has been fired. As a result, the action is important in defining the emotional reaction to an event and if the event should be passed on for appraisal.

- *Consequence Watch*: When an event is performed, the consequence of the event is also of interest to the appraisal process in generating emotions. A Ninja for example would be interested appraising an event in terms of how loud an action is. The consequence watch produces a list of goals and beliefs that should further evaluate the event.

For each event processed by the agent, the Relevance filter creates a list of triggered Goals and Beliefs that will further evaluate the event. The advantage of this approach is that the event is evaluated once per Goal or Belief.

Relevance Filters have been previously been proposed and discussed in [5]. However, our architecture differs in that its use is directed towards a role-based architecture.

The Appraisal Process and the creation of Emotional Memories will now be discussed.

3.3.2 Appraisal

In line with the works Marsella and Gratch and the theoretical works of Lazerus, the process of appraisal is the performed by assessing an event based an agent's goals and beliefs and creating an emotional memory [26, 27]. Extending this work, our appraisal fully encompasses the beliefs, desires and intentions of an agent based on its emotional and personal state. Furthermore, our appraisal model uses the information and functions contained within the role that in-turn allows for the automatic integration of roles into the agent.

Each event is processed by watches triggered by the relevant filter. As a result, during the appraisal process each triggered Goal or Belief, creates a *Emotional Memory*(EM) by calling the associated AppraiseEvent function. EM's are composed of:

- Event: The associated event. This allows for further processing during the coping phase or use for systems that wish to explain why an agent feels a certain way towards

another agent. This provides a mechanism to describe an interaction history between agents.

- Role: The associated role. This ensures the Emotional Memory is applied to the appropriate role.

- List of Emotions: This provides the emotions and their associated strengths that are integrated into the agent after the coping stage. One can think of these emotions as an instinctive reaction of the agent to the event.

In order to ensure that the emotional memory created is a reflection of the agent's state, the appraised emotional memory is modified by an equation involving the importance of the role and the corresponding state utility function for the goal or belief. This creates a stronger emotional reaction when the goal or belief is part of an important role. As well, the reaction is based on the current state of the agent. As a result, if the agent's state describes the event as not being relevant, then the valences of the emotions within the EM are scaled appropriately.

```
Emotion Value =  Role Importance * State Utility * Appraised Emotion Value
```

It is important to note that this approach to appraisal is novel in that an emotional reaction is created, but it is also modified based on the characteristics of the agent.

Having described the appraisal process, the creation of emotional memories and their scaling based on the agents state, the coping stage will now be discussed.

3.3.3 Coping

As discussed in Section 2.4.1.2.2, one's emotional reaction to an event is highly dependent on how one copes with the event. As a result, an important stage in the processing of an event happens after the initial appraisal as the agent examines how a negative appraisal

can be overturned. Our work follows in line with Marsella and Gratch who use the coping stage to modify the emotional memory of an event based on one of two strategies [26,27]. The first is problem focused coping where the event is analyzed to see if there exists a plan that can be created to overturn the negatively appraised event. If no such plan exists then an emotion focused coping strategy should be used. Once one of the strategies is complete, the emotional memory will be passed to the Active Emotional Memory.

Coping also allows the re-interpretation of events. Events initially appraised while angry may cause guilt in a different emotional state. This is accomplished by re-appraising the events held within the emotional memories and scaling them based on their age.

Having described appraisal and coping, the goal selection and planning stages will now be described.

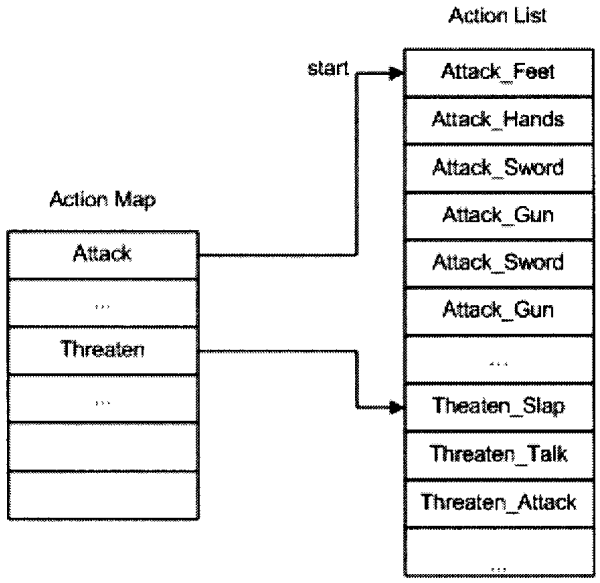
3.3.4 Goal Selection and Planning

With the state of the agent updated, the goal selection and planning processes are performed identifying the next action for the agent. When humans make plan's they are subject to a number of constraints such as their available actions, the consequences to the actions, their emotional state and the urgency of a required decision. These factors should be considered during goal selection and planning.

Traditional approaches have applied utility calculations for goal selection and planning [26,27], and others have applied state utility into planning. However, none have integrated the two. As a result, our architecture is novel in that it makes use of the agent's roles, current state and finally consequence and a care of consequence to create believable action.

In order to facilitate the planning process and automatically integrate new roles, as each role is added to the agent, its actions are added into an agent's action map where the action's consequences map to the actions themselves. This allows for quick retrieval of actions as described in [54]. Figure 3.5 demonstrates this map. This is used in both the

Figure 3.5: Action Map



Goal Selection and Planning processes.

With this structure in mind, goal selection and planning can now be discussed.

3.3.4.1 Goal Selection

After the coping stage, the Goal Selection process selects a goal for the agent. Any goal selected here will have a plan created against it in the planning process. As a result, the goal selection is very important as it has a deep impact on the outcome for the believability of the agent's actions.

The Goal selection process calculates a utility for each goal examining both the state utility function but also the consequences of achieving this goal. An extra parameter to the consequence calculations defines the agent's *Care of Consequence*. This allows designers to model extreme emotional states where one looses the ability to think rationally. When extremely angry for example, one does not consider the negative consequences of one's actions. Rather, the desire to accomplish a certain goal becomes unconditional and

repercussions are ignored. As a result, the equation to define the utility of a goal can be defined as seen below.

```
Goal Utility = Role Importance * State Utility +
               Utility of the Consequence(CareOfConsequence)
```

Here the Utility of the consequence is performed by passing the consequence of the goal to the appraisal process. This in-turn will sum the consequence utilities of the goal against each belief and goal within the agent. The result is a utility value that reflects all of the agent's roles.

This goal selection process is superior to others discussed in Section 2.3 in that the utility of the goal is highly dependent on the current state of the agent. At the time of writing, no known goal selection makes use of this type of information. While Marsella and Gratch do calculate a consequence utility, they do not model how extreme emotional states may affect the reasoning of the agent. By adding the Care of Consequence parameter, our approach is superior in producing believable action.

Once the goal is selected, the planning process is initiated. In the following section, planning is discussed.

3.3.4.2 Planning and Action Selection

In the final stage before an action executes, the planning process takes the current goal and builds a plan against it. This process is similar to GAOP as described (Section 2.3.3.3) in that for each precondition of the goal/or action, an action that solves the precondition is chosen from the action map as seen in Figure 3.5 . This process continues recursively until a plan tree describes a series of actions that the agent should take in order to achieve the goal.

In order to create a believable plan, the search through the tree should be guided by a heuristic. Much like the work of Johns, the utility of each action is described by the

state of the agent [34]. This function is similar to the one seen within goals and beliefs. However, much like the goal selection process, the utility functions defined are novel in that the architecture forces agents to also include the calculation of a care of consequence and the usefulness of the action to solve the goal. The utility function for actions is defined below.

```
Action Utility = Action State Utility +
                 (Care Of Consequence * Utility for Consequence of Action)
```

This use of state utility, care of consequence and the utility of consequence provide the agent with plans that are sensitive to the agent's roles and characteristics. As a result, plans have an increased believability.

