PACM: Player Archetype Change Management System in Role-Playing Games

Natham Thammanichanon and Vishnu Kotrajaras
Department of Computer Engineering
Chulalongkorn University, Bangkok, Thailand
E-mail: natham.t@student.chula.ac.th, vishnu@cp.eng.chula.ac.th

KEYWORDS
User model, Adaptive, Interactive narrative, Case-based planning, Commercial game.

ABSTRACT

An everlasting endeavour of interactive narrative is still the enrichment of impressive storytelling experience while preserving the variety of interaction for players. Most drama management researches usually concentrate on preserving authorial goals while still allowing some degree of interaction with players. However this approach is not geared towards fulfilling the satisfaction of audiences who do not enjoy those goals. It puts limits on alternative story arcs. Our research focuses on creating a drama management system that selects narratives that match a player's personality and adjusts it through player's altered characteristic during gameplay. This paper proposes PACM (Player Archetype Change Management system in role-playing games) and a player personality model to be used with it. At any one time, PACM initiates and selects story that suits a player's personality automatically from existing stories in a story database, using real-time monitored player's traits. It then narrates the story.

1 INTRODUCTION

There are many researches and developments related to computer games. However, most of them do not concern game narrative components. Therefore it is still common practice to offer players choices during gameplay in order to make a game story progress in different directions. Using this conventional method does not allow an open-free world to coexist with great narratives. Composing a great story requires great expertise. It is a time consuming process and not all players are guaranteed to like the story.

There are many efforts to find the solution to solve this problem, such as combining the elements of a story by drama theory or story model, or using needs and beliefs of NPC to narrate the story. Most of them focus on how to narrate a story according to the goal set by the author while still provide feasible player interaction. There is still not much amount of work that seriously focuses on how to tell a well crafted story from a set of stories that also satisfies any player.

One solution is a *drama manager* (DM) that monitors a story progress, readjusts the game world to be suitable for a player according to his actions. Using only action statistics, however, does not provide full understanding of player preferences. Player archetypes are needed. Several works make use of player models but they usually boil down to managing narratives in order to conserve the authorial goal. This is contrary to our research. We believe that different kinds of players favour different kinds of stories. Hence our approach to interactive narrative focuses on creating a drama management system which aims to offer the most decent narrative for players with less emphasis on sticking to the original goal of the author. We use player personality modeling as a crucial key to overcome such challenge.

This paper presents player personality models and PACM, a drama management system that uses the player personality models and manages stories from a computer game. The drama manager translates a player's actions to player's personality model then uses it to indicate which story should be narrated subsequently in order to gratify the player. It runs on Neverwinter Nights (NWN) game (BioWare 2008) environment using NWNScript, jRCEI (Peinado F. 2007) and DLModel (Peinado F. 2008) for system implementation.

In section 2 we discuss related works. We present PACM system in detail in section 3. Our player personality model is explained in section 4. Discussion about our evalution and experiment are presented in section 5. Section 6 concludes the paper.

2 RELATED WORKS

Many researches explored the utility of enacting an interactive narrative with more appropriate and adaptable AI. Here are some examples of them.

One approach is a character-driven narrative, which relies on interaction between players and the environment enlivened by artificial self-determining characters. Cavazza (Cavazza et al. 2002) uses Hierachical Task Networks for each agent decision replanning. Stories develop according to interactions between all characters and their environment. Therefore it is very difficult to control narratives generated by such mechanishm. Players may not see a story plot at all.

An alternative is a plot-driven approach. In this approach, story elements are selected based on the story's history, character relationships and authorial goals.

Some works focus on plot or event generation like DINAH (Ventura and Brogan 2002). DINAH generates plots through the composition of primitive elements from story clips database constrained by preconditions and postconditions according to Braginan cinematic narrative model. Façade (Mateas and Stern 2003) has a beat-based drama manager which orders primitive elements of a story, called beats. The drama manager builds its story from beats. It chooses its next beat from the story arc and player actions. Fairclough (Fairclough and Cunningham 2003) uses his story director to plan a narrative by retrieving similar story cases using game information. Each story contains character actions and performed roles. This method retrieves the story based on current game information and player actions. These works, although use various aspects to assemble their story, do not make use of player archetypes.

