AISB Journal

The Interdisciplinary Journal of
Artificial Intelligence and the Simulation of Behaviour

Volume 2 – Number 1 – April 2010

The Journal of the
Society for the Study of Artificial Intelligence
and the Simulation of Behaviour
http://www.aisb.org.uk

Published by
The Society for the Study of Artificial Intelligence and Simulation of Behaviour

http://www.aisb.org.uk/

ISSN 1476-3036

Contents

AISB Journal **2**(1), © SSAISB, 2010
http://www.aisb.org.uk

Introduction to the AISBJ Special Issue on Artificial Intelligence Narrative and Games

Sandy Louchart and Judy Robertson

School of Mathematical and Computer Sciences, Heriot-Watt University
Riccarton, Edinburgh EH14 4AS, Scotland
Sandy@macs.hw.ac.uk ; Judy@macs.hw.ac.uk

Stories play an essential role in our communication, education and development strategies. Roland Barthes once described them as innumerable; communicated by many means, present in many forms and in any time, period, place, society or class. He defined them as Universal, International, trans-historic and cross-cultural. Interactive Storytelling (IS) proposes to add an extra dimension to traditional storytelling and alter the static nature of stories. As a concept, it has proved to be a remarkable ground for theoretical and technical research and its ability to both unite and divide researchers and communities is proof of its significance. While it first originated within Artificial Intelligence (AI) and computer sciences, it is now home to a wide range of exciting multi-disciplinary research and draws inspiration from other disciplines such as linguistics, speech, traditional arts and media, video-gaming, emotion modeling, education and learning, performance arts and digital media. As a research domain, IS has experienced a remarkable growth over the last decade and offered a truly inter-disciplinary discussion forum. It combines passion and technological development and researchers in the field share genuine interests in narratives. The chances are that if you are reading these lines, amongst the various diverging and concurring opinions; it is likely that you probably have yours and feel quite strongly about them!

In recent times, IS researchers have turned some of their attention towards video games as games are increasingly regarded as the potential medium for the fulfillment of their visions. After all, the rapprochement between IS and games is somewhat logical as both share similar challenges in merging interactivity and narratives. It is also likely that the dream game title in the heads of creative directors and game designers is in-line with what IS researchers had in mind for some times. The forms and contents might vary, but it is likely that the nature of the experience shares the IS raison d'tre. While the possibilities and potential aspects of Interactive Storytelling are many, the underlying functional requirements are on another hand limited to a set of complicated and vast challenges. Amongst these, believable and autonomous synthetic characters along with plot/story articulation stand as the fundamental elements and basis for any interactive storytelling approach.

In this special issue, we selected articles that covered these theme so as to provide the reader with an general understanding of the issues and challenges that faces Interactive Storytelling and Video Games. Shearer et al offer us with an in-depth review and analysis of non-verbal behaviour for believable synthetic characters. In their paper, they take on the task of describing non-verbal behaviour through the analysis of both agents and gestures within the context of video gaming (i.e. Half -Life). Malfaz et al focus on the autonomy

of agents and synthetic characters through the development of an experimental platform for testing decision-making. in their article they investigate a decision-making system for self-motivated autonomous agents via MUDs (Multi-User Dungeons) games. Gillies et al introduce the Demeanour system and provide an in-depth review of the relationships and interactions of multiple characters in interactive narratives. The article focuses on the interactions between characters and how their behaviours (i.e. gaze) can serve narrative needs. Finally, Brom et al cover plot representation through a thorough and detailed description of Petri-Nets and how can these be used in order to represent Story plots in Serious Games.

The exciting research presented in this special issue only represents a fraction of the difficulties and challenges standing before us on our way to a truly immersive interactive narrative. However, these selected papers cover most of the basis upon which future systems will be built. The Interactive Storytelling field is in constant evolution and with increasing involvement from the video game industry, one can hope the the dream game or dream interactive narrative is closer than one might think.

We sincerely hope you enjoy reading this special issue,

Sandy Louchart and Judy Robertson
School of Mathematical and Computer Sciences, Heriot-Watt University

Non-Verbal Behaviour for Believable Synthetic Agents

John Shearer*, Philip Heslop*, Patrick Olivier* and Marco De Boni†

* Newcastle University, Newcastle upon Tyne, NE1 7RU, United Kingdom, *john.shearer@ncl.ac.uk ; philip.heslop@ncl.ac.uk ; p.l.olivier@ncl.ac.uk*
† Unilever Corporate Research, Unilever R&D Colworth, Sharnbrook, Bedford, MK44 1LQ, United Kingdom, *marco.de-boni@unilever.com*

Abstract

Realism for synthetic characters, both in computer games and conversational agent mediated applications, requires both visual and behavioural fidelity. One significant area of synthetic character behaviour, that has to date received little attention, is non-verbal behaviour. In identifying the scope and participants of non-verbal behaviour in computer games we first review the range of spatial and task scenarios that are relevant. We then select the principal categories of non-verbal behaviour: proxemics, gaze, gesture, self-adaptors and passive communication – summarising their role in communication and propose their incorporation in the design of non-player characters. In conclusion we review the question of how to capture the non-verbal behaviour of players or provide them with interaction techniques that might facilitate non-verbal communication with players and non-player characters alike.

1 Introduction

Perhaps the most significant current challenge for synthetic agents in games is the integration of narrative and intelligent character behaviour with a view to further enhancing user engagement. The development of non-player characters has primarily focussed on high-level coordination of non-player character physical behaviours (typically through the characterisation of state space) and little attention has been paid to low level interactions between characters (both non-player characters and player characters) such as different forms of verbal and non-verbal behaviour.

Realistic interaction between characters is a challenging problem. In simple terms it requires non-player characters to have sophistication in their beliefs, desires, and intentions equivalent to the expectations that a player might have of a real character (within the narrow domain of the game). Although, one should keep in mind the possibility of an 'uncanny valley' (Mori, 1970) – that, as characters become more human-like, with a positive emotional response from a human, there becomes a point where the response suddenly becomes strongly negative – the character is almost human, but seems unusual or strange – something is perceived as being not quite right. This is clearly contrary to the desired effect.

Crucially, non-player characters must also have beliefs as to the beliefs, desire and intentions of other characters, and therefore require some ability to monitor the activity of other characters for intentional behaviour. This then allows for communication (the intentional influence of other's beliefs). This requires a non-player character to distinguish between behaviour that is, and is not, intentionally influencing another's beliefs, as many behaviours are interpreted differently depending on the context. For example, opening a door can be an action (enabling continuation of a route) and communication (showing another person respect). Humans (and many higher mammals) are extremely good at separating basic actions from actions intended to communicate (as a conscious act or otherwise), and, in the course of social interactions, people constantly analyse each other's behaviour (e.g. speaking, moving, gesturing, touching) to understand their (the other's) beliefs, desires and intentions. It has been shown that people also do the same for artificial entities (Reeves and Nass, 1996). That is, people have a tendency to attribute intention to all things they interact with, even if those objects have no possible mechanism from which intention might derive (according to our present scientific understanding of the world), for example, shouting at a household object if it is not working correctly, as if the object intended to go wrong. Inevitably, players ascribe intention (often very complex intention) to non-player characters even when the non-player character's behaviour, and/or the underlying control architecture, is very limited.

Non-player characters in present-day games often incorporate a restricted notion of desire and intention, but have no significant ability to perceive the desires and intentions of other characters (and thereby infer their beliefs and other aspects of internal state). Thus, intention is usually exhibited through direct action – in a first person shooter non-player characters display their intention to frag (kill) a character simply by shooting at that character. Indeed, in games where the role of non-player characters is not so clearly defined, players quickly habituate to repetitive behaviour as this "supporting cast" mills around according to simple rules or scripts.

A discussion of the development of interactions in all areas of games is beyond the scope of our analysis. Instead, we address communication between characters, specifically non-verbal communication through gesture, facial expression, eye gaze and other bodily movement. That is, we address non-verbal interactions between characters that are not direct actions (i.e. not actions such as attacking). Most games portray the intentional-

ity and personality of characters with cut-scenes – scenes inserted, pausing game-play, at various points of the game – with fully animated characters using non-verbal and verbal behaviour in an attempt to provide an engaging narrative for a game. In contrast, during actual game-play a character's behaviour is much simpler, and very little intentionality is usually displayed.

The advent of high performance 3D graphics hardware and software, and high quality audio has given rise to the use of geometrically detailed, skeletally animated characters, that exhibit some non-verbal behaviour. For example, *Half-Life 2* (Valve, 2004) makes extensive use of non-verbal behaviour in non-player characters, to the extent that player engagement and narrative development can occur effectively without the use of cut-scenes. Non-player characters attempt to talk with the player and when engaged in talking to a player character they will: move around to maintain eye contact during the conversation; respond to the players actions; and show expressions and gestures appropriate to the game-play.

While the characters within *Half-Life 2* appear to have desires, personality and intentions, their actual behaviour is, in fact, highly scripted. Most non-player characters have only a relatively small set of simple behaviours and a number of complex scripts that occur only once or twice. These complex behaviours replace the traditional cut-scene, but without the need to break game-play. However, the integrity of the non-verbal behaviour relies on the skill of the artists in coordinating the non-player character's action with the game-play and the non-player character's speech. This can clearly be seen when observing non-player characters' interactions with each other. Aside from pre-scripted cut scenes, non-player characters do not interact with each other, other than to avoid collision. Even then, they still walk on paths that lead to collision, but evade each other at the last moment (sometimes accompanied with a sound effect along the lines of 'watch where you are going'). Indeed, following an individual non-player character through the environment reveals that they are predominantly on pre-fixed paths that simply enter the environment at one point and walk off at another. Some of the inhabitants even walk in a constant looping path around the environment.

Similar behaviour is exhibited in other games, such as *Grand Theft Auto 3* (Rockstar, 2002). Although more densely populated, the non-player characters exhibit similarly aimless behaviour, reacting only to certain events (such as explosions or being attacked) but otherwise wandering through the environment with no apparent destination. There are games in which the behaviour of non-player characters is more sophisticated, such as *The Sims* (Maxis, 2000), where the characters fall half-way between being user controlled and AI controlled. The characters converse with each other and utilise objects in their environment. Though, even in this case the characters exhibit only the desire to satiate certain undesirable internal states, such as hunger and tiredness, by using the objects and characters around them. They have no longer term and more complex goals, and the conversations between characters do not actually convey any specific information but rather the act of conversation itself satisfies some current short-term need; the characters can obtain the same effect by using an object in the environment, such as reading a book or listening to the radio. In practice the scope of application of non-verbal behaviour extends far beyond the replacement of the cut-scene and in the next section we consider the applicability to games of the relevant spatial and task contexts over which non-verbal behaviour is known to be utilised in humans.

2 Spatial and Task Context

Communication can be considered to occur in four different task contexts: cooperation, coaction, competition and conversation (Knapp and Daly, 2002). In other words, communication occurs in order for some number of parties to: perform a task together (cooperation), to exist in the same vicinity (coaction), to perform a task at the expense of another (competition), and to entertain or pass on information (conversation). Communication varies across these different contexts and also with the physical proximity of the communicating parties. As already described, non-verbal behaviour provides information as to the beliefs, desires, and intentions of a character, or alternatively it can be considered as providing indicators as to another character's cognitive, emotional, physical, intentional, attentional, perceptual, interactional and social status. A set of distinct but common communication situations (spatial and task contexts) for computer games is illustrated in Figures 1-6 using screenshots from *Half-Life 2*.

Figure 7 maps out the range of spatial and task context for these examples. Synthesizing non-verbal behaviour for conversational partners at close physical proximity is particularly difficult, due to the full movement of a character (both body and face) being visible in detail to the player. Furthermore, people are highly attuned to interactions in intimate, personal, and social spaces and are sensitive to many subtle cues and nuances in non-verbal behaviour. Thus, at close proximity players are very aware of errors, unrealistic, or unnatural behaviour in non-player character. At further distances less detail of a character's movement is apparent. Moreover, there is a significant transition in non-verbal behaviour from situations where intimate verbal communication is possible to those where it is not. The sensitivity of non-verbal behaviour to proximity is due to a number of factors, including the more public nature of non-verbal gesture in open spaces, and the requirement on particular physical behaviour to carry the full communicational load (e.g. subtleties in gaze and facial expression are not visible at a distance).

Figure 1 shows an example of cooperation in intimate space. The male character demonstrates his attentional state – that he is attending to the female character – with his body orientation, face orientation, and gaze direction. Of course, people are rarely static, but different non-verbal channels (e.g. face orientation, body orientation, gaze direction, body position) are closely coordinated in demonstrating attention. Thus, the male character could look away but still communicate his attention sufficiently through his posture and pose. In an interaction between unfamiliar participants, however, strong or constant facing or looking at a person is widely considered an aggressive signal. It is considered rude, or at least off-putting (Knapp and Daly, 2002). Figure 1 also illustrates non-verbal behaviour using facial expressions and kinaesthetic (touching) behaviour.

Figure 5 illustrates a situation at the other end of the spatial scale, cooperation at a distance between the player and a non-player character (in fact, navigation and negotiation, a subset of cooperation). The non-player character shown and the player will collide if they do not arrive at an agreement as to how to pass one another and communicate this – the characters must cooperate through the use of non-verbal behaviour to resolve a potential conflict. In the real world, people in this situation use a range of subtle non-verbal mechanisms such as gaze and body turning to initiate and mutually negotiate space. Non-player characters in *Half-Life 2* will avoid the player, but will not exhibit non-verbal behaviour in doing so and simply move around the players as they approach. Without non-verbal behaviour it is difficult for the player to decide which way to move out of the way (indeed they do not need to) and this absence of social convention (and the ability to break them, to invite conflict) that both undermines the engagement of players with the game and limits their expressivity.

Between the proximal and distant spatial scales are social spaces and figure 6 is an example of conversation in a social space. Here non-verbal behaviour facilitates a number of aspects of the interaction (and the dialog in particular) including the mediation of conversation flow, such as whose turn it is to speak (interactional state). Turn-taking mediation is a complex coordination of behaviours, but in simple terms speakers provide opportunities to allow the listener to take a turn (such as, a slightly prolonged pause, or a look up into the eyes), at which point other listeners can, if they choose, take a turn. If not, then the speaker will continue. Additionally, the 'speaker' can indicate that they would like to speak, with signals such as increased eye contact, leaning forward or standing taller (Duncan and Fiske, 1977). Turn-taking mediation is not required in *Half-Life 2* because the game developers have not allowed the player to speak, but it is potentially a very important component of games that hope to include natural language interactions (particularly spoken interaction) between player and non-player characters.

Finally, figures 2, 3 and 4 illustrate the remaining task contexts: coaction, conversation and cooperation, and competition. Characters sharing the same approximate area of space engage in coaction behaviour, corresponding to mutual monitoring – this can be interpreted as communication by virtue of the fact that watching a character implies that you might react to it – that is, there is an implied reason (intention) for watching. Coaction can be considered the default task context, which develops into the other contexts. Competition contexts give rise to distinctly different forms of non-verbal behaviour from other contexts, but these still serve to communicate internal states. In figure 4 the raised baton serves to communicate"you have crossed a line – back off or I will hit you".

Figure 1: Cooperation in intimate space

Figure 2: Coaction in social space

Figure 3: Conversation in personal space

Figure 4: Competition in personal space

Figure 5: Cooperation at a distance

Figure 6: Conversation in social space

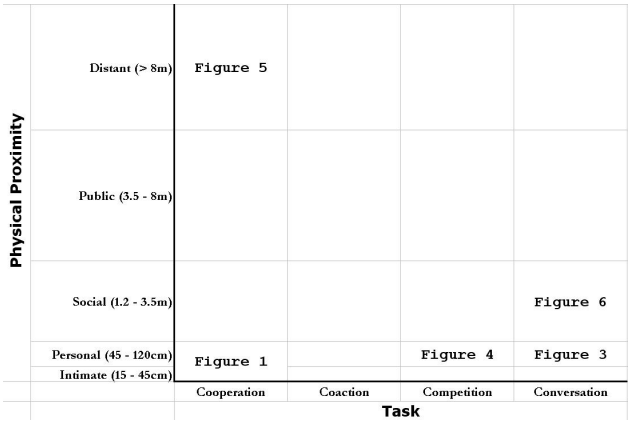

Figure 7: Task and spatial contexts.

Table 1:

3 Non-Verbal Behaviour

Non-verbal behaviour in synthetic characters should be based on the equivalent behaviour exhibited by real people in human-human interactions. Hence, the forms of non-verbal behaviour that are particularly relevant to the synthesis of non-verbal behaviour in virtual characters include:

- Proxemics – (the use and arrangement of the self in the physical world).
- Gaze – (where the eyes are looking and pupil dilation).
- Gesture – (movement of hands and head).
- Self-Adaptors – (movements that serve to alter the self).
- Passive Communication – (communication that occurs without a specific action).

It should be noted that there are very significant individual differences in the precise nature of their non-verbal behaviour. Non-verbal behaviour is influenced by age, gender, social status/hierarchy position, culture, presence of others, illness and physical ability as well as the spatial and task context. This presents problems for both the study and the synthesis of non-verbal behaviour, but also furnishes a potentially rich behavioural vocabulary by which the individuality of characters can be developed. Fidelity in non-verbal behaviour requires that each distinct non-player character employs its own characteristic behaviour in this respect. In the following sections we characterise non-verbal behaviour in people, generally from a Western cultural perspective, with a view to identifying which aspects should be addressed in the design of engaging virtual characters for computer games.

3.1 Proxemic Behaviour (Proxemics)

Proxemic behaviour is defined as the closeness and arrangement of the self in physical world compared to others (Hall, 1966). This is the use of personal space and territory, and although spatial measurements are sometimes difficult to judge in game environments (people rarely feel their personal space is invaded in games), the use of space does have an impact on player behaviour. Moreover, as games become more immersive (due to improved graphics, larger screens, head-mounted displays, immersive environments, and surround sound) and spoken dialog with non-player characters becomes a possible occurrence, the impact of proxemics will increase. As discussed, non-verbal behaviour varies significantly with distance (and spatial arrangement), so its consideration is therefore important in the synthesis of non-player character behaviour.

Proxemics also exhibit some of the largest (but consistent) cultural variations, especially in conversation. In addition to distant space for interaction at over 8 metres (25 feet), where people still interact but do not conduct conversations, we can define, in order of increasing distance (or decreasing proximity), the spaces for UK/US culture (Hall, 1966):

- intimate space for embracing, touching or whispering (15-45 cm, 6-18 inches);
- personal space for interactions among good friends (45-120 cm, 1.5-4 feet);
- social space for interactions among acquaintances (1.2-3.5 metres, 4-12 feet);

- public space used for public speaking (over 3.5 metres, 12 feet).