Having described the agent processing in detail, the concept of dynamic role instantiation is discussed followed by a summary of the design.

3.4 Dynamic Role Instantiation

An important consideration of this architecture was the ability to plug and play where roles can be added at any time allowing the agent to change its relationship to the environment dynamically. We define this ability as Dynamic Role Instantiation and it is the automatic integration of new roles into an agent at run-time. In this section we define three different types of instantiations, namely *environmental*, *Appraisal based* and *prescribed.*

- *Environmental*: The first type of dynamic role instantiation is based on an association through the environment. In the video game 'The Sims', behaviour of each agent was managed based on the proximity of an object and his or her needs [5]. This approach is efficient as it minimizes the processing performed by the agent as well being very flexible as new behaviours are simply added for each situation/object.

Our approach achieves similar behaviour by dynamically adding roles and removing roles when appropriate. For example, as an agent needs to accomplish a goal such as shopping, when the agent reaches the shopping mall, the role of 'shopper' should be added to the agent. By adding the role, the agent now has all the information (through its beliefs) required to not only guide its behaviour in lineups, but also make judgments on the behaviours of others. However, our architecture does much more in creating this believable behaviour. As the importance of goals, and beliefs are dictated by personality model and emotional state, the agent's behaviour becomes a reflection of these traits.

- *Appraisal-Based*: Another type of dynamic role instantiation is performed through associated roles. Here one role may cause the run-time addition of another role to the agent. As a police officer observes the event of an agent stealing from another, a new role could be added to the police officer dictating that the agent is a thief. This further describes the relationship of the officer to its environment and informs the officer on how to proceed. This can be seen as a simple form of learning as the agent adds new information based on a specific appraisal of events in the environment.

- *Prescribed*: Another type of dynamic role instantiation is when one agent assigns a role onto another agent. Prescribed instantiations could take the form of a general assigning a role to a soldier. These assignments could also play an important aspect of the interaction between an AI Drama Manager and agents within the game world. Creating the role of a protagonist could allow an agent special abilities to move the story in a particular direction. This should provide useful for the creation of emergent narratives.

As our architecture is built to facilitate this ability to dynamically assign roles to an agent, it demonstrates a high usability within video games but also aids in the believability

of the agents.

3.5 Summary of Design

In summary, the design presented within this work is focused on creating believable behaviour for agents while allowing a wide breadth of features to be easy designed and re-used. The design focuses on the creation of believable decision making by clearly defining the agent's relationship to the environment through its roles, current emotional state and personality traits. This definition combined with Appraisal Theory and Utility-Based decision making into a single data-driven Agent allows for decision making that could be perceived as believable. This is done through the integration of Appraisal Theory specific cognitive processes into roles that allowed for utility-based decision making. Here, roles and their components Beliefs, Desires, Intentions, Emotions and Personality Traits play important roles in each stage of cognitive processes.

This work also extends the state-of-the-art by personalizing roles through the inclusion of emotions and personality traits(basic and motivational) into the roles. With the addition of basic and role specific emotions and an Active Emotional Memory, the architecture provides the ability to create agents with a wide variety of behaviours given the current state of the agent. This also provides an important feature in that roles can be re-used by as each agent while still providing unique behaviour. Here, each role is subjective to the agent's personal and emotional state.

Finally, this work allows for the automatic integration of roles by tightly integrating roles with the cognitive processes of the agent. As a result, dynamic role instantiation is possible at run-time. This allows for efficiencies such as automatic role addition and removal given the environmental object. Furthermore, this allows for automated agent learning through role additions given specific events. This automatic role integration is

also important to the ability for agent groups to be created dynamically, and the AI drama managers to enlist certain characters with specific roles.

It is important to note that the design of this architecture is vastly different from the work of [87]. The main similarities are the inclusion of roles and the ability to abstract both personality, and emotional models to make decisions. The main difference is the cognitive abilities of this agents built using this architecture are highly enhanced via the integration of roles, appraisal theory. Notably, this allows for not just believable reactions, but also acting taking into account consequences to actions.

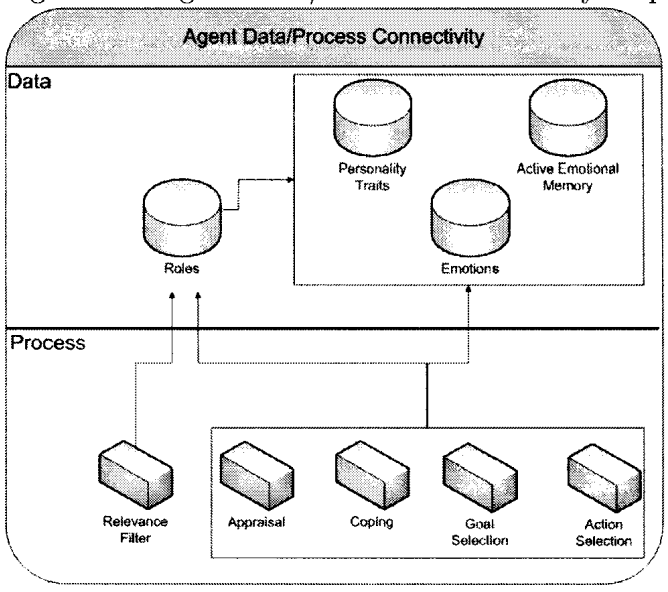
In order to establish the validity of this architecture, and implementation will now be discussed.

Chapter 4

Implementation

To demonstrate the validity of our approach and this architecture, an proof-of-concept implementation was built in C++ using Visual Studio 2008 in Windows Vista. An output log was created at run-time containing the actions of the agents, as well as their processing and current state during each stage. The general architecture was created composed of each aspect described in the design. This included, emotions, personality traits, an Active Emotional Memory(AEM), as well as roles and their associated goal, beliefs, actions and role specific emotions. Furthermore, the processing stages of Relevance Filter, Appraisal, Goal Selection and Planning/Action Selection were fully implemented. These elements, representing the key features of this work, should produce believable decision making and consequently should be sufficient to validate the design. A basic coping stage was implemented as a method to integrate emotional memories into the AEM, but more complex coping strategies, as discussed by Marsella and Gratch [27] were left as future work. A world layer was built to create events from the actions selected by agents. However, an AI drama manager was not built as it is optional to the architecture and is a significant undertaking on its own, outside the scope of this thesis. Dynamic role instantiations was implemented to demonstrated the run-time ability to automatically integrate new roles.