An interactive narrative architecture proposed by Young (Young et al. 2004) generates plans annotated with a rich

causal structure, monitors player actions, replans or prevents actions that are story threats. Bates' (Bates 1992) handles *search-based drama management* like an optimisation problem similar to a minimax game-tree search that has been used in chess-like games. It finds actions to guide the players that maximise the evalution function for making each player follow author's story. Magerko (Magerko et al. 2004) proposes an architecture that takes a prewritten plot and autonomous characters. Its story director manages a plot by guiding nonplayer characters to take corrective action if a player character is likely to affect the plot. El-Nasr (El-nasr 2004) presents *Mirage*, which utilises player models analysed from player's behaviour. Player models are used for changing the behavior of nonplayer characters to encourage players to achieve the original story's goal. These works focus on keeping the player on the plot. Our research, on the other hand, tries to change the story according to a player changing his playing model.

Some research uses machine learning. *Declarative optimization-based drama management* (Nelson et al. 2006) uses reinforcement learning (learned from simulated random players) to predict how a player is trying to shape his story. The drama manager adjusts its actions, such as providing extra guides, in response to player actions.

Sharma utilises a drama manager with a player preference model (Sharma et al. 2007). The model represents a player's interest in his played story path. This method constructs a player model by having the player fill in a questionnaire after he finishes a game. The player's likes and dislikes are recorded from the questionnaire. When a new player plays the game, his actions are compared with recorded actions from existing players. If a match or near-match is discovered, the game will try to steer events towards the existing player's likes and dislikes for the new player. Our work differs from this work because we construct player models from players actions in real time. Sharma's work does not select a new story, unlike our work.

Methodologies mentioned in this section have no support for the case where a player dislikes the author's goal. Instead of trying to maintain the goal, substituting the existing story with a more suitable story (and goal) is preferable. This is our approach.

3 PACM

The implementation of PACM system is shown in figure 1. It consists of three modules , which are 1) *a game connector* module, responsible for actually sending game commands and receiving facts from NWN game, 2) *a player personality modeling* module, responsible for analysing actions of a current player, developing the player's model and updating it in real-time, 3) *a drama manager* module, responsible for influencing the game progress and making it more appealing to the player according to the player personality model and its change during gameplay.

3.1 Game Connector Module

The game connector acts as a middleman between the NWN game engine and other parts of PACM. It is split into two parts. The first part is in the NWN game module, implemented using NWScript to monitor and execute game commands. The other part is in PACM, which continuously sends commands, retrieves game facts and stores them as part of game information. Figure 2 illustrates the game information. This information is sent to the player personality modeling module. Facts such as characters' actions, changed relation and updated environment data are stored as character states and relation in the game information. Player states and actions are separated from other information. A collection of scenes is stored as story state. A scene is a narrative element composed of its preconditions, character's actions and postconditions in XML format, as shown in figure 3. Also all the game commands from the drama manager are sent and executed by the game connector.

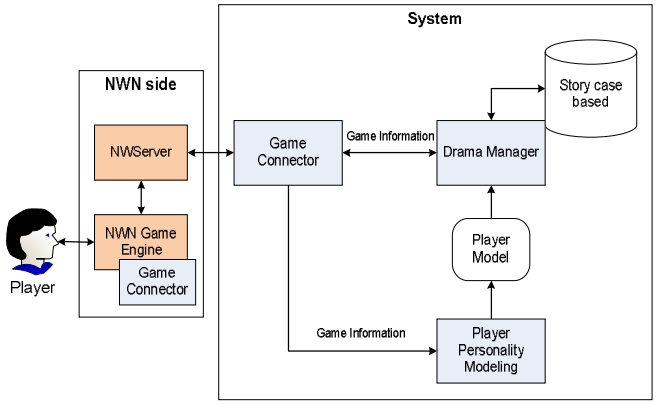

Figure 1: Basic scheme of the components in our approach.