Proxemic behaviour stems from the idea of territory. Territories vary dynamically and are dependent upon many different factors, but the categories above are likely to be sufficient for basic synthesis. Synthetic characters should use proxemics in the same way that people do – in order to be socially correct use: social space for interacting with relatively unknown people, personal space for people known well and intimate space with people known very well. Violation of these rules is socially incorrect. Invasion of personal space can be intimidating, flirtatious, or can be the result of interaction between participants from different cultural backgrounds. Whilst the requirements of a task, such as needing to be close to attend to a wound, may override the usual norms, in such circumstances the progression through the spaces is still mediated by non-verbal behaviours.

The synthesis of complex proxemic behaviour requires further parametrisation with respect to types of territory: primary, secondary and public (Altman, 1975). Primary territories are the exclusive domain of the owner (such as a home), secondary territories are those felt to be partly owned (such as the local pub), and public territories those available to almost anyone for temporary ownership (such as a park bench). Temporary ownership means a person behaves very differently towards that object or space while they 'own' it (Knapp and Daly, 2002), it becomes part of their territory.

3.2 Gaze and Eye Based Communication

The use of the eyes is an important component of human-human communication, and it involves far more than just what a person is looking at. Peoples' eyes are constantly moving from place to place (not smoothly, but jumping from one locus of attention to another). As Knapp and Daly (2002) note, "we associate various eye movements with a wide range of human expressions: downward glances are associated with modesty; wide eyes with frankness, wonder, naivete, or terror". Where the eyes are looking (or gaze) is the primary (and most obvious) form of communication by gaze, but people are also sensitive to, and react to, pupil dilation/constriction. There is also a close relationship between gaze behaviour and facial expression (not considered here).

Kendon (1967) identifies four functions of gaze behaviour (in addition to looking at specific items for information gathering), and Knapp and Daly (2002) built on this, classifying five functions of gaze as:

1. Regulating the flow of communication.
2. Monitoring feedback.
3. Reflecting cognitive activity.
4. Expressing emotions.
5. Communicating the nature of an interpersonal relationship [added by Knapp].

Within the context of a non-player character, the five categories pose challenges for synthesis that are considerably more challenging than simply having the character look at where it is interested. The regulation of communication flow, gazing briefly at another person (specifically at the face) establishes an obligation to interact; further and longer gazing shows a desire to increase the level of interaction; while decreased and shorter gazing desires a decrease in the level of interaction. During an interaction eye glances serve as turn-taking signals and also highlight grammatical breaks, conceptual unit breaks,

and the ends of utterances (a sequence of speech separated from another by a marked gap). These glances also allow feedback on the interaction by monitoring the reactions of the other person.

When under increased cognitive load (for example trying to process difficult or complex ideas) both listeners and speakers tend to look away. It is thought that this averted gaze reflects a shift in attention from the external to the internal. Additionally, there is evidence that the eye gaze direction under this condition changes with different forms of cognitive load, according to the active hemisphere of the brain (Ehrlichman and Weinberger, 1978; Weisz and Adam, 1993; Wilbur and Roberts-Wilbur, 1985).

The eyes are also a site for the display of emotions: surprise; fear; disgust; anger; happiness; sadness, in addition to blends of these and more complex emotions. Interestingly, in certain contexts emotions displayed with the eyes will not always match the facial emotional expressions (e.g. during emotional masking) and can be very transitory. However, people are adept at detecting emotional state from the eyes, and there is evidence that different emotions are in fact detected from different areas around the face (Ekman et al., 1971; Ekman and Friesen, 1975), but that it is the facial area around the eyes that displays the emotion, not the eyes themselves.

Finally, eye gaze can communicate the nature of an interpersonal relationship. Gazing and mutual gazing is found most in conversations with a high-status addressee, lower with a very high status addressee, and minimal with a very low-status addressee (Hearn, 1957; Efran, 1968). Aggressive encounters and interactions between lovers or mothers and babies have extended periods of mutual gazing.

3.3 Gesture

Gesture is body movement that serves to communicate, mainly involving movement of the hands and the head. There are many forms of gesture, but of particular relevance for non-verbal behaviour synthesis in non-player characters are emblematic gesture (gesture with specific meanings that occur without speech) and spontaneous gesture (hand and head movements that occur with speech). Emblematic gestures are well defined in both their form and meaning and are therefore readily synthesised with standard skeletal animation and scheduling frameworks. For example, the 'come here' gesture is performed using the moving of a finger, fingers, hand, hands, arm, or arms towards the body from the direction of the addressee (often in a repetitive form). Synthesis of spontaneous gesture for non-player characters is significantly more challenging.

Spontaneous gesture is performed by people while speaking, in synchrony with the speech and is generally made with the head or hands. When needs be people are apt to use any available body part, or even the whole body (e.g. pointing with a foot when one's hands are full). Spontaneous gesture continues to be studied in depth by the psychology and psycholinguistics communities, and though most studies are descriptive in nature, recent research on growth points is leading to theories of how gesture relates to other cognitive processes (McNeill, 2005).

It has been found that the gesture stroke (the semantic, or meaningful part) of a gesture commonly coincides with the peak phonological stress – the most emphasised phoneme – of the speech stream. Gesture is tied closely to the underlying speech and both speech and gesture are widely believed to be generated from a single underlying conceptual representation (McNeill, 1992). Indeed, spontaneous gesture can be complementary, supplementary, or contrastive to the speech. In other words, gesture can re-iterate or emphasise the speech, add information to the speech, or communicate something contradictory (or slightly different) from the associated speech. In contrast to speech, gesture has few con-

straints on how it is constructed. As McNeill (1992) notes "the important thing about gestures is that they are not fixed. They are free and reveal the idiosyncratic imagery of thought." McNeill (1992) identifies five categories of spontaneous gesture: Emphatic, Deictic, Cohesive, Metaphoric and Iconic.

Emphatic gesture (also known as beat gesture, or baton gesture) provides emphasis to parts of speech: phrases, words and phonemes. Emphatic gesture consists of just two movement phases (up/down, in/out, left/right) with the transition from one phase to the other (such as, up to down) being the point of emphasis. These have little variation in form other than the scale and speed of the phase transition, with larger, faster transitions indicating more emphasis (within an individual). Emphatic gesture can, and frequently does, utilise all body parts, especially the head and hands, but additional movement of more of the body provides further emphasis. This form of spontaneous gesture is distinct from other forms in that it can overlay any other gesture as it indexes a part of speech rather than providing semantic content (though it is also frequently used independently).

Deictic gestures are simply pointing actions that refer to an object or objects, generally using fingers, hands, or head. This is complicated by the fact that deictic gesture can, in addition to referencing concrete objects, reference more abstract objects, such as where an object was previously, the physical space referred to previously with the introduction of an idea or object, or almost any abstract space, including time. For example, when describing a cartoon involving two characters people will often reference one specific area of their gesture space for one character and another separate area for the other (McNeill, 1992). It should be noted that deictic gestures can take the form of whole body movement towards a space as well as the orientation of body parts.

Cohesive gestures serve to connect related parts of discourse that are temporally separated. For example, when listing items people often provide an emphatic gesture on each item. The emphatic gesture marks each item, while the repetition of the same gesture form connects them together to say 'here's one, and another, ...'. Cohesive gesture usually consists of the repetition of a specific gesture and so they require the use of other gesture forms (which could simply be emphatic gesture).

Metaphoric and iconic gestures are in essence animated representations. An iconic gesture is a pictorial animation of a concrete entity or action, while metaphoric gesture animates an abstract concept as if it were a physical object. For example, a speaker saying 'and he bends it way back' while illustrating the action of bending a stick is performing an iconic gesture (McNeill, 1992). In contrast metaphoric gesture occurs in situations such as when a speaker says 'I had this great idea' and marks the 'great idea' with a cupped hand gesture (i.e. metaphoric container). Metaphoric gesture takes abstract ideas and grounds them in real entities. In practice only a few types of object are portrayed, mostly containing or enclosed objects, though it is often not clear what object is being portrayed.

Iconic gestures are probably the most sophisticated and developed class of gesture; each gesture attempts to portray some aspect(s) of a situation, event, or object in the physical world and therefore absorbs much of the complexity of the physical world. Typically the gesture portrays the most important and semantically salient features in the context of the interaction, and therefore a gesture about a specific object, such as a teacup, can vary significantly depending on the context. For example, a person talking about a teacup in context of drinking tea may perform a gesture of lifting a teacup by its handle and 'drink', while if the context was about how much tea was in the cup, the gesture would be distinctly different – perhaps portraying the size of the teacup or the level of the tea as opposed to how it is lifted. Iconic gesture is the form of gesture that mostly allows for the addition of extra information in the gestural channel and almost anything is permitted.

For objects or ideas that are common in interaction, the gesture forms portraying them can often become stylised, and eventually can become symbols, akin to emblems.

In addition to gesture accompanying speech, gesture can be used in a similar way to eye gaze and many vocal signals to regulate communication flow and the rhythm of interaction (Knapp and Daly, 2002). Head nods are the most frequent form of these gestures, but hand and body gestures can also serve for flow regulation. Unsurprisingly, flow regulation gesture frequently coincides with flow regulation signals in other channels.

3.4 Self-Adaptors

Self-adaptors are movements that serve to change the self, such as scratching an arm; as such they are typically not intended to serve a communicative function – their purpose is to adjust something about the self. Certain reading as to the internal state of a speaker, based on their use of self-adapters, have been proposed for communication. Self-adaptors can take on many forms and are distinctly idiosyncratic. The meanings assigned to them by conversational participants and bystanders vary considerably. For example, flicking hair out of the face, while a practical movement, is often interpreted as a flirtatious behaviour.

3.5 Passive Communication

Finally, another consideration to be taken into account is passive communication – communication that occurs without a specific action on the part of the communicator, and usually takes place at the co-action level of communication, and at a greater distance than conversational communication. The communicator expresses information about themselves simply by their physical appearance, demeanour and observed behaviour. These factors can be considered by the user (or indeed by other non-player characters) before initiating a more direct method of communication. For example, a large aggressive character might be approached with more caution than a small timid looking character. Similarly, witnessing a character commit some violent act would instil more caution in the user than a character who has been doing something less impactful. Passive communication is often unintentional at the time of communication itself, but can be premeditated, for example wearing a certain type of clothing may convey a specific message.

4 Modelling Synthetic Agents

4.1 Behavioural Models

The objective of behavioural models is to isolate a particular behaviour or group of behaviours of the subject and create a simple mechanism to specifically simulate behaviour or behaviours. The goal is not to produce a full representation of the subject, restricting the simulation to the specific actions the chosen behaviours require.

Reynolds (1987) paper neatly outlines this philosophy and how such a system might be implemented. Reynold's aim was to simulate the flocking behaviour of birds (or the herding of animals or shoaling of fish which are visually similar behaviours). In order to accomplish this he took the approach of defining the behaviour itself as a series of rules that each participating 'boid' would follow. He did not for example attempt to simulate the sensory system of a bird or fish, or attempt to create a functioning 'brain' that reasons deliberatively.

Reynolds took the notion of a particle system (Reeves, 1983) and its physical rules (for example representing gravity) and augmented them with 'social rules', such as 'try to fly in the same direction as near neighbours' or 'try to fly towards the centre of the flock'.

Following the rule-based behaviour, the boids did eventually exhibit realistic behaviour, but it was not simply a result of defining a set of rules and allowing the boids to follow them. The interaction of each rule was a difficult factor to consider, especially if rules contradicted each other. Taking the average of rules was not enough, Reynold's gives the example: "Consider flying over a grid-work of city streets between the skyscrapers; while 'fly north' or 'fly east' might be good ideas, it would be a bad idea to combine them as 'fly north-east'." Instead, a rule hierarchy was created and weights attached to each rule. Each rule that was triggered had its weight added to a rule accumulator, and its effect (on acceleration) was added to another accumulator. If either accumulator reached a maximum value, the rest of the rules were ignored for that pass.

Behavioural modelling can produce a simple and elegant solution to portray a specific phenomenon, and its simplicity allows many synthetic characters to be simulated at once. However as it is focused on a specific behaviour, the synthetic characters are restricted, and do not fully encapsulate all the behaviours that may be expected from a rich synthetic character.

4.2 Cognitive Models

While behavioural modelling can produce realistic, if restricted, behaviour, cognitive modelling takes a more holistic approach to simulating a synthetic character. Funge et al. (1999) characterizes that a cognitive model is "What a synthetic character knows, how information is acquired and how the synthetic character uses it" – a model of memory (knowledge), senses (acquiring information) and ultimately the decision making process (how to use the information).

Funge describes a 'cognitive layer' controlling the behaviour of a synthetic character (and ultimately behaviour goes on to control physical which in turn determines kinetic forces which finally specifies geometric changes). The cognitive layer would make choices and decisions based on its memory and sensory input to choose which behaviour to exhibit. The modelling of such a process readily incorporates learning. The cognitive model can be trained through experience to make decisions that are not initially programmed into the character. This allows the modelling of expandable and adaptive synthetic characters that potentially have a rich set of behaviours that cover many situa-

tions. The computational cost of reasoning with such cognitive models is substantial. If there are many sensory inputs that need to be considered, or a large knowledge base to process, a character may require a complex and time consuming deliberative process. This overhead can limit the number of concurrent characters acting in a virtual environment.

When specifically considering modelling a human-like synthetic agent, the concept of a cognitive model can be extended beyond a deliberating 'brain' to incorporate models for human senses, memory and other human cognitive functions (Hartmann et al., 2002; Peters et al., 2006). The purpose of adding these elements is to recreate realistic socially enabled synthetic characters by emulating the internal processes that occur in humans. The agents produced are specifically embodied, multi-modal (in that they incorporate visual gesturing as well as verbal communication) conversational agents. They have sophisticated sensory inputs, for example vision is based on processing of actual renders of the virtual environment from the viewpoint of the character, with image processing techniques used to obtain areas that require the characters attention (Peters and O'Sullivan, 2002). Synthetic agents use this technique to perceive other characters' gaze during communication, pick up gestures and even read facial expressions.

Sensory, long term and short term memory are also simulated. The layering of memory allows data to be prioritised (by having it in short term memory), allows repeated data to not be re-stored (by referring to long term memory) and allow sensory data to be attended to (if in sensory memory) or ignored (if short term memory is full).

The synthetic agents also have a complex decision making process, or 'theory of mind'. This is where sensory and memory data is evaluated to determine what to attend to and which behaviours to exhibit. The communication between characters is handled by 'reading' visual and vocal cues and transmitting the same when indicating turn taking etc. The decision making process uses the sensory data to pick up cues and decide what to do next. To introduce individualism into the synthetic characters, reactions and animations are varied, which leads to different sensory data in the communication flow and hence different resulting reactions. The result is richer more interactive conversational synthetic characters that talk, gesture, gaze and respond realistically during conversation. However the processing overhead means that the system is currently limited to small groups of characters, and conversations with more than two characters look awkward and lethargic.

4.3 Conversational Non-Verbal Behaviour Synthesis

Conversational non-verbal behaviour in synthetic characters can be generated by analysis of the data stream being communicated (usually text based). Cassell et al. (2001) introduced a system that allowed animators to generate realistic non-verbal behaviours to accompany verbal communication by performing linguistic and contextual analysis of the text to be vocalised. The Input text is separated and tagged by the system, so that different animation fragments can be played to represent the non-verbal behaviour associated with pertinent sections of the text. As well as creating textual input, animators are required to create scripts that define the set of eye, face, head and hand behaviours and associate them with phrases.

Olivier et al. (2006) implements a similar system which utilises natural language processing to analyse text typed by a user conversing with an Embodied Conversational Agent (ECA). This requires the synthetic character to perform the non-verbal behaviours in real-time and in synchrony with the utterances.

4.4 Crowd Modelling

When modelling the behaviour of larger groups of synthetic characters such as crowds, a wide range of methods are used. However most have their basis in either behavioural modelling, with the characters obeying a set of rules, or cognitive modelling where each character is individually modelled and crowd behaviour is an emergent property, or more usually a hybrid between the two systems, where some group behaviour is governed by rules, but individuals have a decision making process as well. Often a behavioural model is seen as a 'top down' approach, where overall crowd behaviour is designed from the outset, and a cognitive model is seen as a 'bottom up' approach, where group behaviour is emergent from the individuals behaviour.

Rymill and Dodgson (2005) take a bottom up approach, where individual characters are modelled based on theories from psychology. Observed human behaviour is used to control collisions and character flow through an environment, incorporating collision avoidance (including avoiding oncoming collisions, overtaking and avoiding glancing collisions while travelling in similar directions). Collision avoidance techniques that do not require a character to change their path, such as the 'step and slide' observed in real human behaviour (Wolff, 1973) are also incorporated. Although individuals were modelled without an overall behavioural model, the observed results were very similar to those found in real crowd movements. For example individuals formed lanes through bottlenecks, and after collision avoidance, characters tend to return to original path rather than creating a whole new path similarly to how real humans behaved in the studies conducted by Daamen and Hoogendoorn (2003); Hoogendoorn and Daamen (2005).

Shao and Terzopoulos (2006) also take an individual approach to modelling crowds. Synthetic characters use a simple cognitive model: they have a series of attributes that are satisfied by different actions in the environment, for example 'hunger' might be satiated by finding a food source in the environment, at which point a preset animation would take place to represent eating, and the hunger attribute would be reset. These attributes decay over time and so need to be topped up by locating the appropriate object or individual, and having different decay rates allowed individuals to behave slightly differently. Sensory inputs allow collision avoidance with objects and other individuals. The model was used to populate recreated archaeological sites with realistically behaving crowds.

Musse and Thalmann (2001) created a control hierarchy for modelling crowd behaviour, a hybrid of behavioural and cognitive modelling. The hierarchy incorporated an overall crowd control, smaller subgroups and individuals. The overall group control was scripted depending on the situation, (examples include evacuation or reaction to user stimulus), but individuals had control over their own actions within the scripted event. Certain individuals were designated as leaders of subgroups and their actions would influence the overall goals and actions of other members of the group. This method produced good performance, with large sizes of crowds being possible, and the behaviour was suitably adaptable that it produced a realistic reaction to user stimulus.

5 Facilitating Interaction

The discussion of non-verbal behaviour above concentrates on the different forms of non-verbal behaviour in an attempt to establish requirements for next generation virtual characters. Mechanisms for the synthesis of such behaviours are inevitably beyond the scope of this analysis but there have been a number of recent attempts to synthesise spontaneous gesture (Kopp and Wachsmuth, 2004), (Olivier, 2004), (Cassell et al., 2001) and gaze behaviour. However, whilst we can imagine a situation whereby non-player characters have sophisticated cognitive models and the ability to both synthesise and interpret non-verbal behaviour, there is no mechanism for the player to communicate non-verbally with either the non-player characters or other players.