Figure 4.1: Agent Data/Process Connectivity Map

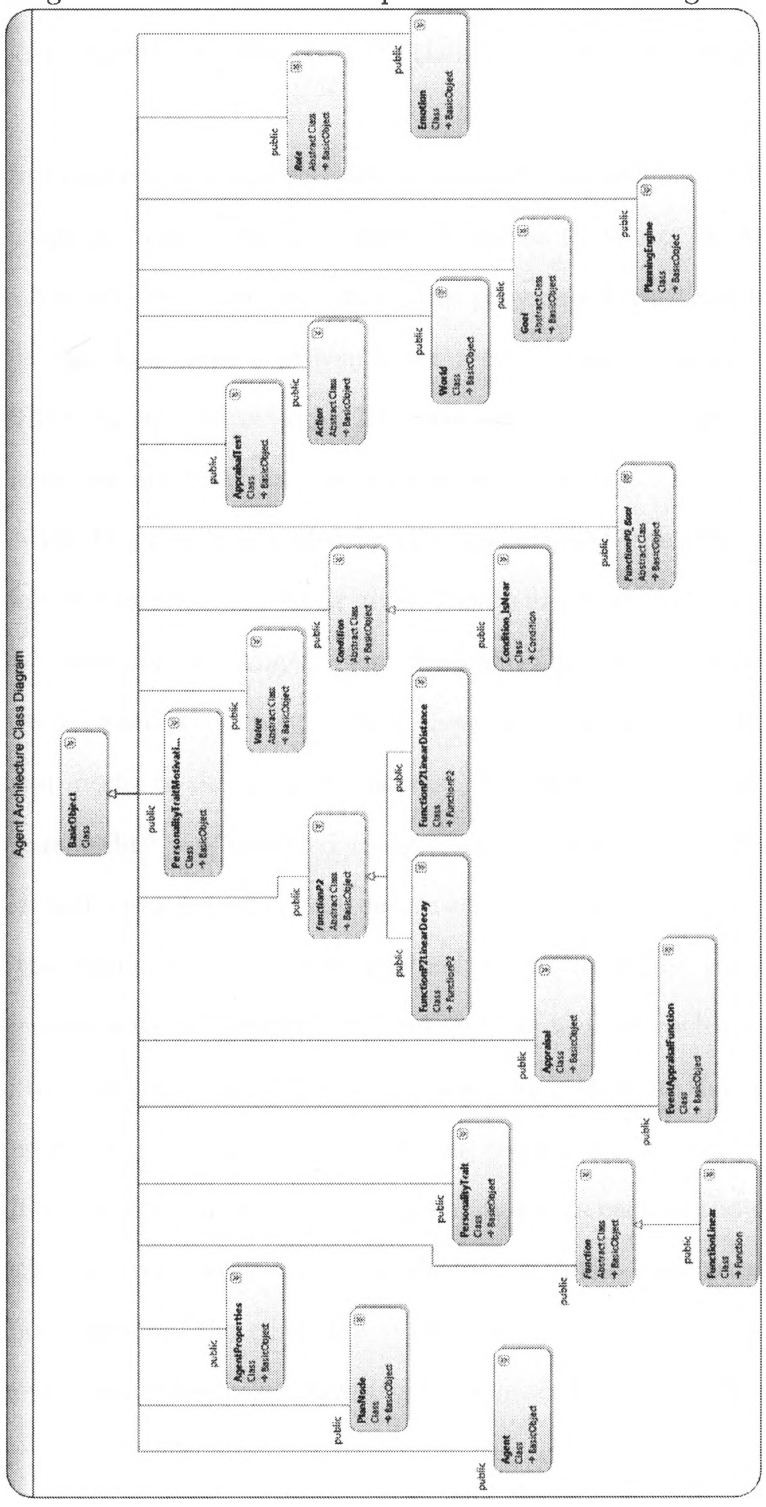


The world layer was responsible in alerting the agents of the presence of new events. As discussed, the data-driven paradigm was maintained for the roles. This resulted in new behaviour of agents simply by adding new roles. Furthermore, each role was personalized by the agent's personality traits and emotional state. Figure 4.1 illustrates the connectivity among processes and data within an agent.

In total, the agent architecture is composed of 25 classes including abstract classes and full functionality for each process. Figure 4.2 presents a class diagram of this architecture. It is important to note that these classes only represent the logic of the agent's processing and features many abstract classes. To create a functioning program, classes need to be implemented as discussed in Section 4.1, Section 4.2 and Section 5.

The following sections describe the implementation of the architecture. Section 4.1 describes the implementation of the agent data while Section 4.2 describes the implementation of the agent's cognitive processes.

Figure 4.2: Architecture Implementation Class Diagram



4.1 Agent Data

As described above, agents are described by their Emotions, Personality and the roles currently active.

For this proof-of-concept, the agent's basic emotional state was modeled using Eckman's Six Universal Emotions. These emotions were chosen as they provide are both compact in representation but are also numerous enough to provide a breadth of human emotions. Finally, Eckman's emotions have a proven track record being in many agent implementations. As described earlier in Section 3.2.1, each emotion was composed of a value, an activation and de-activation function and a range between zero and one. At each iteration, the deactivation function was scaled by the time elapsed from the last update. This assumption models the fact the emotions given time will return to a baseline being zero. The Six Universal emotions are Anger, Fear, Sadness, Disgust, Surprise and Happiness. As each emotion is accessed frequently, each of these emotions was directly embedded into the basic emotional model of the agent to increase the efficiency of the processing.

The agent's personality was modeled using Reiss' sixteen basic trait desires as listed in Section 2.4.1.1 and modeled after the design described at Section 3.2.2. During the initialization of the agent, the trait was assigned a current value, set point and desire and that would output a value between zero and one. As well, at each iteration, the decay function reduced the current value of the trait. This models the idea that a *desire increases* over time. As described in the action definition at Section 3.2.3.3, only their completion may reduce the desire within the trait. As with emotions, as each of the personality traits are used frequently, the traits were directly embedded into the basic personality model of the agent in order to increase the efficiency of processing.

With each agent, a list of the roles known as the *role set* held each role currently applied to the agent.

Figure 4.3: Role Definitions

Role	Target	Goals	Values	Interest emotions and traits	Actions
Ninja	self	none	BeStealthy, BeDeadly	anger, fear	Attack_Hand, Attack_Sword, Defend_Yell, Defend_Separate,
Enemy	group	Attack, Defend, Threaten	BeDeadly	anger, fear, vengeance, honour	Threaten_Yell
FamilyMember	group	Socialize	BeLoyal	anger, fear, vengeance, honour, social contact	Talk
Human	self	none	BeVengeful		Idle

A further discussion on the implementation of the role follows in the next section.

4.1.1 Role Implementation

The general structure of the roles was implemented as the description discussed at Section 3.2.3. In this prototype implementation, only the concept of a value was implemented within the beliefs of a roles; the addition of norms and worldview are left for future work.

A number of roles were created that would be used during the evaluation. These roles were Family Member, Enemy, Ninja, and Human role and were applied to agents given the scenario. During the creation of each of these roles, predefined goals, values, actions and role specific emotions are assigned to the roles, as the roles are simply a method of organizing various information describing the relationship and behaviour of the agent to the environment. Each of the components is inserted into a map based on their pointer value and mapping to the component itself. Figure 4.3 shows a chart of the roles, and their various components and their associated state utility information.

It is important to note that the Family Member role and Human role both had values that performed dynamic role instantiation as described in Section 3.4. Here, when the role's target was attacked, threatened, or killed, a new Enemy role would be dynamically assigned to the attacker and added to the role set.

4.1.2 Implementing Utility

4.1.2.1 State Utility Calculation

As discussed through Section 2.4.1.2, a state utility function is an important aspect of deciding how important a Goal, Belief or value is to the agent. Using the works Johns [34] as a reference for state utility. This function defined the usefulness of a current state based as a sum of the Euclidean distance between a number of parameters being the emotional state and the desire of a personality trait. During the initialization of the Goals, Value and Actions, a number of parameters are assigned a value outlining when they would be most useful. Let us take the definition of action Attack within the Lover Role as an example.

```
Action Attack
Emotion["Jealousy"] = 0.8f;       // role specific emotion value
Traits["Honor"] = 0.5f;             // personality trait desire value
Emotion["Anger"]  = 0.8f;         // basic emotion value
```

Here the utility function should return a high value when the agent's internal state closely matches the values described above. As a result, for each parameter, the utility function would measure one minus the absolute distance between the parameter and its associated characteristic within the agent. For any role specific emotion such as jealousy the utility function becomes

```
Utility Role Specific Emotion = 1 - ABS(Role.Emotion["Jealousy"] -
                                        Action.Emotion["Jealousy"])
```

The result of this calculation is that as the distance between the jealousy value in the role and predetermined jealousy value in the agent increases, its utility decreases. Basic traits will have a similar equation. Basic emotions however have a slightly different evaluation.

```
Utility of Basic Emotion Anger = 1- ABS( (Agent.basic emotion["Anger"] +
                                    Role.emotions["Anger")/2.0f - Action.Emotion["Anger");
```

By averaging the parameter value from 'mood' of the agent and the role specific emotion, a middle ground is reached. Taking anger as an example, if the agent's mood is very angry but its relationship to a friend involves no anger, the overall anger is diminished. This allows the utility to reflect its good relationship with its friend despite its anger.

The final utility of the state is defined by as the normalized average between zero and one of each of the parameters.