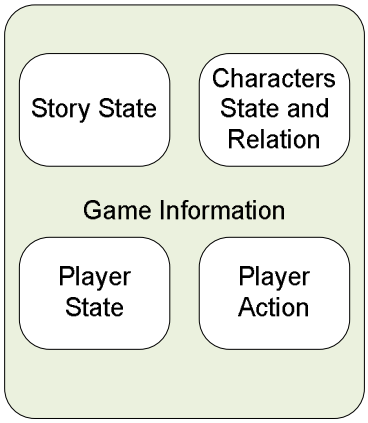

Figure 2: Game information used in the system

3.2 Player Personality Modeling Module

The Player Personality Modeling Module (PPMM) creates, maintains and updates a player personality model for a player of the current game in real time. It updates the player personality model by observing player actions. Each player action matches at least one type of player personality archetypes. Collecting and analysing actions allow us to determine how a player likes to play his game. Stories can therefore be chosen appropriately.

When a game session starts, PPMM obtains the player character status from the game connector and creates an initial personality model which has low confidence value. An initial story is chosen according to this model. During gameplay, each player action is monitored to be used for updating the current player personality model. If a player's actions are consistent with a personality model currently in use or contribute to the story progression, then the current model gets more confidence. This represents the player's

fondness of the current story. However, if actions do not belong to the currently used player personality model, the model gets less confidence, meaning the player is not interested in the story. When the confidence is below a threshold value, the current player personality model changes. The drama manager then alters the story to a new one that is more suitable for the new player personality model.

```
<scene>
<indexChapter> 2 </indexChapter>
<indexScene> 9 </indexScene>
<preCondition>
        <characterRelation>
                <characterName> player </characterName>
                <relation>
                        <name> is </name>
                        <object>
                                <name> boarHeadInn </name>
                        </object>
                </relation>
        </characterRelation>
</preCondition>
<event>
<RCEI>
        <messageNumber> 0 </messageNumber>
        <command>
                <subject> lucinda </subject>
                <predicate>
                        <process> speak </process>
                        <dirComp> player </dirComp>
                </predicate>
        </command>
</RCEI>
</event>
<postCondition>
</postCondition>
</scene>
```

Figure 3: Scene example

3.3 Drama Manager Module

The input to the Drama Manager Module (DMM) are: 1) the player personality model handled by the PPMM, 2) the current game state. With this information, the goal of the DMM is to provide the most suitable story for a player. The underlying assumption behind this is each player can vary his playing styles.

When a game starts, after PPMM finishes player personality model initialisation, the DMM compares this model with other personality models for each story in the case base. A story which has the closest matching personality model is selected. DMM then prepares the game environment to narrate the story.

DMM executes its narrative in sequence. If the current player personality model has its confidence value above a given threshold value, DMM continues the story in its usual way. However, when the personality model's confidence is below the threshold value, DMM must take action. It will first check whether the player is involving with a character that plays a crucial role in the current story. If so, DMM then continues with its current story until the player no longer has anything to do with the character (the PPMM still keeps on updating the player personality model confidence). If not, DMM will search for a new story that matches the current player personality model the most and applies the new story. The new story is initialised with its initial confidence value. The previous story status and information are stored for use later. Therefore, when the current player personality model changes back to a model that was used before, DMM can restore a non-finish story status and information in order to continue the story accurately.

4 MODELING PLAYER PERSONALITY

Our player personality model is based on Bartle's "player category" (Bartle 2004) which describes player categorisation into {*achiever, explorer, socialiser, killer*}. A model is formed from a combination of the percentage of these categories and its confidence value. An example is shown in figure 4. The confidence value represents how much a current player is trying to follow the story (and the player personality model that matches the story). A player's action always updates the player personality model. The updated model is then compared with the starting model by using equation (1). If the difference between them is more than a given value, the confidence value decreases. How it decreases is defined in equation (2). At the same time, the confidence value can increase according to equation (3) and (4). This mechanism implies that a single action that does not match the starting model, but is an unintended action by the player, cannot make an immediate impact on the confidence value. When the confidence value is below the threshold value, the drama manager attempts to change the story.