Present-day technology allows collection of full data on all aspects of human physical behaviour that may be a channel for non-verbal behaviour. This includes body position, body movement, hand shape, eye gaze direction, pupil dilation, facial expression, vocal behaviour, voice, and a variety of other biometrics. Unfortunately, such data can only be collected accurately using specialised (expensive) invasive equipment. With that commercial consideration, in addition to joystick (or controller), keyboard and mouse inputs it is only reasonable to expect relatively basic additional input devices for computer games in the near future. These devices include webcams and microphones, but also less standard input methods, such as dance mats, light guns, and low point motion capture devices, such as the Gametrack 3D motion tracker.

Within current commercial and technological constraints, three alternatives for the control of player character non-verbal behaviour generation can be identified:

1. Simulation – non-verbal behaviour of human players is simulated as for non-player characters, and is independent of the actual non-verbal behaviour of the human player.

2. Augmentation – as for simulation, but specific controls are given to allow a player to explicitly alter the simulation, such as a slider to indicate how happy the player is, or a button to increase the level of interest in an interaction.

3. Tracking and mapping – the human player is tracked using equipment such as webcams and the coarse features that can be identified are either mapped to explicit controls (as in the case of augmentation) or directly to the character animation.

The challenge for augmentation is to design an interface that is intuitive, non-obtrusive, and useful all at the same time. While the challenges in player non-verbal behaviour data collection are daunting, the challenges of understanding or recognising behaviours or meaning from that data are even more so. Gesture recognition, for example, is in its infancy and mostly addresses the use of gesture as an explicit interaction technique and little research has been conducted into the automated recognition or understanding of spontaneous gesture.

6 Conclusion

Synthetic characters in virtual environments, such as computer games, are currently under utilised as a vechicle for providing believable and immersive experiences. Human-like characters require behavioural fidelity across all task and behavioural contexts, and not only at key points in competition and narrative progression. Behavioural fidelity demands that synthetic characters express themselves to players and non-player characters alike, utilising aspects of non-verbal communication identified in the social psychology literature. Synthetic characters should use appropriate gesture behaviours to augment their verbal communication. Non-verbal behaviour helps aid the flow of conversation, highlights key points during communication and complements and supplements verbal communication through iconic and metaphoric gestures. Gaze is also an important non-verbal communication behaviour that synthetic characters should emulate. Like gesture, gaze regulates flow during communication, also indicating the state of the communicator, reflecting cognitive activity, expressing emotions or even showing the nature of a relationship between two individuals. At greater distances, gesture is also used to communicate intent.

Synthetic characters should behave appropriately depending on their task context. As well as an augmentation to conversation, non-verbal communication occurs between characters when they are competing against each other, when they are cooperating to accomplish some task and when coacting in the same shared social space. They should also display human like proxemic behaviour, taking into account their closeness, and arrangement in regard to, other characters in the shared space. There are physical limitations to communication over different distances or in certain environments, conversations are difficult at large distances, or in a loud place for example. As well as obeying the physical rules for communication, synthetic characters should behave in a socially correct manner – the interaction distance between characters depends on their familiarity and amiability. Proxemic Behaviour is based on the idea of territory, that is ownership of space. Synthetic characters should behave appropriately towards primary, secondary and public territories. The dynamic nature of public territories in particular require characters to obey different social rules depending on if the 'ownership' of space.

Modelling synthetic characters has been approached using several techniques, from behavioural rule-based systems for controlling large numbers of characters, to simulations of aspects of cognition, such as perception and memory. Models have not yet concentrated on the simulation of human-like communication. Modelling the verbal and non-verbal communication behaviours has its own complexities, and in particular requires the ability to perceive and reason about character-character interactions as a third party. Interacting with synthetic characters should also embody the notion of communication through verbal and non-verbal behaviours. Technology is sufficient to allow human users to interact with sufficiently enabled synthetic characters – gesture, spatial position, gaze direction and a variety of other biometrics can all be utilised a inputs to existing interaction technologies – requiring that the synthetic character would be able to recognise these interactions as non-verbal signals.

References

Altman, I. (1975). *The environment and social behaviour*. Brooks/Cole, Monterey, CA.

Cassell, J., Vilhjálmsson, H. H., and Bickmore, T. (2001). Beat: the behavior expression animation toolkit. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 477–486, New York, NY, USA. ACM Press.

Daamen, W. and Hoogendoorn, S. P. (2003). Experimental research on pedestrian walking behavior. *Transportation Research Board annual meeting*, pages 1–16.

Duncan, S. and Fiske, D. (1977). *Face-to-face interaction: research, methods, and theory*. L. Erlbaum Associates, Hillsdale, N.J.

Efran, J. (1968). Looking for approval: Effects on visual behaviour of approbation from person differing in importance. *Journal of Personality and Social Psychology*, 10:21–25.

Ehrlichman, H. and Weinberger, A. (1978). Lateral eye movements and hemispheric asymmetry: A critical review. *Psychological Bulletin*, 85:1080–1101.

Ekman, P. and Friesen, W. (1975). *Unmasking the face*. Prentice-Hall, Englewood Cliffs, NJ.

Ekman, P., Friesen, W., and Tomkins, S. (1971). Facial affect scoring technique: A first validity study. *Semiotica*, 3:337–58.

Funge, J., Tu, X., and Terzopoulos, D. (1999). Cognitive modeling: knowledge, reasoning and planning for intelligent characters. In *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pages 29–38, New York, NY, USA. ACM Press/Addison-Wesley Publishing Co.

Hall, E. (1966). *The hidden dimension*. Doubleday, Garden City, N.Y.

Hartmann, B., Mancini, M., and Pelachaud, C. (2002). Greta: Formational parameters and adaptive prototype instantiation for mpeg-4 compliant gesture synthesis. *Proceedings of Computer Animation, IEEE Computer Society*, pages 111–119.

Hearn, G. (1957). Leadership and the spatial factor in small groups. *Journal of Abnormal and Social Psychology*, 54:269–72.

Hoogendoorn, S. P. and Daamen, W. (2005). Pedestrian behaviour at bottlenecks. *Transportation Science*, 39(2):147 159.

Kendon, A. (1967). Some functions of gaze-direction in social interaction. *Acta Psychologica*, 26:2622–63.

Knapp, M. and Daly, J. (2002). *Handbook of interpersonal communication*. SAGE Publications, Thousand Oaks, CA.

Kopp, S. and Wachsmuth, I. (2004). Synthesizing multimodal utterances for conversational agents. *The Journal of Computer Animation and Virtual Worlds*, 15(1):39–52.

Maxis (2000). *The Sims*. Electronic Arts.

McNeill, D. (1992). *Hand and mind: what gestures reveal about thought*. University of Chicago Press, Chicago, IL.

McNeill, D. (2005). *Gesture and thought*. University of Chicago Press, Chicago, IL.

Mori, M. (1970). Bukimi no tani [the uncanny valley]. *Energy*, 7:33–35.

Musse, S. R. and Thalmann, D. (2001). Hierarchical model for real time simulation of virtual human crowds. *IEEE Transactions on Visualization and Computer Graphics*, 07(2):152–164.

Olivier, P. (2004). Gesture synthesis in a real-world eca. *Lecture Notes in Computer Science*, 3068:319–322.

Olivier, P., Jackson, D., and Wiggins, C. (2006). A real-world architecture for the synthesis of spontaneous gesture. *Proceedings of 19th annual conference on Computer Animation and Social Agents (CASA2006)*.

Peters, C. and O'Sullivan, C. (2002). Synthetic vision and memory for autonomous virtual humans. *Computer Graphics Forum*, 21(4):743752.

Peters, C., Pelachaud, C., Bevacqua, E., Ochs, M., Chafai, N. E., and Mancini, M. (2006). Social capabilities for autonomous virtual characters. *GAME 2006*, pages 37–48.

Reeves, B. and Nass, C. (1996). *The media equation: how people treat computers, television, and new media like real people and places*. Cambridge University Press, New York.

Reeves, W. T. (1983). Particle systems - a technique for modeling a class of fuzzy objects. *ACM Transactions on Graphics*, 2(2):359–376.

Reynolds, C. W. (1987). Flocks, herds and schools: A distributed behavioral model. *SIGGRAPH Comput. Graph.*, 21(4):25–34.

Rockstar (2002). *Grand Theft Auto 3*. Take 2, Edinburgh, Scotland.

Rymill, S. J. and Dodgson, N. A. (2005). Psychologically-based vision and attention for the simulation of human behaviour. In *GRAPHITE '05: Proceedings of the 3rd international conference on Computer graphics and interactive techniques in Australasia and South East Asia*, pages 229–236, New York, NY, USA. ACM Press.

Shao, W. and Terzopoulos, D. (2006). Populating reconstructed archaeological sites with autonomous virtual humans. *Proc. 6th International Conference on Intelligent Virtual Agents (IVA 06)*, 4133:420–433.

Valve (2004). *Half-Life 2*. Take 2, Bellevue, Washington.

Weisz, J. and Adam, G. (1993). Hemispheric preference and lateral eye movements evoked by bilateral visual stimuli. *Neuropsychologia*, 31(12):1299–1306.

Wilbur, M. and Roberts-Wilbur, J. (1985). Lateral eye-movement responses to visual stimuli. *Perceptual and motor skills*, 61:167–177.

Wolff, M. (1973). *Notes On The Behaviour Of Pedestrians, In: Peoples In Places: The Sociology Of The Familiar*, pages 35–48. Praeger, New York.

Using MUDs as an experimental platform for testing a decision making system for self-motivated autonomous agents

María Malfaz and Miguel A. Salichs

RoboticsLab, Carlos III University of Madrid
28911, Leganés, Madrid, Spain
mmalfaz@ing.uc3m.es ; salichs@ing.uc3m.es

Abstract

In this paper a decision making system for autonomous and social agents who live in a virtual world is presented. This world was built using a text based multi-user game: a MUD (Multi User Domain). In this world the agents can interact with one other, allowing social interaction, as well as interaction with the other objects present in the world. In this paper, the usefulness of using this kind of text based multi-user games as test beds for designing decision making systems of artificial agents, is proved.

The proposed decision making system is composed of several subsystems: a motivational system, a drives system and an evaluation and behaviour selection system. The selection of behaviours is learned by the agent using reinforcement learning algorithms. The dominant motivation is considered as the inner state of the agent. In order to simplify the learning process, the states related to the objects are considered as independent from one another. The state of the agent is a combination between his inner state and his state in relation with the rest of agents and objects. This system uses happiness and sadness, defined as positive and negative variations of the wellbeing of the agent, as the reinforcement function.

1 Introduction

The final goal of the work presented in this paper is to design a decision making system for an autonomous and social agent with no *a priori* knowledge. This means that the agent is the one who decides its own actions, and it interacts with other agents. One important feature of the agent is that, using reinforcement learning, it learns the right behaviours to execute through its own experience. This decision making system could be implemented on virtual agents as well as on real robots. In fact, this research was originally oriented to the design of a decision making system for autonomous robots. However, before implementation of this system in a real robot, we used MUDs as an experimental platform for testing this decision making system.

The agent lives in a virtual world where objects, necessary for survival, and other agents exist. This agent must learn a policy of behaviour to survive, maintaining all his needs inside acceptable ranges. The policies establish a normative about what to do in each situation. This means that the agent must learn the proper relation between states and actions. In this system the agent knows the properties of every object, i.e. the agent knows which actions can be executed with each object. What the agent does not know is which action is appropriate in each situation. In order to carry out this learning process, the agent uses reinforcement learning algorithms. In order to create this virtual world a text based game, available on the net and called CoffeMud, gave us the perfect tool to carry out our objective.

Emotions, in general, are used for showing the emotional expression of the agents, as a way of communication among users and for making the agent more believable. Nevertheless, emotions have a fundamental role in human behaviour and social interaction. They also influence cognitive processes, particularly problem solving and decision making (Damasio, 1994). Emotions can also act as control and learning mechanisms (Fong et al., 2002). In this work, emotions are used to attempt to imitate their natural function in learning processes and decision making.

The remainder of the paper is organized as follows. In section 2 the concept of autonomy, and its meaning from several points of view, is introduced. This autonomy implies the introduction of new concepts: motivations and drives. Both concepts are explained in this section. Next, in section 3 the decision making system proposed in this work is presented, later the state of the agent is defined, as well as the reinforcement function used in the learning process. Section 4 presents the experimental procedure used in this work. First, the environment, the virtual world where the agents live, is presented and the experimental settings of the agent are described. Finally in this section, the indicators of performance of the agent are introduced. In section 5 and section 6 the experimental results, when the agent lives alone in the world and when he shares the environment with others, are presented and discussed. Finally, the main conclusions of this paper are summarized in section 7.

2 Autonomy

In order for agents to be truly autonomous, not only must they be capable of intelligent action, but they must also be self-sustained (Arkin, 1988). In other words, autonomy implies a decision making process and this requires some knowledge about the current state of the agent and environment, including his objectives (Bellman, 2003).

According to Cañamero, autonomous agents are natural or artificial systems in constant interaction with dynamic and unpredictable environments, with limited resources.

In general, these agents are sociable and they must satisfy a set of possible conflictive goals in order to survive (Cañamero, 2003).

From this same point of view, Gadanho defines an autonomous agent as an agent with goals and motivations. This agent has also some way to evaluate behaviours in terms of environment and his own motivations. The motivations are desires or preferences that can lead to the generation and adoption of objectives. The objectives are situations that must be reached. These final objectives of the autonomous agent, or motivations, must be oriented to maintaining the internal equilibrium of the agent (Gadanho, 1999).

In games, the autonomy of the agents that the user can find while playing is an essential issue for giving them a life-like appearance. For this reason the decision making system presented in this paper is designed for giving the agent the ability to select his own actions. In this work it will be considered that an autonomous agent is the one that is self-motivated, and decides which behaviours to select in order to maintain the internal equilibrium of the agent.

2.1 Homeostasis, drives and motivations

Homeostasis means maintaining a stable internal state (Berridge, 2004). This internal state can be parameterized by several variables, which must be around an ideal level. When the value of these variables differs from the ideal one, an error signal occurs: drive. These drives constitute urges to action based on bodily needs related to self-sufficiency and survival (Cañamero, 1997).

One of the oldest theories about drives was proposed by Hull in 1943. Hull suggested that privation induces an aversion state in the organism, which was termed drive. According to his theory, drive increases the general excitation level of an animal. Drives were considered as properties of deficit states which motivate behaviour (Hull, 1943).

The word motivation derives from the Latin word *motus* and indicates the dynamic root of behaviour, which means those internal, rather than external factors, that urge the organism to action (Santa-Cruz et al., 1989).

There are several motivational theories that attempt to explain the human and animal behaviour. Nevertheless, there is not a unique classification of those theories. In this section some of those theories will be presented.

2.1.1 Homeostatic theories of motivation

According to these theories, human behaviour is oriented to the maintenance of the internal equilibrium. Among several homeostatic theories, one of them will be selected: The drive reduction theory.

The drive reduction theory

Many drive theories of motivation between 1930 and 1970 posited that drive reduction is the chief mechanism of reward. If motivation is due to drive, then, the reduction of deficit signals should satisfy this drive and essentially could be the goal of the entire motivation (Berridge, 2004).

Hull proposes the idea that motivation is determined by two factors. The first factor is drive. The second one is the incentive, that is the presence of an external stimuli that predicts the future reduction of the need. For example, the presentation of food constitutes an incentive for an hungry animal (Hull, 1943).

2.1.2 Incentive motivation theory

Incentive motivation concepts rose as drive concepts decrease, beginning in the 1960s. Several studies with animals suggested that motivation is more compatible with incentive concepts of taste reward than with earlier drive reduction concepts. It was proposed that individuals were motivated by incentive expectancies, not by drives or drive reduction (Berridge, 2004).

Nevertheless, clearly, a physiological drive state is important for motivation, even if drive is not equivalent to motivation. One does not seek out food when one is thirsty. Physiological deficits such as hunger or thirst depletion signals do modulate motivation for rewards such as food. To incorporate physiological drive/deficit states into incentive motivation, Toates suggested that physiological depletion states could enhance the incentive value of their goal stimuli. This was essentially a multiplicative interaction between physiological deficit and external stimulus, which determined the stimulus' incentive value (Toates, 1986).

3 Decision making system

In this section the decision making system proposed in this work is presented. This system has been developed based on motivation, drive, emotion and learning concepts. These concepts are essential for research in human and animal behaviour.

The objective of this work is to obtain a completely autonomous agent and therefore, an agent with the capacity of making his own decisions. This decision making process has to be learnt through his own experience: his successes and failures. During his experience, using reinforcement learning algorithms, the agent learns the right policy of behaviour in order to survive.

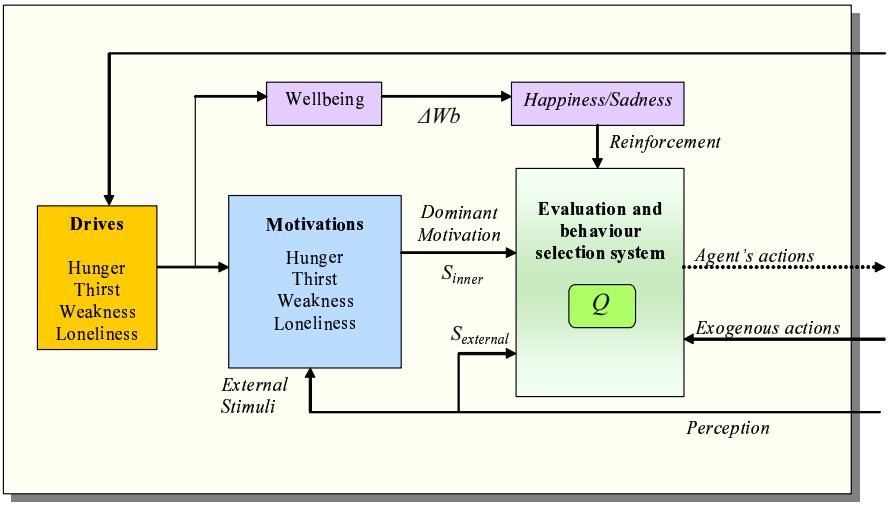

Figure 1: The proposed decision making system

The proposed decision making system is composed of several subsystems: a motivational system, a drives system and an evaluation and behaviour selection system, see Figure 1. The reinforcement function is happiness or sadness that, as it will be shown

later, are related to the variation of the wellbeing of the agent. How this system works is described next.