4.1.2.2 Care Of Consequence

The implementation of an agent's Care of Consequence, is the simple calculation taking into consideration the emotional value of anger. This was chosen as one's care of consequence is intuitively reduced while angry allowing the testing to demonstrate changes to processing.

```
Care Of Consequence = 1 - Emotion.anger
```

This equation defines that the agent has less care about the consequence of its actions as its anger increases. This is an effective, yet believable method of defining one's care to the consequence of actions. However, future work should include a more accurate care of consequence calculation that takes into account all of the various and emotions and personality traits. For example, when one is in a state of ecstatic joy, they may not properly evaluate the consequences of their actions. Similarly, certain individuals have a better quotient for avoiding rash decisions, this personality trait could become an important aspect of the care of consequence calculation.

4.1.2.3 Consequence Utility Calculation For An Action

The consequence utility calculation plays an important aspect in the goal selection and planning stage. As discussed in Section 3.3.4.2, the total consequence utility for an action or goal is defined by the summation of each consequence utility calculation for each goal and belief of the agent. A pseudo code of the implementation is described below.

```
float utility  = 0.0f;
For Each Triggered Goals or Beliefs
{
       float actionUtility =  Triggered.CalculatedConsequence(anAction)
       if(actionUtility > -Care of Consequence)
       {
            utility += actionUtility;
        }
}
return utility;
```

Here, the care of consequence only allows negative appraisals to be added to the utility for the action if they are above its value.

It is important to define how Goals and Beliefs may calculate the Consequence Utility for an action. As an example of how an individual function is implemented, the BeStealthy Value within the Ninja role will be described.

Intuitively, being stealthy involves attempting to stay as quiet as possible, and so, the BeStealthy Value is concerned with actions that contain the "loudness" consequence. Here, if an action's loudness is above a certain threshold, then the action returns a negative appraisal. This signals to planning processes that this action is not favourable for this value. However, if the action's loudness is beneath a certain threshold, the utility becomes positive. Implementing this type of utility function is trivial as see below.

```
CalculateConsesquenceUtility(Action &action)  // BeSneaky Value
{
    float loudness = action.consequences["loudness"];
    return this.threshold - loudness;
}
```

In the following section, the implementation of each of the agent's cognitive processes is outlined.

4.2 Process information

The Agent's cognitive processing was implemented based on the design outlined in Section 3.3.

The relevance filter was implemented as three maps namely, target to beliefs, action to beliefs and consequence to beliefs. When a role is added, the belief is added into the map watching for the associated item of interest. When an event occurs, the filter examines the target, the action and the consequences and then builds a list of triggered beliefs.

The appraisal stage is implemented in combination with the beliefs. Here, each triggered belief is passed the event to appraise and returns an emotional memory that is scaled as discussed in Section 3.3.2. The item is then added to a list of new emotional memories.

The coping stage takes the list of emotional memories and updates the active emotional memory. Future research will attempt to overturn negatively appraised events as seen in the works of Marsella and Gratch [26, 27]. Specific for this architecture, the re-interpretation could occur changing the importance of a role's beliefs or desires.

As the selection of a goal must reflect the consequence of how it will be achieved, the selection of a goal needs to reflect what actions are available to the agent. However, the aggregation of actions are dependent on each role of the agent. To accomplish goal

selection under these settings, this implementation of the goal selection process performs a one action deep search of actions fulfilling its requirements and selects the best one based on its state utility, care of consequence and utility of its consequences. Future research will extend this work by performing an analysis of a goal and creating a heuristic value based on an average of the consequences. Such an approach would likely take place when a role is added to an agent.

The planning process is also implemented as described in Section 3.3.4.2. Currently, when a goal is selected the planning process first builds a plan tree composed of the available actions and then performs a greedy search from each leaf of the tree moving upwards comparing each node at each level against each other and selecting the node with the highest utility.

While performance is critical to this type of architecture, our current work focused on the delivery of functionality and establishing the validity of our approach and architecture. The work in [64] is focused on performance optimization, scheduling and scalability of precisely this sort of agent system, and is even applied to our prototype in that work. Consequently, we did not focus on performance specifically in this thesis.

Having described the implementation of the architecture, an evaluation of its ability to create believable action will now be discussed.

Chapter 5

Evaluation

The goal of the architecture design is to create decisions in Non Player Characters in video games by providing the designer with a number re-usable components that lead to believable behaviour. Importantly, these decisions must reflect the roles of the agents, their current emotional state, desires of the personality traits, consequences of actions, as well as the ability to change their behaviour and mental processing due to external events unfolding in the environment. A second goal of the architecture was to create a role-based architecture where roles could be re-used and dynamically added to agents to modify their behaviour. Finally, implicit to the goals of the architecture is that believable decision making would unfold unscripted through the interaction of agents within the world. This emergence of a narrative, has been noted as an important aspect of video games [2,3,5,77].

Unfortunately, establishing the believability of agents and their actions needs to done with user testing which is outside of the scope of this thesis and is left for future work. As a result, the evaluation of this thesis is focused on the identification of properties of the agents' processing that could be construed as believable as well as evaluating the re usability of the architecture .

In order to evaluate if these goals are achieved by the architecture, multi-agent *dilemma-*

based scenes were created and the processing of agents analyzed. Dilemma-based scenarios are chosen for two reasons. First, they are a staple of traditional storytelling and provide a backdrop to which the quality of interaction among agents can be examined. Furthermore, dilemma-based scenes involve difficult but believable decision making where the audience can relate to the thought process and actions of the characters [7]. This applies particularly well to our architecture as difficult dilemmas can be easily translated to a state of *role overload* where an agent's goals are mutually exclusive. Consequently, the architecture can be seen as successful if the agent's interaction and choices could be construed as believable.

In order to analyze the multi-agent dilemma-based scenarios, each scene was instantiated and the agent's actions written to a Log file containing information on the Agent's roles, emotional and personality state, as well as process information from the Relevance Filter, Appraisal, Coping, Goal Selection, Planning/Action Selection and the creation of events. The resulting Log files are an effective mechanism in validating the architecture as they allow for the analysis of the scenes' events; allowing for the believability of the agent's actions to be assessed. Furthermore, the Log files are advantageous to observing in-game characters as they expose the agents' internal processing and state allowing for a detailed examination without the time consuming production of artistic game assets. The examination of the processing and internal state is also desirable as it allows for the attribution of causality into the effects of each process when considering the state of the agent. These logs files provide the information required to fully understand the architecture and validate its processing. To evaluate the scene, a significant amount of programming went into implementing classes discussed in Section 4. In total, four roles, four goals and four values were created and a total of six separate actions were implemented. All of Reisses' sixteen personality traits and Eckman's six emotions were implemented and integrated into the roles. Figure 5.1 outlines the roles and their associated values, goals and actions. Also

included in this figure is the initial layout of each scene including the character and their associated roles.

In order to assess the re-usability of roles within the architecture, many of the same roles were applied in each scene unchanged. As a result, the major differences between each scene are the number of roles applied to the agent and the emotional and personality characteristics of the agents. Also, dynamic role instantiation was tested through an appraisal-based addition of an enemy role as a result of specific appraisals within the Family Member Role.

In this thesis, three dilemma-based scenes are presented and discussed to demonstrate the functioning of the agent architecture. During each iteration, agents performed the complete processing algorithm discussed in Section 4 and details were written to a Log file. Specially, the Agent's roles, emotional and personality state, as well as process information from the Relevance Filter, Appraisal, Coping, Goal Selection, Planning/Action Selection and the creation of events were stored. The resulting Log files are an effective mechanism in observing and understanding the processing of the agents as well as providing a powerful argument in validating the architecture.

Each scene was run for 100 iterations during which each agent within the scene would perform actions. One hundred iterations was chosen as it provided sufficient data about the agent's processing to come to an understanding about it features.

In the following section, Section 5.1 the implementation of three dilemma-based scenes is described and the behaviour of the agents analyzed. Section 5.2 describes a summary of the findings.