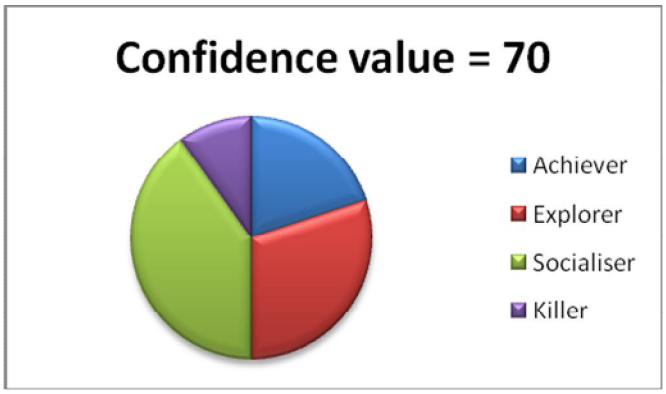

Figure 4: An example of player personality model

$$D_{ij} = \sum_{c=1}^{C} (P_i^c - P_j^c) \quad \text{---(1)}$$

When D_{ij} = distance from model$_i$ to model$_j$
C = {*achiever, explorer, socialiser, killer*}
c = a member of set C
P_i^c = the personality c score of model$_i$

$$confidence'(p) = confidence(p) \times (1 - 0.05 \times \frac{D_{op}}{T}) \text{ -(2)}$$
$$confidence'(p) = 1 \quad if\ (confidence'(p) < 0)$$

When $confidence(p)$ = player's current model confidence
$confidence'(p)$ = player's updated model confidence
D_{op} = distance from player's original model to player's current model
T = threshold value

$$confidence'(p) = (1.1) \times confidence(p) \quad \text{---(3)}$$
if the player progresses the story.

$$confidence'(p) = confidence(p) \times (1 + \delta P_p^c) \text{ -(4)}$$
if distance is not more than the given value.

When δP_p^c = the difference between updated score of personality c and the current score of c

For example, if an initial player personality model is { *achiever* 10%, *explorer* 50%, *socialiser* 40%, *killer* 0%} and its confidence value is 70, when the player goes into an inn and talks to some non-story-relate NPCs. His talking actions will update his personality model to {*achiever* 6%, *explorer* 52%, *socialiser* 42%, *killer* 0%}. The distance value between this new model and the previous model is 24, but the given value for the distance between models is 1200. Therefore the difference does not exceed the given value. Hence the original model's confidence value is increased to 72 by equation (4).

On the other hand, if the player talks with other characters very frequently, his model may become {*achiever* 0%, *explorer* 30%, *socialiser* 70%, *killer* 0%}. This has its distance value from the starting model more than our given limit. Therefore the confidence value will decrease instead.

5 EXPERIMENTS

This section will discuss the result of using PACM with NWN to narrate stories and evaluate players' playing styles.

Stories used in this experiment are brought and adapted from Dungeon issue #93 (Wizards of the Coast, Inc., 2002), #95 (Paizo Publishing, 2002), #102 (Paizo Publishing, 2003a) and #103 (Paizo Publishing, 2003b).

We first asked each participant to evaluate himself according to Bartle's model. Each of them assigned a percentage score to each Bartle's category. This is shown in table 1. Then the participants were asked to play a NWN game augmented with our system until any one story was completed (story could change in-between) for each particant. A screenshot of a game session is shown in figure 5.

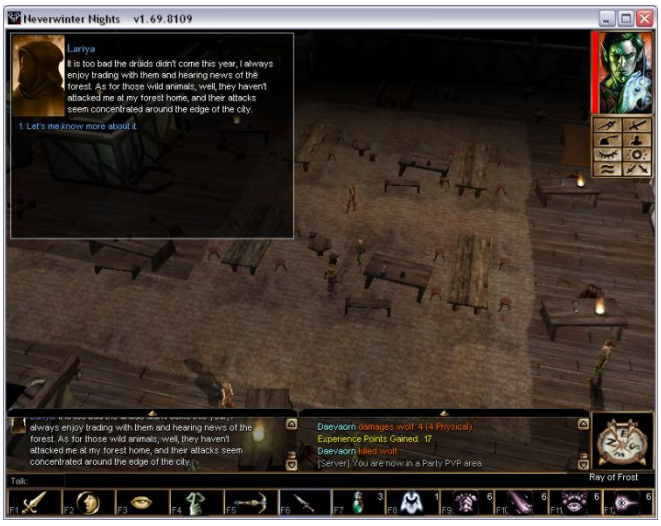

Figure 5: Example of the experiment on NWN.