When the drives are satisfied their value is zero and as needs increase, the values of the drives increase in each simulation step, with each of them following a certain dynamics. These values are introduced, together with the external stimuli values, in the motivational system. In this system the intensity of the motivations related to each drive are calculated. The motivation with the highest intensity is the dominant motivation. This dominant motivation determines the inner state of the agent. This inner state and the external state define the state of the agent.

The evaluation and behaviour selection system chooses the behaviours according to a certain selection policy. The evaluation of behaviours, when the agent is in a certain state, is done using reinforcement learning algorithms, more specifically Q-leaning. Therefore, the agent learns which action to select in a particular state. The reinforcement used to evaluate the result of the execution of an action are the emotions happiness and sadness. These emotions are defined based on the variation of the wellbeing experimented by the agent: ΔWb. Wellbeing is a function of the needs of the agent. Therefore, this reinforcement measures the effect of the selected action on the needs of the agent. As will be described later in this paper, the positive and negative variations of wellbeing are directly related to happiness and sadness respectively. The agent will use these emotions to evaluate his own actions and to learn which of them are most suitable for each state.

3.1 Wellbeing

The wellbeing of the agent is defined as the degree of needs satisfaction. Therefore, when all the drives of the agent are satisfied, their values are zero and the wellbeing is maximum.

As is shown in equation (1), the wellbeing of the agent is a function of its drives values, D_i, and some personality factors, α_i. These personality factors weigh the importance of each drive in the wellbeing of the agent.

$$Wb = Wb_{ideal} - \sum_i \alpha_i \cdot D_i \tag{1}$$

Wb_{ideal} is the ideal value of the wellbeing of the agent. As the values of the drives of the agent increase as time goes on, or due to the effect of any other action, the wellbeing of the agent decreases. Depending on the values of the personality factors, the increase of the drives can affect, to a certain extent, in the wellbeing of the agent. Every time that a drive reduction exists, there is an increase in the wellbeing.

The wellbeing of the agent is calculated at every simulation step, as well as its variation ($\triangle Wb$). This wellbeing variation is calculated as the current value of the wellbeing minus the value in the previous step, as it is shown in the next equation:

$$\Delta Wb^{k+1} = Wb^{k+1} - Wb^{k} \tag{2}$$

The biggest positive variation of the wellbeing will be produced when the drive related to the dominant motivation is satisfied.

3.2 Reinforcement learning

The agent that uses reinforcement learning tries to learns, through interaction with the environment, how to behave in order to fulfil a certain goal. The agent and the environment are continuously interacting, the agent selecting actions and the environment

responding to those actions and presenting new situations to the agent. The environment and the proper agent also give rise to rewards that the agent tries to maximize over time. This type of learning allows the agent to adapt to the environment through the development of a policy. This policy determines the most suitable action in each state in order to maximize the reinforcement. The goal of the agent is to maximize the total amount of reward he receives over the long run (Sutton and Barto, 1998).

Reinforcement learning has been successfully implemented in several virtual agents and robots (Isbell et al., 2001), (Martinson et al., 2002), (Bakker et al., 2003), (Ribeiro et al., 2002), (Bonarini et al., 2006), (Thomaz and Breazeal, 2006).

3.3 Q-learning

The goal of reinforcement learning is to learn a mapping from states and actions to a measure of the long term value of taking that action in that state, known as the optimal value function (Smart and Kaelbling, 2002). The Q-learning optimal value function is defined as:

$$Q^*(s,a) = E\left[R(s,a) + \gamma \max_{a'} Q^*(s',a')\right] \tag{3}$$

This represents the expected value of the rewards received by the agent for taking action a from state s, leading to the new state s', and then acting optimally from there. The parameter γ ($0 < \gamma < 1$) is known as the discount factor, and is a measure of how much attention the agent pays to possible rewards that the agent might get in the future. In other words, it defines how much expected future rewards affect decisions now (Humphrys, 1997).

The Q-function is frequently stored in a table, indexed by state and action. Starting with arbitrary values, one can iteratively approximate the optimal Q-function based on the observations of the world. Every table entry $Q(s,a)$ is then updated according to (Smart and Kaelbling, 2002):

$$Q(s,a) = (1-\alpha)\cdot Q(s,a) + \alpha \cdot (r + \gamma V(s')) \tag{4}$$

Where:

$$V(s') = \max_{a \in A}(Q(s',a)) \tag{5}$$

is the value of the state s' and is the best reward the agent expect from state s'. A is the set of actions, a is every action, s' is the new state, r is the reinforcement, γ is the discount factor and α is the learning rate.

In other words, the Q value is the expected rewards for executing action a in state s and then following the optimal policy from there. The goal of the Q-learning algorithm is to estimate the Q values.

The learning rate α ($0 < \alpha < 1$) controls how much weight is given to the reward just experienced, as opposed to the old Q estimate (Humphrys, 1997).

3.4 Behaviour selection

At the beginning of each experiment the initial values of all Q values are equal to zero. The agent, through his experience in the world, will explore all the possible actions and will update those values. Random exploration takes too long to focus on the best actions,

so instead a method will be used that interleaves exploration and exploitation of the best learnt policy.

The specific control policy used is a standard one already implemented, obtaining good results, in (Watkins, 1989). The agent tries out actions probabilistically based on their Q values using a Boltzmann distribution. Given a state s, it tries out action a with probability:

$$P_s(a) = \frac{e^{\frac{Q(s,a)}{T}}}{\sum_{b \in A} e^{\frac{Q(s,b)}{T}}} \tag{6}$$

Temperature T controls the amount of exploration, i.e. the probability of executing actions other than the one with the highest Q value. If T is high, or if Q values are all the same, this will pick a random action. If T is low and the Q values are different, it will tend to pick the action with the highest Q value.

In order to select the value of T that, as has been shown, will determine the randomness in the action selection, several experiments were carried out. Those experiments showed that this T value is dependent on the Q values. Therefore, in this work it is proposed that, in order to maintain a fixed randomness, T must be defined as a function of the average value of the Q values:

$$T = \delta * average\ value\ of\ Q \tag{7}$$

In the experiments the parameter δ must be tuned in order to determine the exploration/exploitation level. As is shown in (7), a high value of δ will favor the exploration of all the possible actions. On the contrary, a low value of δ will favor the exploitation of the most suitable actions.

3.5 State of the agent

As has been stated for the decision making process, it is necessary to know the state of the agent. In this system the state is the agent is the combination of his inner state, S_{inner}, and his external state, $S_{external}$.

$$S = S_{inner} \times S_{external} \tag{8}$$

Next, both states, inner and external, will be defined.

3.5.1 Inner state

The inner state depends on the motivations that are related to the needs of the agent i.e. the drives. In this system other factors, that may affect the human inner state such as psychological factors, will not be considered.

Motivational states represent tendencies to behave in particular ways as a consequence of internal (drives) and external (incentive stimuli) factors (Ávila García and Cañamero, 2004). In other words, the motivational state is a tendency to correct the error, the drive, through the execution of behaviours.

In order to model the motivations of the agent we used the Lorentz's hydraulic model of motivation as an inspiration (Lorenz and Leyhausen, 1973). Lorenz's hydraulic model is essentially a metaphor that suggests that motivational drive grows internally and operates a bit like pressure from a fluid reservoir which grows until it bursts through an outlet.

Motivational stimuli in the external world (food, water, sexual and social stimuli, etc.) act to open an outflow valve, releasing drive to be expressed in behavior. In Lorenz's model, internal drive strength interacts with external stimulus strength. If drive is low, then, a strong stimulus is needed to trigger motivated behaviour. If the drive is high, then a mild stimulus is sufficient (Berridge, 2004).

Have been also introduced activation levels (L_d) for motivations. Therefore the intensity of the motivations, whose related drive is higher than this level is calculated, following the idea of the Lorenz's hydraulic model, as the sum of the intensity of the related drive (D_i) and the related external stimuli (w_i). In other case, the intensity of the related motivation is zero, as is reflected in the following equation:

$$\begin{array}{l} \text{If } D_i < L_d \text{ then } M_i = 0 \\ \text{If } D_i \geq L_d \text{ then } M_i = D_i + w_i \end{array} \tag{9}$$

The external or incentive stimuli are the different objects that the player can find in the virtual world. These incentive stimuli are the same used by Cañamero (Cañamero, 1997). Therefore, certain behaviours, consummatory ones, can only be executed when these stimuli are present. According to (9), the intensity of a motivation can be high due to two reasons:

1. The value of the correspondent drive is high.

2. The related motivational stimulus is present.

This model can explain the fact that due to the availability of food in front of us, we sometimes eat although we are not hungry.

In this decision making system, as is proposed in (Balkenius, 1993) and (Balkenius, 1995), once all the intensities of the motivations are calculated, these compete one another. The motivation with the highest intensity is the dominant motivation and it is the one that determines the inner state, as shown in equation (10). It can happen that none of the drives of the agent has a value higher than that limit. In that case, there is not any dominant motivation and it can be considered that the agent has no needs, he is "OK".

$$S_{inner} = \begin{cases} \arg\max_i M_i \rightarrow & If\ \max_i M_i \neq 0 \\ OK \rightarrow & In\ other\ case \end{cases} \tag{10}$$

3.5.2 External state

The external state is the state of the agent in relation to all the objects, passive and active, that the agent can interact with:

$$S_{external} = S_{obj_1} \times S_{obj_2} ... \tag{11}$$

Since this definition implies a huge number of states, in this system is considered that the states related to the objects are independent from one another. This means that the agent, in each moment, considers that his state in relation to the food is independent from his state in relation to water, medicine, etc. This simplification reduces the number of states that must be considered during the learning process of the agent.

Without this simplification the number of states in relation to all the objects would be huge. For example, if there were 10 objects present in the world and it was assumed that for each object there exist 3 logical variables: having the object, being next to the object

and knowing where the object is, we would have $2^3 = 8$ states related to every object. If the external state of the agent is his relation to all the objects, $8^{10} = 1.073.741.824$ states will exist, as was previously stated, a huge number of states. Nevertheless, using the simplification, it is considered that the external state is the state of the agent in relation to each object separately, therefore for the 10 objects present in the world we would obtain $10 \times 8 = 80$ states, which is a great reduction in the number of states.

3.6 Modification of Q-learning

The simplification made on the states in relation to the objects causes, for example, the agent to learn, when he is hungry, what to do with the food ($s \epsilon S_{hunger} \times S_{food}$) without considering his relation to the rest of objects. Therefore the total state of the agent in relation to each object is defined as follows:

$$s \epsilon S_{inner} \times S_{obj_i} \tag{12}$$

This definition implies that the value of the actions executed in relation to a certain object are independent of his relation with the rest of objects present in his environment. This is not really true, if for example, the agent is beside the object water and executes the action "go for food", and at the end of this action the agent is next to food. Therefore, the agent is no longer beside the object water, so his state in relation to water has changed although the action executed was related to food.

Therefore, in order to take into account these "collateral effects", a modification of the Q-learning algorithm is proposed:

$$Q^{obj_i}(s,a) = (1-\alpha) \cdot Q^{obj_i}(s,a) + \alpha \cdot \left(r + \gamma \cdot V^{obj_i}(s')\right) \tag{13}$$

Where:

$$V^{obj_i}(s') = \max_{a \in A_{obj_i}} \left(Q^{obj_i}(s',a)\right) + \sum_m \Delta Q^{obj_m}_{\max} \tag{14}$$

is the value of the object i in the new state considering the possible effects of the executed action with the object i, on the rest of objects. For this reason, the sum of the variations of the values of every other object is added to the value of the object i in the new state, previously defined in equation (5).

These increments are calculated as follows:

$$\Delta Q^{obj_m}_{\max} = \max_{a \in A_{obj_m}} \left(Q^{obj_m}(s',a)\right) - \max_{a \in A_{obj_m}} \left(Q^{obj_m}(s,a)\right) \tag{15}$$

Each of these increments measures, for every object, the difference between the best the agent can do in the new state, and the best the agent could do in the previous state.

3.7 Reinforcement function: Happiness and sadness

Considering the definition of emotion given by Ortony (Ortony et al., 1988), it is considered that the emotion occurs due to an appraised reaction (positive or negative) to events. According to this point of view, in (Ortony, 2003), Ortony proposes that happiness occurs because something good happened to the agent. On the contrary, sadness appears when something bad happened. In our system, this can be translated into the fact that

happiness and sadness are related to the positive and negative variations of the wellbeing of the agent:

$$\begin{array}{l} \text{if } \Delta Wb > L_h \Rightarrow \text{Happiness} \\ \text{if } \Delta Wb < L_s \Rightarrow \text{Sadness} \end{array} \quad (16)$$

Where ΔWb is the variation of the wellbeing, defined in equation (2), and $L_h \geq 0$ and $L_s \leq 0$ are the minimum variations of the wellbeing of the agent that cause happiness or sadness.

Rolls (Rolls, 2003) proposes that emotions are states elicited by reinforcements (rewards or punishments), so our actions are oriented to obtaining rewards and to avoiding punishments. Following this point of view, in this proposed decision making system, happiness and sadness will be used as the positive and negative reinforcement function respectively, during the learning process.

The use of happiness and sadness, as the reinforcement function in the learning process, is also related to the drive reduction theory. This relationship is based on the definitions of happiness and sadness as the positive and negative variations of wellbeing, respectively. The positive variations, according to (1) and (2), are related to the reduction of drives, while the negative variations are related to their increase.

4 Experimental procedure

As has been already said, the final goal of this work is the design of a decision making system for autonomous and social agents. This system has been tested using a virtual agent who lives in a virtual world where some objects and other agents exist.

4.1 Description of the virtual environment

Bellman in (Bellman, 2003) proposes the use of virtual worlds as test beds for experiments with artificial agents. One of the most important things that is needed is an environment in which one can explore very difficult mappings between goals, agent capabilities, agent behaviours, and interaction with the environment, and consequences or results in that environment. Using virtual worlds this can be reached, but the disadvantages of course, are that these worlds are not nearly as rich as real worlds.

Virtual worlds rose from three mayor lines of development and experience: (1) Role-playing, multi-used Internet games called MUVEs (multi-user virtual environments); (2) Virtual reality environment and advanced distributed simulation, especially those used in military training exercises; and (3) Distributed computing environments, including Internet.

The use of these virtual worlds as experimental platforms is being extended among the robotic and artificial intelligence community. For example, in (Isbell et al., 2001) the research on reinforcement learning of an artificial agent that lives in a multi-user environment called LambdaMOO, is presented. This environment is one of the oldest text-based multi-user role playing games, and it is formed by interconnected rooms, with users and objects that can move from one room to another. The social interaction mechanisms are designed to reinforce the illusion that the user is present in the virtual space. Another example of the use of computer games as experimental platforms is the work presented in (Thomaz and Breazeal, 2006). In this work the players interactively train a virtual robot to do a task. As in LabdaMOO, it is an external player who gives the reinforcement to the agent during the learning process.

4.2 The virtual world: Coffeemud

MUD stands for "Multi-User Dungeon", originally developed in 1979, and refers to a text-based multi-user game based on the fantasy adventure genre such as Dungeons and Dragons. The choice of this text-based game instead of using a modern with 3D graphics one arises from the need to simulate a robot with sensors and actuators living in a real world. We wanted a virtual world with a very easy way to send and acquire information. Using this text based game, for our virtual agent, acquiring information is equivalent to reading a text and acting (move, take, etc) is equivalent to sending a text. For our purpose, a MUD offered the perfect way to create a virtual environment containing all the necessary objects (food, water, medicine) to interact with.

Among quite a lot of different MUD codebases, the java-based CoffeeMud (Zimmerman, 2007) has been chosen due to its available documentations and clear explanations.

In a typical MUD, a person would connect to a MUD Server using a Telnet client, and play. Since we want our agents to play, we have created several programs, in C language, that connect to our mud server through the Telnet interface, simulating different players. These agents will behave according to the proposed decision making system.

In order to set up our experiments, we decided to create the area *Passage*. *Passage* was designed in a way similar to the System and Automation Engineering Department plant of the Carlos III University. This means that is formed by a long corridor with rooms situated on both sides since one of the future applications is implementing this decision making system in our robot which will be moving around that scenario.

4.3 Agents at the *Passage* area

This area is formed by 20 rooms, 8 of them forming a corridor and the rest of the rooms are offices distributed at both sides of the corridor. In this area, as has been previously stated, the player can find different objects. These objects can be passive, which are not capable of executing actions, or active, which can execute actions.

The objects that are present in this world are the following:

- Food (passive)
- Water (passive)
- Medicine (passive)
- World (passive)
- Another Agent (active)

Except for the agents, which are moving around autonomously, the rest of the objects are distributed in rooms in such a way that there is a room with food, another with medicine and another with water. The amount of objects present in those rooms are huge, and therefore, it is considered that the agent has unlimited resources. The agent at the beginning of the game does not know where to find those objects. Throughout his time life, the agent finds the objects and remembers their position so if the agent needs some object, he will know where to find it.

There are no doors in this area and the way the agent moves in the world is giving commands of direction: north, south, east and west. With one movement command, the agent passes from one room to another. The commands used for interacting with the passive and active objects will be described later. It is worthy of mention that there are two

movement behaviours: "explore" and "go to" which use two kinds of mathematical algorithms. In the case of the “explore” behaviour is the DFS (Depth First Search) algorithm, which gives a route to explore all the rooms of the area. For going to a certain room, the Dijkstra algorithm solves the shortest path problem between the current and the final rooms.

4.4 Graphic interface

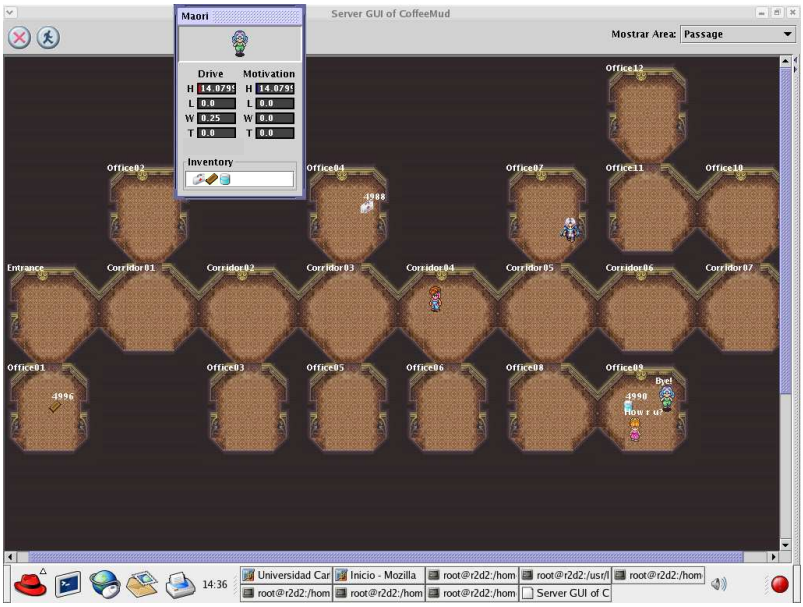

Figure 2: The graphic interface

Figure 3: The agent’s sub-window

Due to the nature of a MUD, as we have stated, the interaction between a player and the game is text based. Although it is quite easy to detect all the objects in the area, it is quite difficult to have a global view of the game since one can only "see" the room where one is placed. Furthermore, when there are several players connected at the same time, the only way to watch the player's evolution is by using a graphic interface developed by the authors for this purpose, see Figure 2.