Figure 5.1: Scene Roles and Initial Scene Configuration

Role	Values	Goals	Action
Enemy	None	Defend	Defend Separate Defend Yell
		Attack	Attack Hand Attack Sword
		Threaten	Treaten Yell
Family Member	Be Friendly Be Loyal	Socialize	Talk
Human	Be Vengeful		
Ninja	Be Stealthy Be Deadly	None	None

Scene 1: Unknown Assasination: Part 1	
Character	*Role - Target List at beginning of scene*
Ninja	Human - Self Enemy - Pirate Ninja - Self
Pirate	Human - Self

Scene 2: Unknown Assasination: Part 2	
Character	*Role - Target List at beginning of scene*
Ninja	Human - Self Enemy - Pirate Ninja - Self Family Member - Wife
Wife	Human - Self Family Member - Ninja

Scene 3: And Fire-eyed Fury Be My Conduct Now	
Character	*Role - Target List at beginning of scene*
Mercutio	Human - Self Enemy - Tybalt Family Member - Romeo
Tybalt	Human - Self Enemy - Mercutio Enemy - Romeo
Romeo	Human - Self Family Member - Mercutio Family Member - Tybalt

5.1 Scene Evaluation

5.1.1 The Unknown Assassination: Part 1

The first scene was created to evaluate the decision making and behaviour of a Ninja who is sent on an assassination. Having been instructed to assassinate an unknown person at a specific location, the Ninja opens the door to find that its target is his sworn enemy, the infamous Pirate. While the outcome of this dilemma is not particularly difficult to imagine, it is important in aiding the reader understand the decision making that unfolds.

The scene was created using two agents being Ninja and Pirate. Figure 5.1 shows the initial configuration of the both agents. An excerpt of the Log file's Events can be seen below.

```
1) Event: Agent: Ninja Action:   Attack_Hand Target: Pirate Utility: 0.74775
2) Event: Agent: Pirate Action: Threaten_Yell Target: Ninja Utility: 0.846375
3) Event: Agent: Ninja Action:  Attack_Sword Target: Pirate Utility: 0.756563
4) Event: Agent: Pirate Action: Attack_Sword Target: Ninja  Utility: 0.713281
                                    ...
      ******************** Pirate Has Died ************************
```

Examining this output demonstrates a few interesting actions. One might wonder why the Ninja's first action was to attack with its hand instead of the sword. Examining the planning/Action Selection reveal that the utility of the Ninja's State was better suited toward this action. Furthermore, the Ninja's Value of BeStealthy and BeDeadly helped define the utility of the hand attack as being more desirable then the sword's louder yet more deadly attack. To an observer, it could be expected that Ninjas would prefer a silent assassination instead of a loud battle, the action selected reflects the Ninja's role and could be seen as believable.

The Log's excerpt of this evaluation is seen below.

```
Action Selection: Attack_Hand Targets: Pirate  Utility 0.9325
                                        State: 0.7525
                                Consequence: 0.18
Action Selection: Attack_Sword Targets: Pirate  Utility 0.6875
                                        State: 0.6275
                                Consequence: 0.06
```

In response to being struck, The Pirate becomes angered and its Human role BeVengeful value instantiates an Enemy role towards the Ninja. The appraisal of this event can be seen below.

```
Enter Appraisal: Pirate
      RoleName:RoleHuman    typeclass BAA::Role_Human
   Appraising Event: Trigger:Ninja Action: Attack_Hand Targets:
     Target Watch Triggered: Pirate
       Triggered Values: BeVengeful
    Adding Role: Enemy_Ninja
                          Adding Goal: Threaten
                          Adding Goal: attack
                          Adding Goal: defend
              - Emotional Memory Integration
                      For Role:Enemy_Ninja
                      Emotion:anger Value: 0.2
Leaving Appraisal: Pirate
```

Having added the enemy role, the Pirate state results in a threat sent at the Ninja. The Ninja in Event 2, angered by the Pirates threat, slightly loses its care of consequence. This in-turn decreases the importance of the Value BeStealthy. As a result, the consequence for an all out sword attack is found to be the most useful.

```
* Care of Consequence: 0.738
      Action Selection: Attack_Hand Targets: Pirate  Utility 0.9905
                                    State: 0.8105
                              Consequence: 0.18
      Action Selection: Attack_Sword Targets: Pirate  Utility 1.2255
                                    State: 0.6855
                              Consequence: 0.54
- Event: Agent: Ninja Action: Attack_Sword Target: Pirate  Utility: 0.6855
```

The Pirate extremely angered by the attack from the sword is no longer content with simply yelling at the Ninja. Rather the sword attack becomes the most useful action. The battle ensues until the Pirate is killed by the Ninja.

This scene demonstrates many important aspects the architecture as we can see the state of the agent, the appraisal of events and the consequence being important aspects in deciding which action is taken. The result of the scene could be construed as believable in that the decision making utilities of the both agents reflects the roles and their current situation.

5.1.2 The Unknown Assassination: Part 2

In following previous scene, we demonstrated how a Ninja would assassinate a sworn enemy. To create a scene with a greater dilemma and demonstrate how decision making is changed when conflicting roles are added, we switch the unknown assassination target from the Pirate to the Ninja's Wife.

The scene was created using two agents being Ninja and Wife. Figure 5.1 shows the initial configuration of the both agents. An excerpt of the beginning of the Log file's Event can be seen below.

```
1)  Event: Agent: Ninja  Action: talk Target:  Wife  Utility: 0.89375
2)  Event: Agent: Pirate Action: talk Target:  Wife  Utility: 0.89375
3)  Event: Agent: Ninja  Action: talk Target:  Wife  Utility: 0.706875
                                    ...
```

Not surprisingly, when the Ninja enters the room and discovers its Wife, the couple talks. The reason is that the role importance values for the Enemy and FamilyMember role affect the utility of actions. Another consideration is the Value BeLoyal directed at the wife. This value would reduce the utility of an action inflicting harm on to the wife. This processing and hence the actions taken are both expected and believable. However, what happens near the end of the scene is surprising as the Ninja strikes its wife and then speaks with her on the next iteration.

```
96) Event: Agent: Ninja Action: Attack_Hand Target: Wife  Utility: 0.77
98) Event: Agent: Ninja Action: talk Target: Wife  Utility: 0.7
                                    ...
```

These actions are a little incredulous and tuning the parameter of importance of the Ninja's Enemy Role would have be averted this situation. However, it demonstrates the desires of the personality traits at work. Examining the Log files we see that the desire for Power and Vengeance are silently increasing throughout the scene while the social contact desire decreases. As Vengeance and Power are associated with the attack goals, the utility of this goal increased throughout the scene. The respective trait values can be seen below.

```
1) Trait: power           SetPoint: 0.5 Value: 0.49 Desire: 0.00999999
   Trait: socialcontact SetPoint: 0.8 Value: 0.49 Desire: 0.19375
   Trait: vengeance    SetPoint: 0.5 Value: 0.49 Desire: 0.00999999


96) Trait: power           SetPoint: 0.5 Value: 0.394001 Desire: 0.105999
    Trait: socialcontact SetPoint: 0.8 Value: 0.999      Desire: 0
```

```
     Trait: vengeance     SetPoint: 0.5 Value: 0.394001 Desire: 0.105999

98) Trait: power            SetPoint: 0.5 Value: 0.394001 Desire: 0.105999
      Trait: socialcontact  SetPoint: 0.8 Value: 0.998      Desire: 0
      Trait: vengeance     SetPoint: 0.5 Value: 0.403001 Desire: 0.0969988
```

Having sated the need for vengeance by attacking another (Note: Vengeance reduced), the Ninja then returns to talking with its Wife.

This result being un-expected was however, welcome as clearly demonstrates that the personality traits play an important yet subtle role in the decision making of the agents. Furthermore, the scene demonstrates how the importance of conflicting roles within the agent affects its decision making. Finally, the Ninja's decisions could be construed as believable as the Ninja was supposed to assassinate his wife and the utility of the choices represents a conflicted state.