The system generated an initial player personality model for each player using information from the character creation screen (see table 2). Only about half of the players had been given matching personality models. What important was whether the model and story could change to match a player's style during play. Table 3 shows each player model just after they finished playing a story. The models followed how players wanted to play for six of the seven players we recruited. This is most evident for player 1. Player 1's main archetype, according to his opinion, were explorer and socialiser. The initial model given just after his character creation, however, had everything almost equal, with socialiser receiving the lowest score. During gameplay, our system adjusted the story to a new one more suited to his style, reducing the achiever aspect and increasing his explorer and socialiser aspects to almost equal to the values given by his opinion.

For player 5, the only player whose story did not suit his actual playing style, his initial personality model generated by the system was very different from the player's model. This was due to the player opted not to create a character, but use a character provided to him by the game instead. During gameplay, the chosen story ended quickly. This prevented the system from being able to alter the story in time. Nevertheless, the system showed promise. His explorer trait value, one of the main trait value the player admitted being his play style, increased for quite great amout for the short period that he was playing. His achiever and killer trait values also reduced towards the values of his intended style.

Table 1: Players archetype according to their opinion.

Player	**Player archetype by opinion**			
	Achiever	Explorer	Socialiser	Killer
P1	15	50	25	10
P2	35	30	0	35
P3	15	35	40	10
P4	15	40	35	10
P5	15	40	40	5
P6	20	30	20	30
P7	10	40	10	40

Table 2: Initial predicted archetype.

Player	**Initial predicted archetype**			
	Achiever	Explorer	Socialiser	Killer
P1	28.125	25	21.875	25
P2	37	25	8	30
P3	18.75	31.25	34.375	15.625
P4	20.3125	25	32.8125	21.875
P5	35.59	28.82	0	35.59
P6	29.82	38.6	0	31.58
P7	28.57	47.62	0	23.81

Table 3: Observed archetype after finishing a story.

Player	**Observed archetype**				
	Achiever	Explorer	Socialiser	Killer	Confidence
P1	7	43	28	22	65.49
P2	42	24	0	34	1349.16
P3	17.67	32.86	33.23	16.24	515.27
P4	18.86	27.17	31.97	22	393.96
P5	31.7	34.72	0.45	33.13	142
P6	27.23	39.81	1.44	31.52	111.01
P7	22.93	50.25	0.57	26.25	69.93

A comparison between an archetype from a player's opinion, an initially predicted archetype and an observed archetype at the end of a game for one player is shown in

figure 6. Due to the reason of space, we can only show one player. Most players generate similar results. The score for each category from the players' model is close to the score that each player gave himself. Figure 7 shows that the average distance between the model from player's opinion and the model after playing is 641.28. This value is reduced from the average distance between the model from players' opinion and the originally predicted model, which was at 866.15. This indicates the ability of PACM to recognize a player's style and adjust the story according to that player's style.

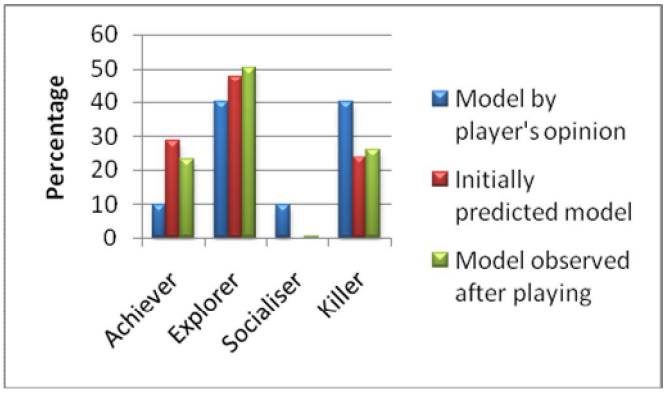

Figure 6**:** P7's compared personality model graph.