Using this interface all the player's actions can be followed, as well as their drives and motivations values. Moreover, all the items that the player is carrying are showed on the little sub-windows developed for each player as can be observed in Figure 3.

4.5 Agent's description

4.5.1 *Drives*

The considered drives and motivations are the following:

- Hunger
- Thirst
- Weakness
- Loneliness

These drives have been selected taking into account the needs the agents in the virtual world. A typical player who lives in CoffeMud needs to eat and drink in order to survive. The Weakness and Loneliness drives have been added to make the working frame more complete. Since our final goal is to develop a decision making system for an autonomous and social agent, the need for social interaction is included as one of the agent's needs.

The values of the Hunger and Thirst drives increment a certain amount at every step simulation. These drives do not grow at the same rate. Physiological studies determine that in most human beings the necessity of water, thirst, appears before the need for food, hunger. In (Gautier and Boeree, 2005) it is presented how Maslow discovered that certain needs prevail over others. For example, if one is hungry and thirsty, one will tend to relieve thirst before hunger. After all, one can survive several days without food, but one can only live a couple of days without water. As a conclusion, thirst is a stronger need that hunger.

Some drives, or needs of the agent, after being satisfied do not start to increase their values immediately, but after a certain time, which we term "satisfaction time". This happens in the same way that when after eating, one is not hungry again until some hours later.

In this system, some of the drives of the agent follow this pattern, previously described. Thus, some drives have these satisfaction times whose values depend on the urgency of each of them. These drives are Hunger, Thirst and Loneliness. The Weakness drive follows a different pattern, as will be described later. In the next equation, the satisfaction times corresponding to these drives are shown:

$$\begin{aligned} T_{thirst} &= 50\ steps \\ T_{hunger} &= 100\ steps \\ T_{loneliness} &= 150\ steps \end{aligned} \tag{17}$$

According to these values, the Thirst drive is the most urgent one, since it takes less time to increase its value again. In general, one is thirsty more frequently than one is

hungry. The social need, the Loneliness drive, takes much more time in increasing its value again since it is not a very urgent need. Once the satisfaction time passes the drives grow as follows:

$$\begin{aligned} D_{thirst}^{k+1} &= D_{thirst}^{k} + 0.1 \\ D_{hunger}^{k+1} &= D_{hunger}^{k} + 0.08 \\ D_{loneliness}^{k+1} &= D_{loneliness}^{k} + 0.06 \end{aligned} \tag{18}$$

As is shown, the growing rate of the Thirst drive is higher than that of the Hunger drive, and this drive in turn increases its value faster than the Loneliness drive.

The variation of the Weakness drive depends on the movement of the agent. Therefore, if the agent is still this drive does not suffer any variation, but if the agent moves the value of the drive increases at every step, as is shown next:

$$D_{weakness}^{k+1} = D_{weakness}^{k} + 0.05 \tag{19}$$

Moreover, while the agent is interacting with another agent some drives can be affected by the actions executed by the other agent. In fact, when the agent is robbed:

$$D_{loneliness}^{k+1} = D_{loneliness}^{k} + 1 \tag{20}$$

and, when the agent is kicked:

$$\begin{aligned} D_{loneliness}^{k+1} &= D_{loneliness}^{k} + 1 \\ D_{weakness}^{k+1} &= D_{weakness}^{k} + 3 \end{aligned} \tag{21}$$

4.5.2 Motivations of the agent

According to the equation (9), motivations are defined as the sum of the value of the drives and the external stimuli. These external or motivational stimuli, w_i, are the different objects that the agent can find in the world during the game, so:

$$\begin{aligned} &\text{If the stimuli is present then } w_i \neq 0 \\ &\text{If the stimuli is not present then } w_i = 0 \end{aligned} \tag{22}$$

Table 1 shows the motivations, drives and their related motivational stimuli.

Table 1: Motivations, Drives and motivational stimuli

Motivation/Drive	Motivational stimuli
Hunger	Food
Thirst	Water
Weakness	Medicine
Loneliness	Another agent

Equation (9) shows the application of the activation levels L_d in order to calculate the value of the intensity of motivations. In the experiments:

$$L_d = 2 \tag{23}$$

4.5.3 Wellbeing

In relation to the wellbeing of the agent, as has been shown, said wellbeing is a function of the drives. Therefore, adapting equation (1) to the agent's design:

$$Wb = Wb_{ideal} - (\alpha_1 D_{hunger} + \alpha_2 D_{thirst} + \alpha_3 D_{weakness} + \alpha_4 D_{loneliness}) \tag{24}$$

Where: $Wb_{ideal} = 100$.

The personality factors, α_i, weigh the importance of each drive on the wellbeing of the agent. In the experiments all the drives will have the same importance, therefore, all the personality factors are equal to one other:

$$\alpha_i = 1 \tag{25}$$

By varying these personality factors we could design different kinds of agents. For example, if we increase the value of α_4, the personality factor related to the Loneliness drive, the lack of social interaction will imply a big decrease of the wellbeing of the agent (sadness). Therefore, since the reinforcement function is related to the variation of the wellbeing (emotions), this agent will be very sociable.

4.5.4 State of the agent

According to section 3.5.1, in this scenario the inner state of the agent is defined as follows:

$$S_{inner} = \{Hungry,\ Thirsty,\ Weak,\ Alone,\ OK\} \tag{26}$$

In relation to the external state, the state related to every passive object, except for the object world, are the combination of three binary variables:

$$S_{obj} = Being\_in\_posession\_of \times Being\_next\_to \times Knowing\_where\_to\_find \tag{27}$$

In relation to the object world, at the moment, the state of the agent in relation to the world is unique, the agent is always in the world:

$$S_{world} = Being\_at \qquad (Always\ True) \tag{28}$$

Finally, in relation to another agent:

$$S_{agent} = Being\_next\_to \tag{29}$$

Every variable is evaluated as $= \{true, false\}$.

Therefore, according to the definition of the state given by the equation (3.6), the agent could be, for example, in the following state in relation to food: " hungry, not having food, not being next to food and knowing where to find food".

4.5.5 Actions of the agent

The sets of actions that the agent can execute, depending on his state in relation to the objects, are the following:

$$A_{food} = \{Eat,\ Get,\ Go\ to\} \tag{30}$$

$$A_{water} = \{Drink\ water,\ Get,\ Go\ to\} \tag{31}$$

$$A_{medicine} = \{Drink\ medicine,\ Get,\ Go\ to\} \tag{32}$$

$$A_{another\ agent} = \begin{cases} Steal\ food/water/medicine \\ Give\ food/water/medicine \\ Greet \\ To\ do\ nothing \\ Kick \end{cases} \tag{33}$$

$$A_{world} = \{Keep\ still,\ Explore\} \tag{34}$$

Among all these behaviours there are some of which cause an increase or decrease of some drives, as is shown in table 2, leading to a variation of the wellbeing of the agent:

Table 2: Effects of the actions over drives

Action	Drive	Effect
Eat	Hunger	Reduce to zero (drive satisfaction)
Drink water	Thirst	Reduce to zero (drive satisfaction)
Drink medicine	Weakness	Reduce to zero (drive satisfaction)
To be greeted	Loneliness	Reduce to zero (drive satisfaction)
To be stolen	Loneliness	Increase certain amount
To be given	Loneliness	Reduce to zero (drive satisfaction)
To be kicked	Loneliness	Increase certain amount
To be kicked	Weakness	Increase certain amount
Explore/ go to	Weakness	Increase certain amount

The “do nothing” action has no effect on the drives of the agent.

4.6 Indicators of performance of the agent

Other authors, (Ávila García and Cañamero, 2002), defined some viability indicators to compare the performance of several agents (robots) that use different decision making architectures. Taking those indicators as a reference, in this work two different indicators are defined for the analysis of the obtained results. For this reason it has to be taken into account that during the experiments carried out, the agents do not die. The agents have a fixed time life. The performance of the agent is determined by the analysis of the wellbeing of the agent, since this information gives an accurate idea as to how well the experiment went.

First, an important concept needs to be introduced: Security Zone. The Security Zone is defined as an interval of wellbeing values, in such a way that it can be considered that if the wellbeing of the agent is inside this interval, then the agent is "doing good". The interval of the Security Zone is defined as $SZ = [100, 92]$

In order to analyze the performance of the agent for every experiment, two indicators have been defined:

1. The average value of wellbeing: This indicator gives a general idea about the performance of the agent, but it does not gives a clear idea about quality of life. This average value of wellbeing can be high due to a good general performance, as well as due to very good and very bad moments.

2. Percentage of permanence inside the Security Zone: This indicator gives a more obvious idea about the quality of life of the agent during the experiment. What is important for the experiment is not only that the agent has a good average value of wellbeing, but also that the agent has a good life.

5 Experimental results: Solitary Agent

In this section, the behaviour of an agent living on his own in the previously described world, is presented. Moreover, we have decided to do the learning parameters adjustment in this environment. These parameters are the following:

- The parameter δ that defines the relation between exploration and exploitation of the actions.

- The learning rate α which controls how much weight is given to the reward just experienced.

- The discounted factor γ that defines how much expected future rewards affect decisions now.

In order to learn a good policy of behaviour all the available actions in every state have to be executed. Therefore, the agent decides which action is the most suitable for every state. In each experience, the agent updates the Q value of every state-action pair. Finally, the most suitable actions for every state will have a high Q value. In order to guarantee that all the actions are explored, the parameter δ must be high. This parameter has already been introduced in section 3.4 and determines the randomness when selecting an action. When its value is high, all the actions have the same probability of being selected, with no preference among them. Since all the possible actions are executed, it causes some of them not to be the most suitable ones. As a consequence, the wellbeing of the agent decreases.

On the other hand, the learning rate α was introduced in equation (4), where the updating of the Q values of the reinforcement learning was defined. The effect of this parameter is to give more or less importance to the learnt values than to the new experiences. A high value of this parameter causes very sudden changes in the learnt Q values. On the other hand, a very low value of this parameter causes the learning process to be slow. This is because the agent is very conservative and he gives little importance to the new experiences. The best option would be an intermediate value of this learning rate. Based on several experiments it was proved that an intermediate value guarantees a good relation between the variability of the Q values and the importance of the new experiences.

There is another parameter related to the Q-learning process (13), which defines how much expected future rewards affect present decisions. This is the parameter γ, called the discount factor. As was explained in section 3.3, a high value of this parameter gives more importance to future rewards. A low value, on the contrary, gives much more importance to current reward. The updating of the Q value, for every state-action pair, is formed by two contributions: the reward received in that moment (r) and the importance of the best that can happen from the new state ($\gamma \cdot V^{obj_i}(s')$).

The experiments showed that in order to get the agent to learn a proper sequence of actions, the discount factor γ must be high. In Figure 4 the Q values of the actions related to the food when Hunger is the dominant motivation, and the value of the discount factor γ is high, are shown.

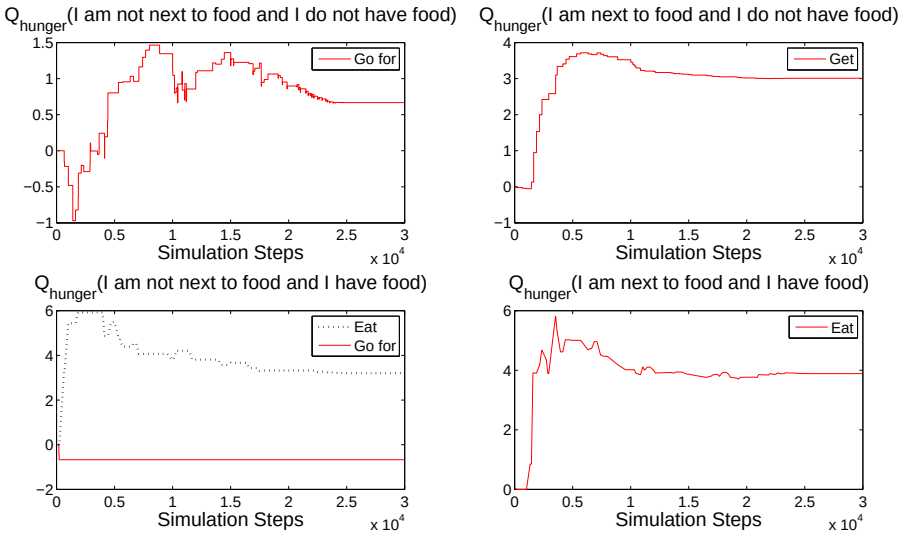

Figure 4: Q values of the actions related to food when the agent is hungry, with $\gamma = 0.8$

When the agent is hungry, next to food, has food and then eats, the Q value is high due to the received reward (see the graph on the bottom right corner of Figure 4). The next time the agent is beside food and gets the food, the Q value of this action is updated as has been previously explained. The new state is "to have food", and the best thing that the agent can do is to eat it, therefore the value of this new state is high. As a consequence, although the agent did not receive an immediate reward for getting the food, the fact that the value of the new state is multiplied by a high value of γ will cause the Q value of "get food" to be high (see the graph on the top right corner of Figure 4). The same will happen when the agent executes the action "go for food" and the new state will be " to be next to food" since its value, as has just been shown, is high. As a conclusion, when the agent uses a high value of γ, he learns the sequence of behaviours that leads him to satisfy the Hunger drive correctly.

Each experiment consists of two phases: the *learning phase* and the *steady phase*. During the learning phase, the agent starts with all the initial Q values equal to zero. The agent, through his experience in the world, learns and updates his Q values. Once the learning phase has finished, the steady phase starts. In this last phase the agent "lives" according to the learnt Q values.

During the learning phase the values of the parameters δ, that determines the exploration level, and the learning rate α decrease gradually. In the steady phase, the agent exploits the learnt policy of behaviour and stops learning. Therefore, in this steady phase the actions executed are the ones whose Q values are the highest and this implies that δ is very low. Since the agent stops learning, the learning rate is equal to zero $\alpha = 0$.

In Figure 5, the wellbeing of the agent along both phases is shown. As can be observed, during the learning phase the wellbeing of the agent increases gradually, being higher than 95 at the end of this phase. In the steady phase wellbeing maintains its high value, in fact the average value is 98.51 and the percentage of permanence in the Security Zone is 100%. Therefore, it can be considered that the agent learned a good policy of behaviour.

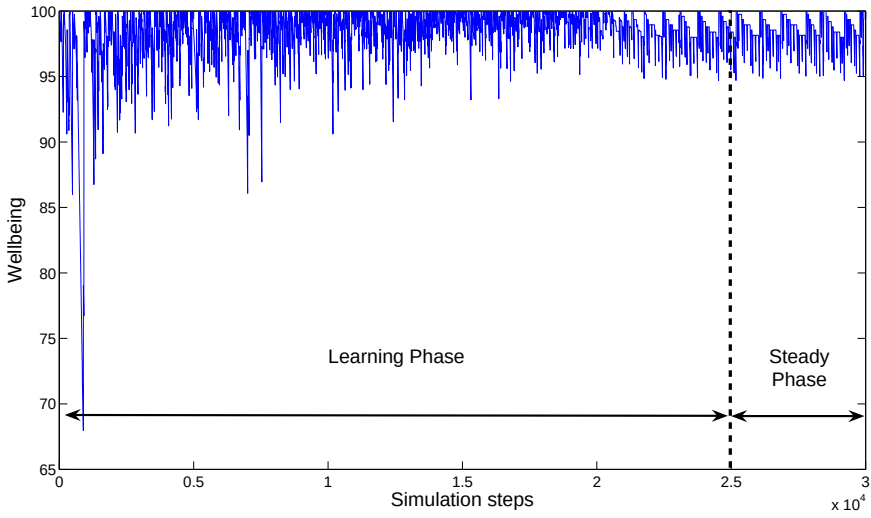

Figure 5: Wellbeing of the agent during the learning and steady phases

In the table 3, the most suitable values of all the parameters involved in the agent's design are shown:

Table 3: Parameters of the agent

	Learning Phase	*Steady Phase*
δ	$1.8 \rightarrow 0.1$	0.1
α	$0.3 \rightarrow 0$	0
γ	0.8	0.8

6 Experimental results: Accompanied Agent

In the case that the agent lives with another agents, the Loneliness drive appears in order to cause social interaction. In the experiments the agent has to live with different kinds of opponents who have fixed policies of behaviour: one of them is a good one, another a bad one and finally, the neutral one. Depending on the kinds of opponents that are living with the agent, different types of worlds are described: the good world, the bad world, the neutral world and the mixed world.

When social interaction exists, the rewards received by the agent depend not only on his own actions but on the action executed by the other agent. Therefore, several multi-agent reinforcement learning algorithms were tested in every world, such as the Friend or Foe algorithm. This algorithm was developed for general-sum games (Littman, 2001). Nevertheless, it was proved that those algorithms do not give any significant advantages in comparison with the Q-learning algorithm. This is because the multi-agent learning algorithms are developed based on game theory, and this implies that both agents are adaptable, which means that both of them are learning. This is not the case for the proposed environments where the opponents have fixed policies of behaviour and therefore, it seems to make sense that the best results are obtained when the agent uses the new Q-learning algorithm proposed in this work.

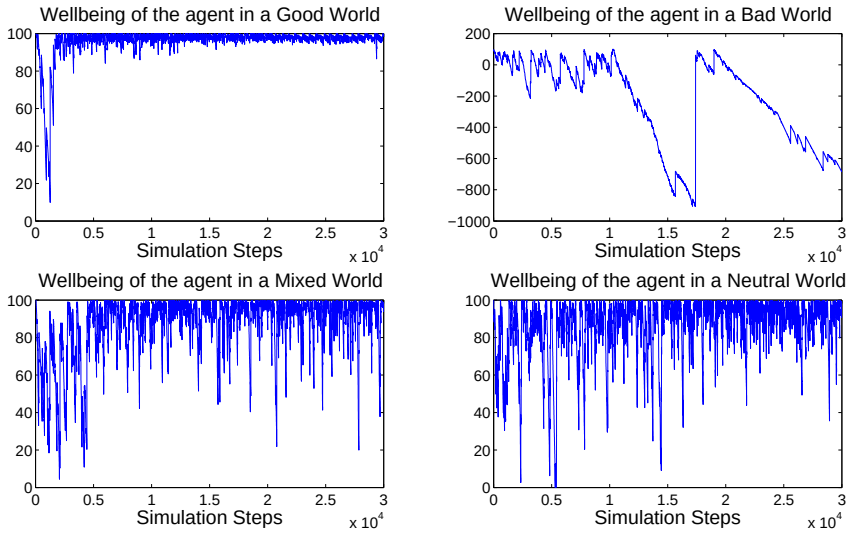

Figure 6: The wellbeing of the agent in every world when using the new Q-learning

In Figure 6, the wellbeing of the agent during the learning and the steady phase when the agent uses during the social interaction the new Q-learning algorithm, is shown for every different world. As can be seen, when the agent lives in a good world, i.e. all his opponents are good, wellbeing is almost near the ideal value during half of the learning phase and during the entire steady phase. In fact, the values of the indicators of performance of the agent, during the steady phase, in this world are very high as is shown in table 4.