5.1.3 And Fire-eyed Fury Be My Conduct Now

Shakespeare's Romeo and Juliet is tragic tale of passionate lovers and is often regarded as one of the most famous stories in the English language. One of the most pivotal moments in the play is in Act 3 Scene 1 as the outcome of this scene ultimately leads to Romeo's exile and the suicide of the lovers. In the scene, Romeo and Mercutio are socializing when the sworn enemy Capulet cousin Tybalt enters. Mercutio and Tybalt fight, and Romeo is faced with a dilemma. Should he fight the sworn enemy of his family or save his lover's cousin. In an attempt to save both lives, Romeo separates the two by putting himself between them. It is then that Tybalt slays Mercutio. Seeing the death of his best friend, Romeo is overcome with rage, and having lost all care of consequence infamously says"And fire-eye fury be my conduct now" and slays Tybalt setting in motion the tragic events in latter parts of the play.

Modeled this scene through the following role configuration as seen in Figure 5.1 .

Before discussing the results, let us first discuss the initial role assignment. Tybalt was quite easy to model as he would be Enemies anyone associated with the Montague (Mercutio and Romeo). Mercutio being a good friend of Romeo, treats him as his brother hence the Family Member role is applied to Romeo. Mercutio also perceives Tybalt to be his enemy. Romeo's role assignment in part reflects his love for Juliet and her values. He sees himself as a brother to Tybalt wishing no ill on anything Capulet. Furthermore as Romeo sees Mercutio as his best friend, it is fair to generalize this relationship to the family member role. From this setup, the dilemma of Act 3 Scene 1 unfolds.

The beginning of the scene can be viewed below in an excerpt of the Log file's events.

```
1)  Event: Agent: Mercutio Action: talk Target: Romeo     Utility: 0.89375
2)  Event: Agent: Romeo    Action: talk Target: Mercutio  Utility: 0.7
                              ** Enter Tybalt **
3) Event: Agent: Tybalt Action: Threaten_Yell Target: Mercutio  Utility: 0.85125
```

As expected, Romeo and Mercutio converse until Tybalt enters the scene and threatens Mercutio. During the appraisal here, Romeo adds the Enemy role towards Tybalt in response to Tybalt threat.

```
4) Event: Agent: Mercutio Action: Threaten_Yell Target: Tybalt  Utility: 0.849304
5) Event: Agent: Romeo    Action: talk           Target: Mercutio  Utility: 0.6915
6) Event: Agent: Tybalt    Action: Threaten_Yell Target: Mercutio  Utility: 0.850875
```

In the second frame, Mercutio angered by Tybalt's threat returns the favor that in-turn causes Romeo to apply the Enemy role onto Mercutio. Nonetheless, Romeo continues to talk with Mercutio and Tybalt continues his verbal assault on Mercutio.

```
7) Event: Agent: Mercutio Action: Attack_Sword    Target: Tybalt  Utility: 0.731439
8) Event: Agent: Romeo    Action: defend_seperate Target: Tybalt  Utility: 0.64784
```

```
9) Event: Agent: Tybalt    Action: Attack_Sword Target: Mercutio  Utility: 0.729146
      ******************** Mercutio Has Died **********************
```

Looking at event seven, we see that Mercutio has loosed his sword in response to Tybalt's threats. We now have a pivotal moment for Romeo whose internal struggle is apparent with his planning/Action Selection Process. Even through both Mercutio and Tybalt are enemies, attacking either of them is not an option. As we see below for the utility calculations for a strike against Tybalt, the consequence utility is zero. This is due to the action having a positive value for the enemy role but a negative value for the friend role.

```
Action Selection: Attack_Hand Targets: Tybalt  Utility 0.64784
                                                    State: 0.64784
                                              Consequence: 0
```

As a result, Romeo's best action is to attempt to separate the two combatants. Tybalt responding to Mercutio's threat also attacks and kills Mercutio. The result here is extreme anger in Romeo. Examining his state after the appraisal of Tybalt's killing blow is rage. Below shows the anger emotion in the AEM both before and after Mercutio's death.

```
Emotion: anger Value: 0.0387244  ( Before Event 9)
Emotion: anger Value: 1              ( After Event 9 )
```

Being completely enraged, Romeo's planning/Action selection reflects this state. We see his care of consequence dropping to zero. As a result, the Attack Sword action's utility becomes positive by ignoring any negatively appraised consequences.

```
* Care of Consequence: 0
    Target Watch Triggered: Tybalt
     Action Selection: Attack_Hand Targets: Tybalt  Utility 0.74575
                                        State: 0.62575
```

```
                    Consequence: 0.12
    Target Watch Triggered: Tybalt
    Action Selection: Attack_Sword Targets: Tybalt  Utility 1.02075
                                State: 0.75075
                    Consequence: 0.27

10) Event: Agent: Romeo Action: Attack_Sword Target: Tybalt  Utility: 0.75075
11) Event: Agent: Tybalt Action: Attack_Sword Target: Romeo  Utility: 0.689941
        ********** Tybalt Has Died **********
```

Therefore, Romeo in "Fire-eyed fury" slays Tybalt. Thus ending the dispute and the scene. This scene could be construed as believable through the processing of Romeo. Not only did we see restraint in his initial actions, action utilities, care of consequence calculations and appraisal of events, but we also saw the raw power that emotion can have on the processes of planning/Action Selection. This resulted in the believable, unscripted re-enactment of one of Act 3 Scene 1 of Shakespeare's Romeo and Juliet.

5.2 Summary and Discussion of Scene Findings

The results from scenes provide the first steps in validating this architecture. The application of roles to agents played a fundamental aspect in shaping their behaviour. Scene One and Two contrasted this. By simply adding a role, the behaviour of the agents changed dramatically while still maintaining believability.

Emotion and Personality traits also played an important role in defining the current state of the agents and in-turn their behaviour. Goals and Actions were selected due to emotions and personality modifying their utility. The Care of Consequence also played a critical role in defining the utility of to the consequence of actions. This was clearly

shown in Scene Three where Romeo's emotions drove his actions to be believably reckless, ignoring the negatively appraisal of actions.

Each of the processes also showed their an ability to modify the agents behaviour. The Relevance Filter correctly triggered the appropriate goals and beliefs. The appraisal process formed emotional memories and instantiated new roles based on the events appraised. This was evidenced in each scene where Anger resulted from being attacked or threatened, and an agent would instantiate a new Enemy Role. This played an important role modifying the agent's state that in-turn would affect the agent's decision making. Coping integrated the emotional memories into the active emotional memory. The Goal Selection and Planning/Action Selection processes appropriately selected actions based on current state while taking into consideration the consequences. Clear evidence was seen in the planning process of Romeo who decided to defend his friend instead of attacking. Each scene demonstrated that the agent's roles set combined with its current state and processing produced behaviour that could be construed as believable.

While the scenes were a success, it was found that the parameters for the utility calculations needed to be tuned to produce realistic behaviour. As a result, designers wishing to use this architecture need to spend some time configuring the system. Once working parameters were found, then applying them into multiple scenes worked as expected. This was evidenced in that each scene used the same roles, with the only changes being the roles applied on the agent and their initial emotional/personality values. Future research should be directed at automating the parameter tuning within the utility calculations. Here expected behaviour could be defined and parameters could be automatically tuned to produce the desired results.

Chapter 6

Conclusion and Future Work

In conclusion, the design presented within this work focused on creating believable behaviour for interactive agents and Non Player Characters in a highly reusable, flexible architecture. While a formal assessment of believability through empircal evaluation is still required, this thesis provides the foundations for such work by integrating the core elements necessary for believability as defined by Loyall. This was achieved through the integration of Role Theory, Appraisal Theory and Utility-Based decision making into a single data-driven agent while at the same time allowing for a flexibility and reusability. This integration allowed for both processing that could be construed as believable all the while automatically integrating roles into an agent's decision making. This work personalizes roles through the integration of emotions and personality traits into their definition. Basic emotions, role specific emotions and the Active Emotional memory played an important role in modifying the goals and actions selected based on the current emotional state. Furthermore, the inclusion of basic and motivational personality models allowed for the novel behaviour to emerge through their interaction with utility-based decisions. This personalization of roles allowed for both controlled and uncontrolled behaviour of agents providing a realistic emulation of classical human behaviour and story telling in an emergent system.