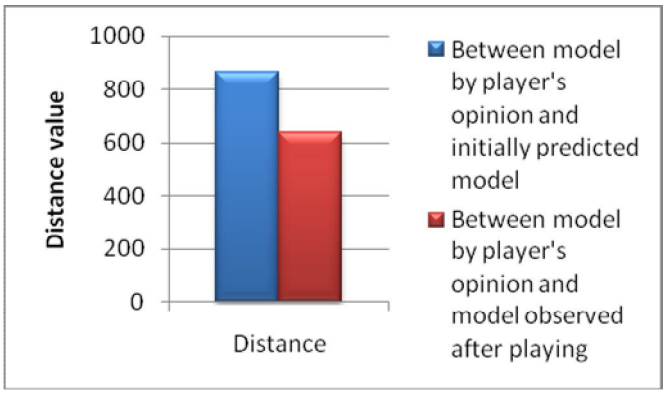

Figure 7**:** Average distance from two comparisons.

6 CONCLUSION AND FUTURE WORK

In this paper, we have proposed a system to handle the drama management problem in interactive narrative. Our approach intergrates player personality modeling and a case-based drama manager which uses the model to maintain the story that satisfies players. We present an implementation of this approach with Neverwinter Night game, and perform initial experiments that confirm the usability of our approach.

Our major contributions are (1) the inclusion of a player personality model which represents current player's actual playing style and its real-time updating. (2) the inclusion of a confidence value for the personality model, which expresses player's affection for the narrative (3) the connection module between the drama management module and a real-time role-playing game that allows us to perform real evaluation on an actual commercial game, with real players.

As part of our future efforts, we plan to experiment with personality models other than Bartle's. Moreover, we plan to use an AI learning technique to adapt the personality model evaluation. Models available from players who finish a story can be learned to correct and adjuxt the mapping between stories and personality models.

REFERENCES

Bartle R. A. 2004. Designing Virtual Worlds. New Riders Publishing.

Bates J. 1992. Virtual reality, art, and entertainment. The Journal of Teleoperators and Virtual Environments, 2(1):133-138.

Bioware. (2008). Neverwinter Nights game. http://nwn.bioware.com/

Cavazza, M., F. Charles and S. J. Mead. 2002. Character-Based Interactive Storytelling, IEEE Intelligent Systems, July/August 2002, pp 17-24.

El-nasr, M. S. 2004. A User-Centric Adaptive Story Architecture: Borrowing from Acting Theories. Proceedings of the ACM SIGCHI International Conference on Advances in computer entertainment technology.

Fairclough, C. R. and P. Cunningham. 2004. AI structuralist storytelling in computer games. Proceedings of the International Conference on Computer Games: Artificial Intelligence, Design and Education.

Magerko, B., J. Laird, M. Assanie, A. Kerfoot and D. Stokes. 2004. AI characters and directors for interactive computer games. Proceedings of the 2004 Innovative Applications of Aritificial Intelligence Conference.

Mateas, M. and A. Stern. 2003. Intgrating plot, character, and natural language processing in the interactive drama Façade. Proceedings of the 1st International Conference on Technologies for Interactive Digital Storytelling and Entertainment (TIDSE-03).

Nelson, M., M. Mateas, D. Roberts and C. Isbell. 2006. Declarative optimization-based drama management in interactive fiction. IEEE Computer Graphics and Applications 26(3):33-41.

Paizo Publishing. 2002. LUST. Dungeon issue #95.

Paizo Publishing. 2003a. CRY WOLF. Dungeon issue #102.

Paizo Publishing. 2003b. FOREST of BLOOD. Dungeon issue #103.

Peinado, F. (2008) DLModel, a tool for dealing with description logics. http://federicopeinado.com/projects/dlmodel/ (last access on Aug 2008)

Peinado, F. 2007. RCEI: An API for Remote Control of Narrative Environments. Proceedings of the 4th International Conference on Virtual Storytelling (ICVS-07).

Sharma, M., S. Ontanon, C. Strong, M. Mehta and A. Ram. 2007. Towards player preference modeling for drama management in interactive stories. Proceedings of the Twentieth International FLAIR Conference on Artificial Intelligence. AAAI.

Ventura, D. and D. Brogan, 2002. Digital Storytelling with DINAH: dynamic, interactive, narrative authoring heuristic. IFIP First International Workshop on Entertainment Computing.

Wizards of the Coast, Inc. 2002. THE STATUE GALLERY. Dungeon issue #93.

Young R.M., M. Riedl, M. Branly, A. Jhala, R. Martin and C. Sagretto. 2004. An architecture for intergrating plan-

based behavior generation with interactive game environments. Journal of Game Development 1(1).