On the contrary, when the agent shares the environment with three bad opponents, this is a bad world, then as is shown, the wellbeing of the agent is quite negative at the

end of the steady phase. Therefore, in this world the indicators of the performance of the agent are not calculated. The wellbeing is so negative because the opponents kick him or to steal objects and therefore, the agent will not be able to satisfy his Loneliness drive that, as a consequence, increases its value indefinitely. This means that the agent learns to avoid the social interaction in a bad world, since most of those interactions have very negative results.

In the case that the agent lives in a neutral or mixed world, the results obtained seem to be similar to one other. In these worlds the results of social interaction can be positive or negative and therefore, when the agent tries to satisfy his Loneliness drives sometimes he is able to do so but not always. This is the reason why in Figure 6 the wellbeing of the agent in these worlds has so many dips along both phases. In spite of the existence of all those drops, the indicators, as can be observed in table 4, are quite acceptable.

Table 4: Indicators of performance

World	*Average Value*	*% of permanence inside SZ*
Good	99.2	100
Neutral	90.68	58.7
Mixed	91.81	71
Bad	NO	NO

7 Conclusions

As was shown in the introduction, the final objective of this work is to design a decision making system, using unsupervised learning, for an autonomous and social agent. In order to carry out this objective, this decision making system was implemented in a virtual agent who lives in a MUD called CoffeeMud.

The agent lives in the "Passage" area where he can find different environments. In this paper the performance of the agent living in different kinds of environment has been presented. Looking at these results, we tested the developed decision making system for the agent. The experiments can be separated in two main parts: First, the agent living alone in "Passage", this means with no any other agent, and secondly, the agent living accompanied by another agents with different personalities.

In the first set of experiments it has been proved that, in order to learn a good policy, the agent has to first explore all possible actions. In relation to the learning rate, it was proved that the agent learns correctly with an intermediate-low value. In order to take advantage of the knowledge acquired, we decided to separate the life of the agent into two phases: the learning phase and the steady phase.

During the learning phase the exploration level and the learning rate decrease gradually. This implies that the agent, at the beginning of this phase, explores all the actions and learns from his experience. As the agent lives, he starts to exploit the actions that led to good results, as well as to give less importance to new experiences, i.e. the agent

begins to be more conservative. During the steady phase the agent stops learning and lives according to the policy learned. Moreover, it has also been proved that, in order to learn a correct policy, the discounted factor must be high. It is necessary, when evaluating an action taken in a certain state, to consider future rewards.

As a conclusion, when the agent lives alone in the MUD he is able to learn an appropriate policy of behaviour by himself. The agent uses a modification of the Q-learning algorithm to learn the correct function between states and actions. Using the variation of the wellbeing of the agent, happiness and sadness, as the reinforcement function, the agent learns to survive by maintaining all his drives in acceptable ranges. Therefore, emotions are used not for external communication of the agent but for controlling the learning process.

When the agent lives with other agents he uses the values of the parameters previously tuned. In relation to the need for social interaction, a new drive was implemented: Loneliness. This drive was implemented so that the agent satisfied it by interacting with another agent; therefore, the agent needs to interact with others in order to survive.

It has been proved that the agent is also able to learn appropriate policies of behaviour when he shares his environment with other agents. The best learning algorithm to deal with the social interaction is the new algorithm based on Q-learning. Using this algorithm, the agent was able to survive in a complex world maintaining his drives with low values.

As the main conclusion of this paper, we proved the usefulness of using a MUD for developing and testing a decision making system. This text based game gave us the possibility of creating simple as well as complex environments in a very direct way. As the experimental results showed, the proposed decision making produce very natural results giving the agent a life-like appearance, which is very useful for the design of games.

Acknowledgements

The authors gratefully acknowledge the funds provided by the Spanish Government through the projects called “Peer to Peer Robot-Human Interaction” (R2H), of MEC (Ministry of Science and Education) and the project “A new approach to social robotics” (AROS), of MICINN (Ministry of Science and Innovation).

References

Arkin, R. C. (1988). Homeostatic control for a mobile robot: Dynamic replanning in hazardous environments. In *SPIE Conference on Mobile Robots, Cambrige, MAA*.

Bakker, B., Zhumatiy, V., Gruener, G., and Schmidhuber, J. (2003). A robot that reinforcement-learns to identify and memorize important previous observations. In *the 2003 IEEE/RSJ International Conference on Intelligent Robots and Systems, IROS2003*.

Balkenius, C. (1993). Motivation and attention in an autonomous agent. In *Workshop on Architectures Underlying Motvation and Emotion WAUME 93, University of Birmingham*.

Balkenius, C. (1995). *Natural Intelligence in Artificial Creatures*. PhD thesis, Lund University Cognitive Studies 37.

Bellman, K. L. (2003). *Emotions in Humans and Artifacts*, chapter Emotions: Meaningful mappings between the individual and its world. MIT Press.

Berridge, K. C. (2004). Motivation concepts in behav-ioural neuroscience. *Physiology and Behaviour*, (81):179–209.

Bonarini, A., Lazaric, A., Restelli, M., and Vitali, P. (2006). Self-development framework for reinforcement learning agents. In *the 5th International Conference on Developmental Learning (ICDL)*.

Cañamero, L. (1997). Modeling motivations and emotions as a basis for intelligent behavior. In *First International Symposium on Autonomous Agents (Agents'97), 148-155. New York, NY: The ACM Press.*

Cañamero, L. (2003). *Emotions in Humans and Artifacts*, chapter Designing emotions for activity selection in autonomous agents. MIT Press.

Damasio, A. (1994). *Descartes' Error - Emotion, reason and human brain*. Picador, London.

Fong, T., Nourbakhsh, I., and Dautenhahn, K. (2002). A survey of socially interactive robots: Concepts, design, and applications. Technical report, CMU-RI-TR-02-29.

Gadanho, S. (1999). *Reinforcement Learning in Autonomous Robots: An Empirical Investigation of the Role of Emotions*. PhD thesis, UNiversity of Edinburgh.

Gautier, R. and Boeree, G. (2005). *Teorías de la Personalidad: una selección de los mejores autores del S. XX*.

Hull, C. L. (1943). *Principles of Behavior*. New York: Appleton Century Crofts.

Humphrys, M. (1997). *Action Selection methods using Reinforcement Learning*. PhD thesis, Trinity Hall, Cambridge.

Isbell, C., Shelton, C. R., Kearns, M., Singh, S., and Stone, P. (2001). A social reinforcement learning agent. In *the fifth international conference on Autonomous agents, Montreal, Quebec, Canada*.

Littman, M. (2001). Friend-or-foe q-learning in general-sum games. In *Proceedings of the Eighteenth International Conference on Machine Learn-ing*, pages 322–328.

Lorenz, K. and Leyhausen, P. (1973). *Motivation of human and animal behaviour; an ethological view*, volume xix. New York: Van Nostrand-Reinhold.

Martinson, E., Stoytchev, A., and Arkin, R. (2002). Robot behavioral selection using q-learning. In *of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), EPFL, Switzerland*.

Ortony, A. (2003). *Emotions in Humans and Artifacts*, chapter On making Believable Emotional Agents Believable, pages 188–211. MIT Press.

Ortony, A., Clore, G. L., and Collins, A. (1988). *The Cognitive Structure of Emotions*. Cambridge University Press. Cambridge, UK.

Ribeiro, C. H. C., Pegoraro, R., and RealiCosta, A. H. (2002). Experience generalization for concurrent reinforcement learners: the minimax-qs algorithm. In *AAMAS 2002*.

Rolls, E. (2003). *Emotions in Humans and Artifacts*, chapter Theory of emotion, its functions, and its adaptive value. MIT Press.

Santa-Cruz, J., Tobal, J. M., Vindel, A. C., and Fernández, E. G. (1989). Introducción a la psicología. Facultad de Psicología. Universidad Complutense de Madrid.

Smart, W. D. and Kaelbling, L. P. (2002). Effective reinforcement learning for mobile robots. In *International Conference on Robotics and Automation (ICRA2002)*.

Sutton, R. S. and Barto, A. G. (1998). *Reinforcement Learning: An Introduction*. MIT Press, Cambridge, MA, A Bradford Book.

Thomaz, A. L. and Breazeal, C. (2006). Transparency and socially guided machine learning. In *the 5th International Conference on Developmental Learning (ICDL)*.

Toates, F. (1986). *Motivational systems*. Cambridge (MA): Cambridge Univ. Press.

Ávila García, O. and Cañamero, L. (2002). A comparison of behavior selection architectures using viability indicators. In *Proc. International Workshop Biologically-Inspired Robotics: The Legacy of W. Grey Walter(WGW'02)*.

Ávila García, O. and Cañamero, L. (2004). Using hormonal feedback to modulate action selection in a competitive scenario. In *Proc. 8th Intl.Conference on Simulation of Adaptive Behavior (SAB'04)*.

Watkins, C. J. (1989). *Models of Delayed Reinforcement Learning*. PhD thesis, Cambridge University, Cambridge, UK.

Zimmerman, B. (2007). http://www.coffeemud.org/.

Relationships and Interactions of Multiple Characters for Interactive Narrative

M. Gillies*, I. B. Crabtree† and D. Ballin†

* Department of Computing, Goldsmiths, University of London, New Cross, London SE14 6NW, UK, *m.gillies@gold.ac.uk*
† BT plc., Adastral Park, Ipswich IP5 3RE, UK, *barry.crabtree@bt.com ; daniel.ballin@bt.com*

Abstract

This paper presents a system for generating non-verbal communication behaviour suitable for characters in interactive narrative. It handles interactions of groups of different characters, each with different relationships to eachother. It is possible to customise the behaviour of individual character using a system of character profiles. This allows characters to have a strong individuality and personality. These same profiles also allow the characters' behaviour to be altered in different contexts, allowing for suitably changing behaviour as the story unfolds.

1 Introduction

When we look at the behaviour of a group of people having a conversation in a public place we see a fascinating variety of behaviour, a complex interplay of eye gaze, lively gesturing and a variety of postures. We can often tell a lot about the group merely by watching them without hearing the conversation. It is normally simple to tell who is talking from their gestures and from how much others look at them. It is also often interesting to guess relationships between people from their relative postures. Posture can also give us a clue to the emotional tone of the conversation, for example how straight some one stands can show how confindent they feel. This is particularly true of multiparty interactions in which different people have different personalities and mannerisms, as well as different relationships to eachother.

Diversity of characters is vital to creating stories, the interplay and conflicts between contrasting personalities is one of the most important elements composing narrative. The differences between characters must be clearly visible in their behaviour, and non-verbal behaviour is one of the most important expressions of personality. As such, it must be possible to customise characters, giving each their own specific mannerisms and behaviour. However, characters should not always act in the same way. A key element of narrative is that as the story unfolds the characters' situation changes and in particular the emotional tone of the story alters. This means that the behaviour of the characters should be able to change to express their new situations. This can happen in many ways. For example, characters should behave differently depending on who they are interacting with, and the relationship between them. Characters can also take on different goals, and different behaviour is appropriate in different places and situtations. One of the key elements of narrative is interpersonal relationship. As well as their own personalities, mannerisms and goals, characters should have unique relationships to other characters. These relationships should be easy to understand from their behaviour.

This paper presents work that aims at improving the capacity of multi-user narrative worlds for expressive behaviour, and in particular to capture those aspects specific to multi-party conversations (rather than individual emotional expression, or two person conversation). This paper concentrates on two aspects of expressive behaviour that are particularly relevant to multi-party conversations: relationships between participants and the regulation of flow of conversation. The behavioural displays of relationships and attitude between members of multi-party groups is far more complex than in pairs. Pairs only contain one relationship, between the two participants, whereas in a group each member relates simultaneously to every other and their body language simultaneously reflects their attitude to several individuals and to the group as a whole. Body language also becomes very important in regulating the flow of conversation, the more people involved the more complex turn-taking becomes. These two features are likely to add to the aesthetic appeal of virtual worlds and make them more enjoyable, but they also serve important functions in improving the quality of conversation over the internet. One problem that is often noted with internet communication is the lack of affective communication, messages will often be misunderstood as there is no information about the emotional tone of the conversation. A message might seem hostile when read on an instant messaging system, whereas in face-to-face conversation the accompanying body language would make it clear that it is meant in a friendly way. Supplying animated characters with expressive body language should help solve this problem. The flow of conversation can also be problematic in instant messaging and similar systems, the latency of typing messages often means that messages are received out of order resulting in confusion or at least a rather irritating style of conversation. Replicating some of the non-verbal cues that help people manage face to face conversations in virtual worlds will help solve some of these problems and thus will significantly aid communication in virtual worlds. The work presented here has been implemented in the context of dealing with this problem by augmenting textual chat interfaces with animated characters endowed with appropriate non-verbal behaviour. Section 5.1.1 discusses this in more detail.

The work presented here aims at extending existing research on non-verbal communication and social interaction for autonomous characters and avatars in order to apply it to social interaction in groups rather than pairs. Work aimed solely at simulating social interaction between pairs of people will not capture some of the key complexities of group interaction. One key features of group interaction that we focus on here is that it features multiple social relationships. Each person in an interaction has a relationship with each other person, which must be taken into account, as well as their relationship to the group as a whole. It is therefore critical to simulate how these multiple relationships affect the behaviour of a character. This paper presents methods for handling these multiple relationships in a behavioural model for non-verbal communication. With each character having a set of unique profiles that specify both their individual personality and their relationships to others. Attention and gaze also become critical in group social interactions. When interacting with a group we must divide our attention between the members of that group. How we do this depends on many factors including the flow of conversation and our relationships with the others. In this work we therefore take social attention and gaze as central to the simulation of social interaction.

This paper presents a system for generating expressive behaviour for small groups of animated characters. We focus on displaying attitudes of characters to each other and on flow of conversation. To display these features we use the modalities of gaze, attention, postures and, to a degree, gestures. We have developed a framework, called Demeanour, for the generation of non-verbal communication in avatars . It is able to generate non-verbal behaviour in real-time based on a user definable behaviour model (currently we are working with psychologically-based models as described in section 3). Avatars react appropriately to each other's body language, making it

unnecessary for the user to explicitly react to the actions of others. The aspects of the Demeanour framework that deal with interactions between characters are described in more detail in Gillies and Ballin (2004). After discussing related work, two sections will describe some of the psychological models behind our work and the behaviour generating architecture we use. We then describe how the architecture can handle interactions between different characters. Finally we discuss our system of profiles that can represent the individuality of characters and their relationships.

2 Related Work

Our work builds on a body of research on autonomous characters for virtual environments, for example, Blumberg and Galyean (1995); Badler et al. (1993); Tu and Terzopoulos (1994); Perlin and Goldberg (1996), and Rickel and Johnson (1999). There have been a number of general models of non-verbal communication. These include, Guye-Vuilléme et al. (1999), who have demonstrated avatars with a wide range of controllable expressive behaviour, however, they do not consider how the behaviour of one character should interact with that of another. APML (Affective Presentation Markup Language) is an XML based language for defining the expressive behaviour of characters which is associated with many projects on expressive characters e.g. DeCarolis et al. (2004) and Bevacqua et al. (2004), but it again does not take into account interaction between characters, merely behaviour of one character. Cassell *et al.*'s various systems particularly their virtual real estate agent, Rea (Cassell et al. (1999)), but they deal only with interaction between pairs of characters, or between one real person and one character. Our work aims to extend such work to groups of character. Jan and Traum (2005) simulate turn taking in multi-party conversation, however, they do not consider relationships between people in a group and have a more limited simulation of non-verbal communication and attention.

Work on simulating eye gaze for social situations includes Garau et al. (2001) and Colburn et al. (2000) simulate the patterns of eye gaze between pairs of characters based on frequencies of mutual gaze. Vilhjálmsson and Cassell (1998) use eye gaze to help regulate the flow of conversation including turn taking behaviour. Rickel and Johnson (1999), in their character based virtual reality tutoring system, use gaze primarily as a method of indicating to the user an area of interest in the environment. Thórisson (1998) simulates eye gaze in the context of more general work on multi-modal communicative behaviour during conversation. **?**) simulates the role of attention in the potential initiation of social interaction. Our gaze model uses many aspects of the work of these authors, however it aims at extending them to multi-party conversation. There has been less work on eye groups in groups of more than two, though Vertegaal et al. (2000) have produced a simple model for a single agent interacting with multiple human users.

Among research on posture, Cassell et al. (2001a) have investigated shifts of postures and their relationship to speech, but not the meaning of the postures themselves. As such their work is complimentary to ours. Coulson (2002) uses an OCC model of emotion to generate postures. Bécheiraz and Thalmann (1996) use a one-dimensional model of attitude, analogous to our affiliation, to animate the postures of characters in pairs of interacting characters.

The generation of gestures has been studied by a number of researchers. For example, Cassell et al. (1999) have produced a character capable of extensive non-verbal behaviour including sophisticated gestures. Chi et al. (2000) present a way of generating expressive movements, similar to gestures using Laban notation. Gestures are closely related to speech and should be closely synchronised with it. Cassell et al. (2001b) present a system that parse text and suggests appropriate gestures to accompany it. Much of this work relates to a a detailed analysis of verbal communication,

which is very closely linked to gesture. Such analysis is out of the scope of work, however, so we use a much simplified model of gesture relative to these authors.

Maya et al. (2004) have investigated how to create variation between animated characters. They use XML based profiles which are merged with an XML based specification of the affective content of a particular piece of speech, using an XSLT based system, to produce a final piece of behaviour. However, they do not provide any user friendly system for customising characters, nor does their system work in real time. The use of profiles and context dependence has also been used in other types of agent technology, for example, Soltysiak and Crabtree (1998).