This architecture provides many avenues for future work. First, and most important is undoubtedly its use in bigger scenes with more characters simulating a real world environment. Another important direction of future work is the integration into a video game with agents capable demonstrating the actions and emotional state in real time. The objective of this research would further demonstrate the validity of this architecture and believable behaviour in scenes with a high number of agents. Also, future research should include performance analysis and optimization of the architecture that is now currently in progress [64]. Other avenues for research would be the inclusion of an AI Drama director into a scene while having it dynamically add and monitor events in order to attempt to create an emergent story. Given the architecture, it would also be possible to make use of roles to create world generation. Here an AI drama manager could populate cities, creating using roles to define the various roles within the society as well as the familial structures. Roles would seem well suited to this trough their high re-usability, personalization and defined behaviour.

Bibliography

[1] E. Andre, M. Klesen, P. Gebhard, S. Allen, and T. Rist. Integrating models of personality and emotions into lifelike characters. *Lecture Notes in Computer Science*, 1814, 2000.

[2] R. Aylett. Narrative In Virtual Environments-Towards Emergent Narrative. In *Working notes of the Narrative Intelligence Symposium*, 1999.

[3] R. Aylett, J. Dias, and A. Paiva. An Affectively-Driven Planner For Synthetic Characters. *Proceedings of ICAPS 2006*, 2006.

[4] C. Bailey and M. Katchabaw. An Experimental Testbed To Enable Auto-Dynamic Difficulty In Modern Video Games. In *In Proceedings of the 2005 GameOn North America Conference*, 2005.

[5] Christine D. Bailey. Towards emergent gameplay: A framework for realistic psychological behaviour in non player characters. Master's thesis, The University of Western Ontario, 2007.

[6] P. Baillie-De Byl. *Programming Believable Characters for Computer Games (Game Development Series).* Charles River Media, Inc. Rockland, MA, USA, 2004.

[7] H. Barber and D. Kudenko. Dynamic Generation Of Dilemma-Based Interactive Narratives. In *Proceedings of the Artificial Intelligence and Interactive Digital Entertainment conference (AIIDE)*, 2007.

[8] C. Bartneck. Integrating the OCC Model of Emotions in Embodied Characters. 2002.

[9] J. Bates, A.B. Loyall, and W.S. Reilly. An Architecture for Action, Emotion, and Social Behavior. *Lecture Notes In Computer Science*, 1994.

[10] B.J. Biddle and E.J. Thomas. *Role Theory: Concepts And Research.* John Wiley, 1966.

[11] D.M. Bourg and G. Seemann. *AI for Game Developers.* O'Reilly Media, Inc., 2004.

[12] H. Bowen. Can Video Games Make You Cry. *GameInformer Magazine*, 2005.

[13] D. Canamero. Modeling Motivations And Emotions As A Basis For Intelligent Behavior. In *Proceedings of the first international conference on Autonomous agents.* Citeseer, 1997.

[14] C.S. Carver, S.K. Sutton, and M.F. Scheier. Action, Emotion, and Personality: Emerging Conceptual Integration. *Personality and Social Psychology Bulletin*, 26(6), 2000.

[15] S. Cass. Mind games. *IEEE Spectrum*, 39(12), 2002.

[16] C. Crawford. *Chris Crawford on Interactive Storytelling (New Riders Games).* New Riders Games, 2004.

[17] A.R. Damasio and S. Sutherland. *Descartes'error: Emotion, reason, and the human brain.* Papermac London, 1996.

[18] J.S. de Freitas, R. Imbert, and J. Queiroz. Modeling Emotion-Influenced Social Behavior for Intelligent Virtual Agents. *Lecture Notes in Computer Science*, 4827, 2007.

[19] J. Dias and A. Paiva. Feeling and Reasoning: A Computational Model for Emotional Characters. *Lecture notes in computer science*, 3808, 2005.

[20] W.E. Dir. Andrew Stanton. *Pixar Animation Studios/Walt Disney Pictures*, 2008.

[21] P. Ekman. An Argument For Basic Emotions. *Cognition & Emotion*, 6(3), 1992.

[22] Jon Evans. Can a video game make you cry? Retrived January 1 2009 From `http://www.maisonneuve.org/index.php?&page_id=12&article_id=3203`, 2009.

[23] T. Fong, I. Nourbakhsh, and K. Dautenhahn. A Survey Of Socially Interactive Robots. *Robotics and Autonomous Systems*, 42(3-4), 2003.

[24] N.H. Frijda. *The Laws of Emotion.* Lawrence Erlbaum Assoc Inc, 2006.

[25] J. Gold. *Object-Oriented Game Development.* Addison-Wesley, 2004.

[26] J. Gratch and S. Marsella. A Domain-Independent Framework For Modeling Emotion. *Cognitive Systems Research*, 5(4), 2004.

[27] J. Gratch and S. Marsella. Evaluating a Computational Model of Emotion. *Autonomous Agents and Multi-Agent Systems*, 11(1), 2005.

[28] L. Gruenwoldt, M. Katchabaw, and S. Danton. Creating Reactive Non Player Character Artifical Intelligence In Modern Video Games. In *Proceedings of the 2005 GameOn North America Conference*, 2005.

[29] A. Guye-Vuilleme and D. Thalmann. A High-Level Architecture For Believable Social Agents. *Virtual Reality*, 5(2), 2000.

[30] D. Higgins. Pathfinding Design Architecture. *AI Game Programming Wisdom*, 2002.

[31] M. Hoeberechts, R. Demopoulos, and M. Katchabaw. A Flexible Music Composition Engine. In *Proceedings of Audio Mostly 2007: The Second Conference on Interaction with Sound*, 2007.

[32] R. Imbert and A. De Antonio. When Emotion Does Not Mean Loss of Control. *Lecture notes in computer science*, 3661, 2005.

[33] K. Isbister. *Better Game Characters by Design: A Psychological Approach.* Morgan Kaufmann Pub, 2006.

[34] M. Johns and B.G. Silverman. How Emotions And Personality Effect The Utility Of Alternative Decisions:A Terrorist Target Selection Case Study. In *Tenth Conference On Computer Generated Forces and Behavioral Representation*, volume 453, 2001.

[35] D. Keltner and A.M. Kring. Emotion, social function, and psychopathology. *Review of General Psychology*, 2, 1998.

[36] S. Kshirsagar. A Multilayer Personality Model. In *Proceedings of the 2nd international symposium on Smart graphics.* ACM New York, NY, USA, 2002.

[37] J.E. Laird. Using a Computer Game to Develop Advanced AI. *Computer*, 2001.

[38] J.E. Laird and M. Van Lent. Human-Level AIs Killer Application. *AI magazine*, 22(2), 2001.

[39] P. Lankoski and S. Bjork. Gameplay Design Patterns for Believable Non-Player Characters. In *Proc. 3rd Digital Games Research Association International Conference*, 2007.

[40] D. Livingstone. Turing's Test And Believable AI In Games. *Computers in Entertainment (CIE)*, 4(1), 2006.

[41] A.B. Loyall. *Believable Agents: Building Interactive Personalities.* PhD thesis, Stanford University, 1997.

[42] B. Mac Namee and P. Cunningham. Proposal for an Agent Architecture for Proactive Persistent Non Player Characters. In *Proceedings of the Twelfth Irish Conference on Artificial Intelligence and Cognitive Science*, 2001.

[43] B. Magerko. Proposal for an Agent Architecture for Proactive Persistent Non Player Characters. *Journal of Game Development*, 2(3), 2007.