3 Expressive behaviour

As described in the introduction we focus on two aspects of expressive behaviour. The first (described in more detail in Gillies and Ballin (2003)) is the way in which non-verbal behaviour expresses relationships between people, or more exactly the attitude of one person to another. The second is how non-verbal behaviour interacts with the flow of conversation. Before we give an overview of the models we use to generate this behaviour, it is important to briefly describe the methodology we employ to design our models. Following authors such as Cassell et al. (1999) or Bécheiraz and Thalmann (1996), we use a psychologically based methodology, our behaviour generation systems are based on theories developed in social psychology and related disciplines. The advantage of this method is that it gives a principled foundation for our algorithms. However, psychological theories can be incomplete and, more importantly, by their nature they tend to focus on general explanation of the behaviour and can miss out specific details that are vital in implementing an animation algorithm. We therefore complement our study of psychology with personal observation of human behaviour.

We have based our model of interpersonal attitude on the work of Argyle (1975) and Mehrabian (1972). Though there is an enormous variety in the way in which people can relate to each other Argyle identifies two fundamental dimensions that can account for a majority of non-verbal behaviour. One dimension is affiliation, the degree to which one person likes or dislikes another. The other is dominance/submissiveness, which represent some sort of power or status relationship between individuals. Attitude and its expression can depend both on the general disposition of the person and their relationship to the other person, for example status depends on whether they are generally confident and whether they feel superior to the person they are with. This issue is complicated in multiparty interactions, as the attitude to the different member of the groups can be different and the resulting behaviour is a combination of the individual attitudes and the attitude to the group as a whole.

Non-verbal behaviour also interacts closely with the flow of conversation. Non-verbal behaviour is used extensively to regulate conversation dealing with aspects such as turn-taking (determining who should speak at a given time). Gaze is a particularly important modality in this respect, Argyle and Cook (1976) have done extensive studies with pairs of individuals to understand levels of eye gaze, and mutual gaze, in conversations. They have produced some useful results concerning the level to which individuals will look at the other while speaking (on average 35%) and listening (75%).We have used these results to influence our model of gaze and mutual gaze in group settings. Expressive behaviour is also related to other aspects of conversation, for example, gestures only ever occur during speech (whether talking or listening). Non-verbal communication also provides a back channel, feedback to the speaker from the listener, which can encourage the speaker or show disagreement, the most common back channel gesture in western culture is the head nod (and the head shake for disagreement).

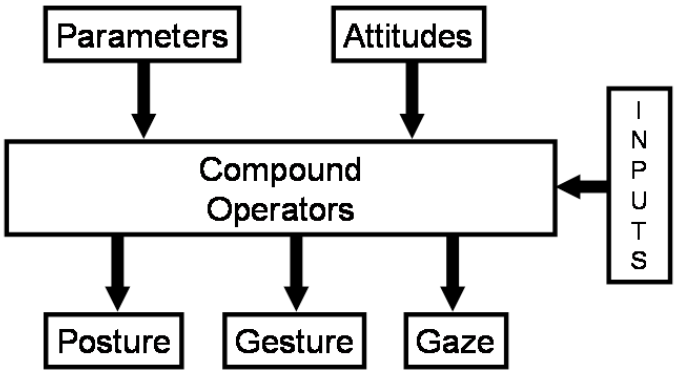

Figure 1: An overview of the Demeanour framework

Expressive behaviour is displayed in a number of ways, or modalities. In our characters we use postures (which is closely related to attitude), gaze, and more generally social attention, (which is connected both to attitude and flow of conversation) and, to a more limited degree, gesture (which is related to flow of conversation). Section 5 describes how these various modalities relate to attitude and flow of conversation, and how the behaviour is generated.

4 The Demeanour Architecture

Demeanour is a framework for the generation of non-verbal communication in avatars (figure 1 shows an overview of the system). It is able to generate non-verbal behaviour in real-time based on a user definable behaviour model (currently we are working with psychologically based models). Avatars react appropriately to each other's body language, making it unnecessary for the user to explicitly react to the actions of others. Non-verbal communication expresses itself in a number of ways, or modalities we are currently interested in three: posture, gesture and social attention. The Demeanour framework is described in more detail in Gillies and Ballin (2004).

The structure of Demeanour is shown in figure 1. In demeanour each character has a behavioural controller that uses a number of factors to generate behaviour, *inputs* from other characters, internal *parameters* that control characters' "personality" and internal *attitudes* to other characters (how a character feels about another character). Inputs are read directly from values in the behavioural controllers of other characters. Parameters are stored internally in a character and attitudes are similar to parameters, but can have different values for different characters (as described below). These factors are used to generate behaviour of a number of modalities (in this case, posture, gesture and gaze). Demeanour uses a number of *compound operators* to generate the behaviour, these operators are used combine inputs, parameters and attitudes in a number of ways. As they can also act on the output of other operators, they can achieve a multi-stage mapping. The output values of certain operators are then used

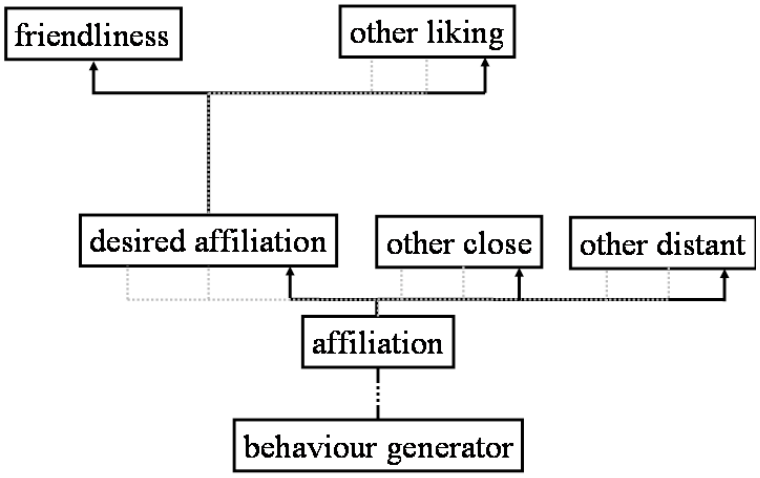

Figure 2: An example inputs, parameters, attitudes and operators for affiliation

as input to the animation modules that control posture, gesture and gaze, as described below.

5 Multi-character interaction

In interactions between multiple characters the relationships between these characters becomes critical. Importantly, a character can no longer be considered to have a single affective state, but a different attitude to each member of the group, with the resulting behaviour being a combination of all these attitudes. Body language can reflect an attitude to the group as a whole or may be focused on one or two members. This section describes how the Demeanour framework handles interactions between groups of multiple characters. First, we discuss features of the Demeanour framework itself that enable the interaction of multiple characters will be discussed. Section 5.1 discusses a modality that is critical for regulating multi-party conversations, eye gaze and, more generally, head and body orientation. In section 5.2 we see how other modalities, posture and gesture, are affected by group interactions.

Additional features have been added to the Demeanour framework to deal with interaction of more than two characters. The behavioural controllers of different characters are joined with the outputs of operators from one controller providing the values of the inputs to others. The controller must be able to generate different behaviour with respect to different characters in a conversation; to achieve this each operator in the controller may be evaluated independently for each character, each of which results in a different value. To be exact, each operator has a number of different types of value:

- A default value is used for parameters, which are the same for all characters, for example internal personality factors. This value is also used in attitudes and inputs when there is no value specified for a particular character.
- A value for each of the other characters in the conversation.

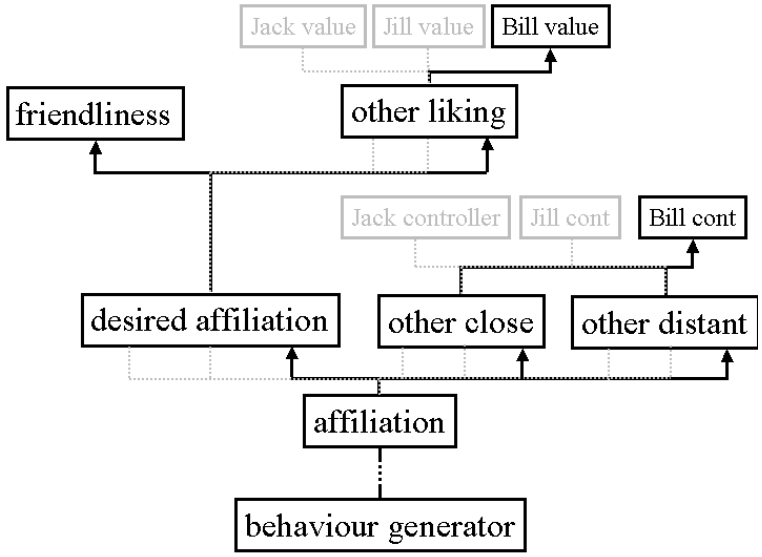

Figure 3: Querying the value of an operator for a particular character

- The average value of all the characters.
- There are a number of roles that are specified for characters, for example the current focus of attention (see section 5.1 for details). A role always stands for a character but that character varies. If a terms is evaluated for a role the result is the same as evaluating it for the character currently occupying that role.

Figure 3 shows how terms are evaluated for a particular character. A behaviour generating module (for example, the posture or eye gaze modules) will request to evaluate an operator for a given character. A compound operator evaluate each of the values it depends on for the character and then combines them as usual. Parameters have the same value for all characters and so can simply be evaluated. The value of an input corresponds to the value of a term in other characters' controllers. The value chosen is therefore the value from the controller of the character we are interested in. Attitudes can have different values for different characters. An example might be a parameter that represents liking towards another character. To give a concrete example, in figure 3 we can imagine evaluating the operator "affiliation" for character "Bill". This depends on "other close", "other distant" and "desired affiliation", so each of these are also evaluated for Bill. The first two are inputs terms and so they query Bill's controller for the appropriate value. Desired affiliation is a compound compound which depends on "other liking" and "friendliness". Other liking is an attitude so Bill's specific value is used. On the other hand friendliness is a parameters that is character-independent so the default value is used.

5.1 Eye gaze and social attention

Gaze plays a vital role in group conversations. Gaze primarily determines which member of a group is a person's focus of attention, which in turn has important implications for group interaction. Firstly, it helps regulate conversation (Argyle and Cook (1976)), listeners tend to look at talkers more and looking at another person can "cede the floor" to them. Gaze also indicates attitude, higher levels of gaze are connected to

affiliation, in group situations liked members will tend to be looked at more. Finally, much expressive behaviour is aimed at the focus of attention and so the focus often determines the nature of the behaviour.

Based on our own observations, we extend our model of eye gaze to a more general model of what we call "social attention". The eyes are not in fact the sole determinants of attention, in our observations we noticed how people can use, for example the torso, to indicate focus and interest while in fact they are looking elsewhere. Two examples can easily be interpreted in terms of attitude. Someone listening to a companion with their head fully oriented towards the talker but whose eyes are wandering, clearly is not entirely interested. Similarly someone looking at a talking companion but whose body is orientated in the opposite direction towards someone else, is likely to be showing a strong preference between their companions. We therefore generalise our eye gaze model to allow for multiple foci of attention, for the eyes, head and torso (or body). The character maintains these three foci of attention, which may often be the same object. The three foci are determined in largely the same way with some variations. The eyes are considered to be the primary focus as they, of course, actually determine what is seen. The other two foci change relative to the eyes. We will therefore start our discussion with the model of the eyes.

Our eye gaze model is a generalisation of that described in Gillies et al. (2004). Each character may either look at another character or a location in the world (when they are looking "away" from other characters). The locations generally have no social implication (beyond the implication of not looking at any character), with one exception. We found, in our observations one particularly interesting behaviour. There was a tendency, when people were not orienting themselves towards a particular individual, to orient towards the average location of members of the group. This tendency was particularly true of the orientation of the torso, which tended to move less. This appears to be a way of maintaining a connection with the group as a whole while momentarily attending to particular individuals. We therefore include an "average" location which is calculated (perhaps slightly excessively deterministically) as the mean position of the other characters. Each character and location, is represented as a potential focus of attention, which that character may look at. These foci are stored in a list for future gaze behaviour.

Each character will look at the various potential foci for a different proportion of time. The actual pattern of gaze is determined by two values for each focus: the *gaze target* and *stare target*. The *gaze target* is the maximum proportion of time that the character can look at a given focus, calculated over a finite time window, while the *stare target* is the maximum length of time that the character can look uninterruptedly at the target. If either of these values has been exceeded, the character must change its current focus of attention and therefore look somewhere else. The natural animation therefore originates in a balance between a tendency to look combined with a (socially-induced) tendency not to stare excessively. The values of these two parameters depend on a number of social factors, which are described in the next section. When a character does shift its gaze, it will choose a new focus, the probability of choosing each potential focus is proportional to the different between its gaze target and the real proportion of time it has been looked at in recent time window. When a character is looked at it adopts the role of "focus of attention" (see above), which is used by the posture module.

The above discussion described how the eyes are animated. The head tends to be less mobile than the eyes and the torso even less so. However, when they do move they tend to follow the eye gaze. Therefore the main parameter that controls the head and body are the probabilities that each will follow the eyes. The head and body also have independent gaze and stare targets for each potential focus, to ensure that they do not point at certain objects excessively. If either is exceeded a head or body movement will be forced. A behaviour we have noted in our observations is that people often

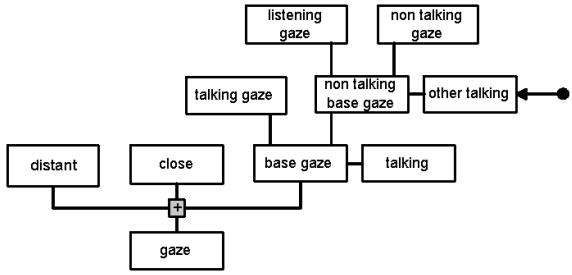

Figure 4: The section of behavioural controller dealing with eye gaze.

orient their torso towards the average position in the group. We therefore give a high gaze and stare target for the body towards the "average" focus. Finally a focus might be at too high an angle to look at without turning the head or body, in this case the head and body are forced to move overriding the above factors.

5.1.1 Calculating social attention parameters

As described above, the social attention model is regulated by a number of outputs from the behavioural controller, most important of which are to the stare and gaze targets for a particular character or location, with either eyes, head or body, and also the probability of the head or body following the eyes. Each focus of attention, and therefore each character has a different value for each of these. The values for each character are calculated by evaluating each output for that character, as described above. They are varied by the two types of expressive behaviour we are dealing with, conversation and attitude.

Social attention is highly influenced by the flow of conversation, for example, Argyle and Cook (1976) have shown that people will look at the other 75% of the time while listening but only 35% while speaking. Gaze is also very important for taking turns in conversation; for example, when a new talker is about to start speaking they will tend to look away while conversational partners will look more.

We can therefore define a number of speech conditions with different gaze targets based on Argyles' proportions. We firstly determine whether each character is talking or about to start. As our system is based on a textual interface where messages are sent instantaneously we must define what it means to be talking at a given time. We define an avatar to be speaking, if their user has sent a message, until another person sends a message or after a short time interval has elapsed. We define a person to be about to talk, if they have started to type a message but not yet sent it. Each avatar's behavioural controller can therefore have available information as to whether the character itself (which we call "self") is talking or about to, or whether other characters (called "other") are. The gaze targets for each character are evaluated independently as described at the start of section 5, so for any given calculation we only need to know whether the self character and one other character are talking. This information gives us 5 conditions (the later conditions in this list take precedence over the earlier ones):

- neither character is talking, in which case gaze and other orientation is low (no data is available on this condition so we use a default of 10%).
- the self character is listening to the other character, in which case gaze is high (75% by default from Argyle and Cook's studies (Argyle and Cook (1976)))
- the self character is talking, in which case gaze is moderate (35% from the same study)

- the other character is about to start talking (we set this value to be exaggeratedly high at 90% for the reasons described below).
- the self character is about to start talking (again a low value of 10%)

Thus each of the conditions has a base value for its proportion of gaze (and proportional values for their stare length and other parameters). This aims to create a realistic variation of gaze during conversation. As well as adding realism, this use of social attention aims to provide some of the social cues that are lacking in traditional instant-messaging systems but that are vital to helping regulate conversation. One particular problem is that users will all tend to type simultaneously and, due to the time it takes to type, they are likely to be reacting to comments made when they started typing which have been superseded when the message is sent. In particular the fourth condition (increased gaze and orienting towards a character that is about to speak) promises to be a strong cue to who is composing a message and so is likely to encourage people to "cede the floor" to them, allowing them to speak before making another comment (as yet we will need to do more tests to know if this will be effective).

This base value is also affected by the affiliation attitude between the avatar that is looking and the one that is being looked at. A close attitude increases the gaze target (up to a maximum of 100%) and distant behaviour reduces it (to a minimum of 0%). This scaling is achieved by combining the base attention values with the values of "close" and "distant" as shown in figure 4. The exact formula used to determine the actual eye gaze is:

$$g = g_{cond} - g_{cond}\frac{distant}{d_{max}} + (1 - g_{cond})\frac{close}{c_{max}}$$

where g is the proportion of time spent gazing at the target on average. g_{cond} is the gaze proportion due to the condition (talking, listening or neither). $distant$ and $close$ are the values for the close and distant attitudes and d_{max} and c_{max} are the values at which the gaze proportion is either 0 or 1. The output of the operators shown in figure 4 is the "gaze" values which is passed to the eye gaze module to actually generate gaze behviour using the calculated proportions.

Stare targets for eyes, head and body are calculated with the same formula as gaze targets, only the base values vary. In our model the probabilities of head or body following the eyes are also calculated in the same way. We have noticed when observing conversations, that turning to some one with head and body is a more emphatic way of displaying attention. People are more likely to turn there head to a speaker, but glance with the eyes at other people and locations, and we might surmise that high affiliation would increase the probability of turning the head towards a person and not another. We therefore assume that the same factors that tend to increase gaze will also increase the probability of turning the head or body. The only difference is that the body has a high probability of pointing to the average location of the people in the group, as we have observed that this is a common behaviour, keeping the torso pointed to the group as a whole and moving head or eyes to look at different people.

5.1.2 Orienting motion

As described in section 5.1 the output of the social attention controller is a number of foci of attention for the character. These can be other characters or locations in the world. The purpose of the attention animation system is to turn the appropriate body part (eyes, head, body), towards the focus. Orienting one part of the body towards a focus is highly dependent on the orientation of other body parts and on the posture of the character. For example, the orientation of the torso alters the position of the head and therefore the rotation required to orient it to a particular focus. We therefore

cannot determine the exact orientation of a body part until the moment it is required. However, it would be desirable to know the motion of the character in advance, in order to ensure smooth movement by interpolating between the previous focus and the next. Rather than storing future orientations of the eyes, head and torso we store the foci themselves as positions in the world, and interpolate these in world space. We therefore have a smooth sequence of positions in world space. At each frame the eyes, head and torso are oriented towards these locations based on their actual position at that frame.