[44] B. Magerko, J.E. Laird, M. Assanie, A. Kerfoot, and D. Stokes. AI Characters and Directors for Interactive Computer Games. *Ann Arbor*, 1001.

[45] M. Mateas. An Oz-centric Review of Interactive Drama And Believable Agents. *Lecture notes in computer science*, 1600, 1999.

[46] M. Mateas and A. Stern. A Behavior Language for Story-Based Believable Agents. *IEEE Intelligent Systems*, 2002.

[47] M. McNaughton, J. Schaeffer, D. Szafron, D. Parker, and J. Redford. Code Generation For AI Scripting in Computer Role-Playing Games. In *Challenges in Game AI Workshop at AAAI*, volume 4.

[48] M. Minsky. *The Society Of Mind.* Simon & Schuster, Inc. New York, NY, USA, 1986.

[49] L. Morgado and G. Gaspar. Emotion Based Adaptive Reasoning Ror Resource Bounded Agents. In *Proceedings of the fourth international joint conference on Autonomous agents and multiagent systems.* ACM New York, NY, USA, 2005.

[50] P. Nilsson. *Empathy And Emotions: On The Notion Of Empathy As Emotional Sharing.* Philosophy and Linguistics, 2003.

[51] J. Odell, M. Nodine, and R. Levy. A Metamodel For Agents, Roles, And Groups. *Agent-Oriented Software Engineering (AOSE) V*, 2005.

[52] J. Odell, H.V.D. Parunak, S. Brueckner, and J. Sauter. Changing Roles: Dynamic Role Assignment. *Journal of Object Technology, ETH Zurich*, 2(5), 2003.

[53] J.J. Odell, H. Van Dyke Parunak, and M. Fleischer. The Role of Roles in Designing Effective Agent Organizations. *Lecture notes in computer science*, 2003.

[54] J. Orkin. Symbolic Representation of Game World State: Toward Real-Time Planning in Games. In *Proceedings of the AAAI Workshop on Challenges in Game Artificial Intelligence*, 2004.

[55] A. Ortony. On Making Believable Emotional Agents Believable. *Emotions in humans and artifacts*, 2003.

[56] A.A. ORTONY and A.A. COLLINS. *The Cognitive Structure of Emotions.* Cambridge university press, 1988.

[57] P. Panzarasa, N.R. Jennings, and T.J. Norman. Social mental shaping: Modelling the impact of sociality on the mental states of autonomous agents. *Computational Intelligence*, 17(4), 2001.

[58] H.V.D. Parunak, R. Bisson, S. Brueckner, R. Matthews, and J. Sauter. A Model Of Emotions For Situated Agents. In *Proceedings of the fifth international joint conference on Autonomous agents and multiagent systems.* ACM New York, NY, USA, 2006.

[59] R.W. Picard. *Affective Computing.* MIT Press, 1997.

[60] H. Prendinger and M. Ishizuka. Evolving Social Relationships With Animate Characters. In *Proceedings of the AISB-02 Symposium on Animating Expressive Characters for Social Interactions*, 2002.

[61] Columbia World Press. Columbia world press of quotations. Retrieved November 17 2008 From `htttp://www.bartleby.com/66/78/12878.html`.

[62] S. Rabin. *AI Game Programming Wisdom.* Charles River Media, 2002.

[63] J Rajnovich. Artificial inteligence for games. 12 2004.

[64] Adam Ranking. In progress. Master's thesis, University Of Western Ontario, 2009.

[65] A.S. Rao and M.P. Georgeff. BDI Agents: From Theory to Practice. In *Proceedings of the first international conference on multi-agent systems (ICMAS-95).* San Francisco, 1995.

[66] S. Reiss. Multifaceted Nature of Intrinsic Motivation: The Theory of 16 Basic Desires. *Review of General Psychology*, 8(3), 2004.

[67] M.O. Riedl and R.M. Young. An Intent-Driven Planner for Multi-Agent Story Generation. In *Proceedings of the Third International Joint Conference on Autonomous Agents and Multiagent Systems-Volume 1.* IEEE Computer Society Washington, DC, USA, 2004.

[68] P. Rizzo, M. Veloso, M. Miceli, and A. Cesta. Personality-Driven Social Behaviors In Believable Agents. In *Proceedings of the AAAI Fall Symposium on Socially Intelligent Agents.* AAAI Press, 1997.

[69] D.L. Roberts and C.L. Isbell. Desiderata For Managers Of Interactive Experiences: A Survey Of Recent Advances In Drama Management. In *Proceedings of the First Workshop on Agent-Based Systems for Human Learning and Entertainment (ABSHLE 07)*, 2007.

[70] D.M. Romano, G. Sheppard, J. Hall, A. Miller, and Z. Ma. BASIC: A Believable Adaptable Socially Intelligent Character for Social Presence. In *Proceedings of the 8th Annual International Workshop on Presence.* Citeseer, 2005.

[71] M. Scheutz, A. Sloman, and B. Logan. Emotional States And Realistic Agent Behaviour. *Proceedings of GameOn*, 2000.

[72] S. Spielberg, S. Zaillian, T. Keneally, J. Kaminski, and J. Williams. Schindler's list. 1994.

[73] E. Steegmans, D. Weyns, T. Holvoet, and Y. Berbers. Designing Roles for Situated Agents. In *5th International Workshop on Agent-Oriented Software Engineering*, volume 210. Citeseer, 2004.

[74] C.R. Strong, M. Mehta, K. Mishra, A. Jones, A. Ram, and N.L.G. Approach. Emotionally Driven Natural Language Generation for Personality Rich Characters in Interactive Games. *AIIDE 2007*, 2007.

[75] P. Sweetser. An Emergent Approach to Game Design–Development and Play. *School of Information Technology and Electrical Engineering, The University of Queensland University of Missouri, Columbia.*, 202006, 2006.

[76] P. Sweetser. *Emergence in Games.* Charles River Media, Game Development Series, 2008.

[77] P. Sweetser, D. Johnson, J. Sweetser, and J. Wiles. Creating Engaging Artificial Characters For Games. In *Proceedings of the second international conference on Entertainment computing.* Carnegie Mellon University Pittsburgh, PA, USA, 2003.

[78] M. Theune, S. Faas, A. Nijholt, and D. Heylen. The Virtual Storyteller: Story Creation by Intelligent Agents. In *Proceedings of the Technologies for Interactive Digital Storytelling and Entertainment (TIDSE) Conference*, 2003.

[79] M. Theune, S. Rensen, R. Akker, D. Heylen, and A. Nijholt. Emotional Characters for Automatic Plot Creation. *Lecture Notes In Computer Science.*, 2004.

[80] Clive Thompson. Can a game make you cry? Retrived January 12 2009 From `http://www.wired.com/gaming/virtualworlds/commentary/games/2005/11/69475`, 11 2005.

[81] S. Thorpe, D. Fize, and C. Marlot. Speed Of Processing In The Human Visual System. *Nature*, 381(6582), 1996.

[82] P. Tozour. The evolution of game AI. *AI Game Programming Wisdom*, 2002.

[83] J. Velasquez. Modeling Emotion-Based Decision-Making. *Emotional and Intelligent: The Tangled Knot of Cognition*, 1998.

[84] J.D. Velasquez. Modeling Emotions and Other Motivations in Synthetic Agents. In *Proceedings Of The National Conference On Artificial Intelligence.* John Wiley & Sons Ltd, 1997.

[85] J.D. Velásquez. When Robots Weep: Emotional Memories and Decision-Making. In *Proceedings Of The National Conference On Artificial Intelligence.* John Wiley & Sons Ltd, 1998.

[86] M. Wooldridge, N.R. Jennings, and D. Kinny. The Gaia Methodology for Agent-Oriented Analysis and Design. *Autonomous Agents and Multi-Agent Systems*, 3(3), 2000.

[87] Jiaming You. Comprehensive believable non player characters creation and management tools for emergent gameplay. Master's thesis, University Of Western Ontario, 2009.