5.2 Posture and gesture

The posture of a character reflects their attitude to other characters: for example high affiliation is associated with close postures such as leaning forward or other close interaction such as a direct orientation whereas low affiliation or dislike is shown by more distant postures, including postures that present some sort of barrier to interaction, such as crossed arms. The values for attitudes are generated by Demeanour as described above and then used to generate postures by interpolating over a set of base postures as described below.

In multi-party interaction it is possible to have different attitudes to different members of a group. It is therefore important to evaluate the attitudes with reference to a particular character. In Demeanour, different postures are associated with different foci of attention, for example, postures of the torso, such as leaning forward, are associated with the body focus whereas the "head cock" posture is associated with the head focus. When postures are generated, the appropriate attitudes are evaluated with respect to the associated focus. If that focus is not a character, the attitude is evaluated as the average of all the characters in the group. Gesture values are calculated similarly but their values are set to zero if the character is not talking (or not listening in the case of back-channel gestures such as head nods).

The same basic engine is used for animating postures and gestures. They are both based on a series of basic postures or gestures (henceforth called basic motions) that are combined to generate new motions. The only difference between the two is that gestures vary over time, while postures are essentially static. The basic motions are combined by a weighted sum, where the weight of each basic motion determines its contribution to the motion of the character. The weights are generated based on the values of the attitudes. Each motion has an associated attitude value. The weights of each motion are generated at random, but from a distribution whose mean is proportional to the value of the associated attitude.

6 Profiles

Demeanour provides a system of character profiles for off-line customization by end-users or world designers. They provided the main mechanism for defining the individual charactersistics of characters and their relationships. By world designers we mean expert content creators with some programming skills or at least the ability to handle technologies such as XML used when defining character behaviour and adjectives for profile creation (see below). Interfaces for end users are aimed at typical computer game players, not experts but familiar with instant messaging and 3D navigation. Player characters are controlled mostly through a text chat interface, through which players can enter text to be spoke and emoticons which control the character's behaviour (as well as choosing profiles). The user interface is shown in figure 5.

A profile is a set of data that determines the unique behaviour of a character, i.e. how it differs from other characters. In Demeanour a character's behaviour is generated by a parameterised behavioural controller (the structure of the controllers

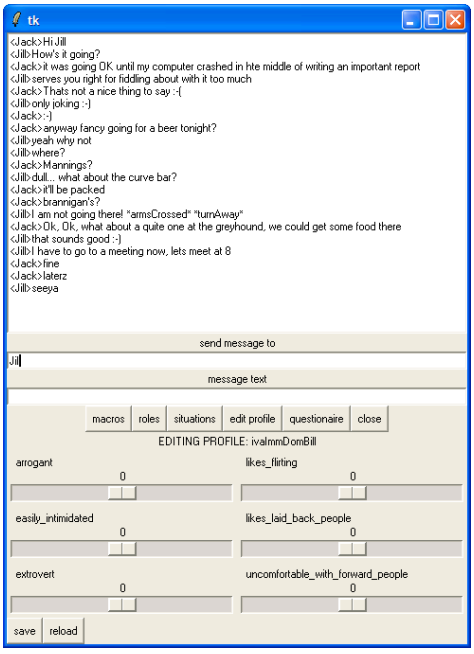

Figure 5: The user interface used for real time control of characters

is discussed in more detail in Gillies and Ballin (2004)). This controller can be the same for each character but changing the values of the parameters allows for different behaviour. Customization is possible by altering the values of the parameters e.g the weighting for how the closeness behaviour of other character affects a character's affiliation. A profile can set this weighting to a positive value to achieve reciprocating behaviour, negative for compensation and a low or zero value for indifference to the other's status.

Thus a profile consists of a number of values for parameters of the behavioural controller. These values are stored in an XML-based format separate from the controller definition. When a profile is loaded into a behavioural controller the values in the profile are used to set the parameters of the controller (profile values are matched to parameters by name). Profiles are used as a means of customising a character, and a means of providing contextual variation. This means there will be a number of profiles loaded in a controller at any given time. They are stored in a stack as shown in figure 6. The base of the stack is always the main profile that contains the context independent customisations of a character. Above this, a number of context dependent profiles are loaded as described in section 6.2. When a new context profile is loaded it is added above all the previously loaded profiles in the stack. Profiles higher up the stack will override profiles lower in the stack, so recently loaded profiles override older ones and user input overrides other profiles. However, this process can be controlled by giving priorities to values within a profile. Values can have two priorities, *required* and *optional*. Required values always override values lower down the stack but optional values only override other optional values, and so are only loaded if no profile has a required value for that parameter.

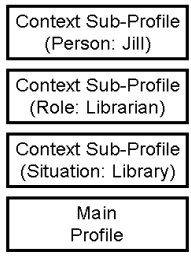

Figure 6: The profiles stack containing a number of loaded contextual profiles

6.1 Profiles for customisation

The primary function of character profiles is the customisation of characters. End users should be able to customise the behaviour of their character and designers of virtual worlds should be able to provide variety in the autonomous agents in their world. Each character has a main profile, at the base of the profile stack, containing values for the parameters of the behavioural controller that determines the unique behaviour of that character. This is the main customisation system for a given character.

To be an effective customisation method, easy to use tools must be provided for designing profiles. The most direct method is for the user to choose values to parameters whether by hand editing files or via a user interface. However, parameters are often closely linked to the internal workings of the behavioural controller and not necessarily intuitive to end-users, so this method should generally be confined to world designers and advanced users.

We propose the use of "adjectives". These are names in natural language that describe a particular character trait or group of traits that is understandable to end-users. These adjectives are mapped onto actual settings of the internal parameters, each adjective affecting a number of parameters. For example, 'extrovert' might combine dominance with high affiliation while 'easily intimidated' might indicate compensation behaviour to dominance (i.e. responding submissively to dominant behaviour). Each adjective is a fixed set of parameter values and therefore is itself a self contained profile. The adjectives themselves are chosen by world designers. They can be designed at the same time as the behaviour network, through direct profile authoring tools as above. An end-user designs their profile as a combination of the adjectives. They are presented with series of sliders each labelled with an adjective name, the values of the sliders represent the proportions of the various adjectives. The values contained in the adjectives are multiplied by the slider values and summed to obtain the final profile. This provides a customization tool that is easy to use, abstracts from the internal workings of the controller, and is itself easily customizable by world designers. Figure 5 shows an example of the user interface for choosing adjective weights.

6.2 Profiles and context

As described in the introduction the variability of human behaviour is not solely between individuals but within individuals. People behave very differently in different contexts and it is important to also model this sort of variability. The importance of this type of adaptation is brought out in work by MacNamee et al. (2002) and Maya et al. (2004). Goffman (1972) provides a fascinating description of how people's behaviour varies in different contexts. This is particularly true for characters in narrative where the character's behaviour must reflect the unfolding story.

In order to handle this sort of variability Demeanour uses a system of sub-profiles for specific context. A sub-profile is a small set of parameter values that are loaded in a given context to alter the behaviour of the main profile. These are loaded above the main profile in the stack as shown in figure 6, with more recently loaded sub-profiles overriding older ones.

The variation of a person's behaviour in different contexts can depend on a number of different factors and so these contexts themselves can have different meanings, for example, a relationship with a colleague may define a context for interaction with that colleague but the context would also depend on whether they are at work or in a social context. We divide contextual sub-profiles into three types depending on when they are loaded and to some degree who designs them. The system could be augmented to add a number of other types.

Person sub-profiles define a relationship to a particular individual and thus represent the attitude to that person. They are loaded when an interaction starts with that individual. These are the primary means of specifying relationships between characters as described below.

Role sub-profiles represent the behaviour when the character is performing a particular role. These roles are often related to work, for example, a waiter behaves very differently when actually serving customers than when not working or when interacting with other staff in the restaurant kitchen (Goffman (1972)). A practical example, might be eliminating all low affiliation behaviour when a waiter is at work to ensure politeness. Role sub-profiles can be loaded by the end-user or automatically loaded in a given situation. Role profiles can be created by world designers to give a user character specific behaviour in a role that the user might not have forseen or they can be created by an end user.

Situation sub-profiles are specific to a particular environment or narrative context, for example, flirting behaviour might be disabled in an office environment but re-enabled in an office party context. They are loaded automatically when the character enters a situation and applied to all characters in that situation.

6.3 Profiles and relationships

The profile system makes it possible for users to specify relationships between characters. A Person profile contains the attitudes of one character to another. Unlike other profiles which set default values for parameters and attitudes, Person profiles set the attitudes for a specific character. Each person profile is associated with another character and is loaded when an interaction with that character begins. At this point the attitude values for that character are instantiated with the values from the profile. Using the multi-party interaction mechanisms described in the first part of this paper it is possible to load multiple person profiles, for the different characters in an interaction. Each of these profiles result in independent attitudes for their characters, thus supporting the multi-party conversations we have described and enabling end users to specify diverse relationships between characters.

7 Conclusion and Results

We have described the Demeanour framework's methods for multi-party interaction with diverse relationships between multiple virtual characters. The system of profiles

allows end users to customize the individual behaviour of characters and the relationships between characters. Figure 7 shows results of an animated conversation generated by our system.

Acknowledgements

We would like to thank BT plc. for sponsoring this research. We would also like to thank the UCL Department of Computer Science Virtual Environments and Computer Graphics group for their help and support, and Amanda Oldroyd for the use of her character models.

References

Argyle, M. (1975). *Bodily Communication*. Routledge.

Argyle, M. and Cook, M. (1976). *Gaze and Mutual Gaze*. Cambridge University Press.

Badler, N., Philips, C., and Webber, B., editors (1993). *Simulating Humans: Computer Graphics, Animation and Control*. Oxford University Press.

Bécheiraz, P. and Thalmann, D. (1996). A model of nonverbal communication and interpersonal relationship between virtual actors. In *Proceedings of the Computer Animation '96*, pages 58–67. IEEE Computer Society Press.

Bevacqua, E., Mancini, M., and Pelachaud, C. (2004). Speaking with emotions. In Olivier, P. and Aylett, R., editors, *AISB workshop on Language, Speech and Gesture for Expressive Characters*, University of Leeds.

Blumberg, B. and Galyean, T. (1995). Multi-level direction of autonomous creatures for real-time virtual environments. In *ACM SIGGRAPH*, pages 47–54.

Cassell, J., Bickmore, T., Campbell, L., Chang, K., Vilhjálmsson, H., and Yan, H. (1999). Embodiment in conversational interfaces: Rea. In *ACM SIGCHI*, pages 520–527. ACM Press.

Cassell, J., Nakano, Y., Bickmore, T., Sidner, C., and Rich, C. (2001a). Non-verbal cues for discourse structure. In *41st Annual Meeting of the Association of Computational Linguistics*, pages 106–115, Toulouse, France.

Cassell, J., Vilhjálmsson, H. H., and Bickmore, T. (2001b). Beat: The behavior expression animation toolkit. In *Proceedings of ACM SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, pages 477–486.

Chi, D., Costa, M., Zhao, L., and Badler, N. (2000). The emote model for effort and shape. In *ACM SIGGRAPH*, pages 173–182. ACM Press/Addison-Wesley Publishing Co.

Colburn, A., Cohen, M., and Drucker, S. (2000). The role of eye gaze in avatar mediated conversational interfaces. Technical report, Microsoft Research.

Coulson, M. (2002). Expressing emotion through body movement: a component based approach. In Aylett, R. and Cañamero, L., editors, *AISB workshop on Animating Expressive Characters for Social Interactions*, Imperial College London.

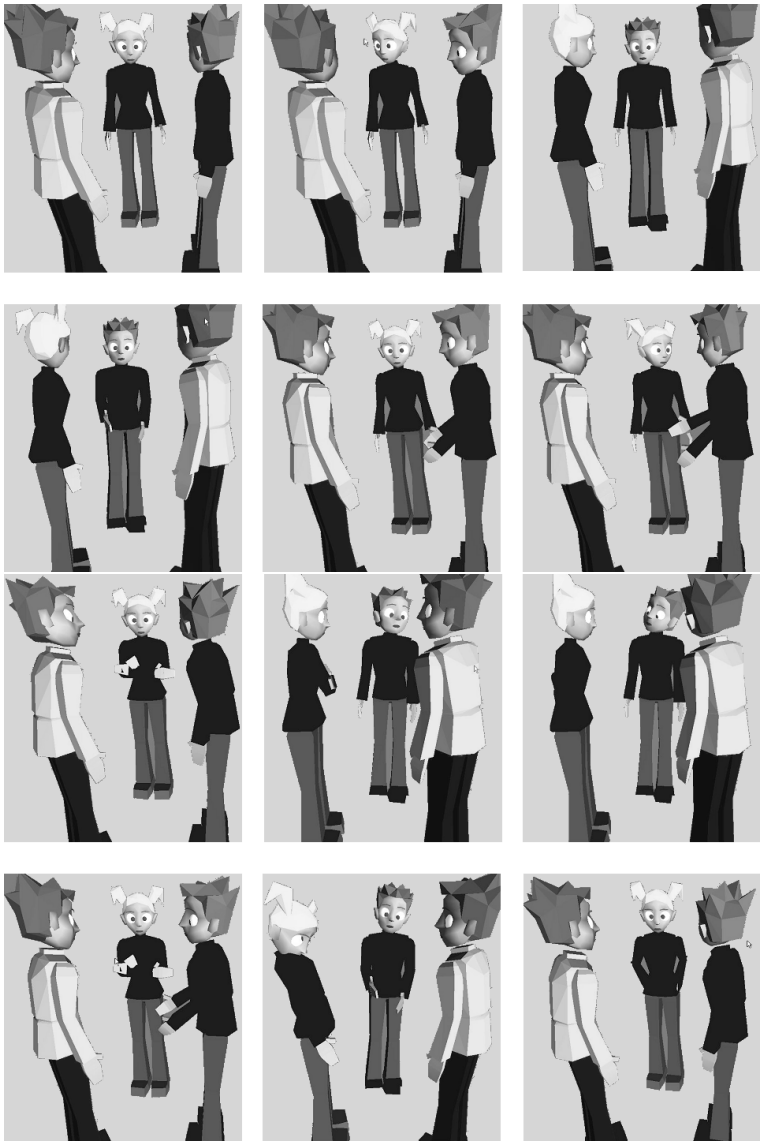

Figure 7: Frames from an animated multi-party conversation generated by our system. The top set of images show a neutral conversation with no specific relationships between characters, while the bottom row shows the effect of specifying relationships.

DeCarolis, B., Pelachaud, C., Poggi, I., and Steedman, M. (2004). Apml, a markup language for believable behaviour generation. In Prendiger, H. and Ishizuka, M., editors, *Life-like characters: tools, affective functions and applications*, pages 65–87. Springer.

Garau, M., Slater, M., Bee, S., and Sasse, M. A. (2001). The impact of eye gaze on communication using humaniod avatars. In *ACM SIGCHI*, pages 309–316.

Gillies, M. and Ballin, D. (2003). A model of interpersonal attitude and posture generation. In Rist, T., Aylett, R., Ballin, D., and Rickel, J., editors, *Fourth Workshop on Intelligent Virtual Agents*, Kloster Irsee, Germany.

Gillies, M. and Ballin, D. (2004). Integrating autonomous behavior and user control for believable agents. In *Third international joint conference on Autonomous Agents and Multi-Agent Systems*, Columbia University, New York City.

Gillies, M., Crabtree, B., and Ballin, D. (2004). Expressive characters and a text chat interface. In Olivier, P. and Aylett, R., editors, *AISB workshop on Language, Speech and Gesture for Expressive Characters*, University of Leeds.

Goffman, E. (1972). *The presentation of self in everyday life*. Pelican.

Guye-Vuilléme, A., T.K.Capin, I.S.Pandzic, Magnenat-Thalmann, N., and D.Thalmann (1999). Non-verbal communication interface for collaborative virtual environments. *The Virtual Reality Journal*, 4:49–59.

Jan, D. and Traum, D. R. (2005). Dialog simulation for background characters. In *Proceedings of the 5th International Working Conference on Intelligent Virtual Agents*, Lecture Notes in Artificial Intelligence, pages 65–74. Springer.

MacNamee, B., Dobbyn, S., Cunningham, P., and O'Sullivan, C. (2002). Men behaving appropriately: Integrating the role passing technique into the aloha system. In Aylett, R. and Cañamero, L., editors, *AISB workshop on Animating Expressive Characters for Social Interactions*, Imperial College London.

Maya, V., Lamolle, M., and Pelachaud, C. (2004). Influences on embodied conversational agent's expressivity. In Ruth Aylett, P. O. and Cavazza, M., editors, *AISB workshop on Language, Speech and Gesture for Expressive Characters*, pages 75–85, Univesity of Leeds.

Mehrabian, A., editor (1972). *Nonverbal Communication*. Aldine-Atherton.

Perlin, K. and Goldberg, A. (1996). Improv: A system for scripting interactive actors in virtual worlds. In *Proceedings of SIGGRAPH 96*, Computer Graphics Proceedings, Annual Conference Series, pages 205–216, New Orleans, Louisiana. ACM SIGGRAPH / Addison Wesley.

Rickel, J. and Johnson, W. L. (1999). Animated agents for procedural training in virtual reality: Perception, cognition, and motor control. *Applied Artificial Intelligence*, 13:343–382.

Soltysiak, S. J. and Crabtree, I. B. (1998). Automatic learning of user profiles – towards personalisation of agent services. *BT Technology Journal*, 16(3):110–117.

Thórisson, K. (1998). Real-time decision making in multimodal face-to-face communication. In *second ACM international conference on autonomous agents*, pages 16–23.

Tu, X. and Terzopoulos, D. (1994). Artificial fishes: Physics, locomotion, perception, behavior. In *ACM SIGGRAPH*, pages 43–49.

Vertegaal, R., Slagter, R., van der Veer, G., and Nijholt, A. (2000). Why conversational agents should catch the eye. In *SIGCHI 2000 short talks*, pages 357–358.

Vilhjálmsson, H. H. and Cassell, J. (1998). Bodychat: Autonomous communicative behaviors in avatars. In *second ACM international conference on autonomous agents*.

Petri Nets for Representing Story Plots in Serious Games

Cyril Brom*, Tomáš Holan*, Daniel Balaš*, Adam Abonyi*, Vít Šisler† and Leo Galamboš‡

* Department of Software and Computer Science Education, Charles University in Prague, Malostranske namesti 25, Prague 1, Czech Republic, *brom@ksvi.mff.cuni.cz ; holan@ksvi.mff.cuni.cz ; addabis@gmail.com ; adam.abonyi@gmail.com*

† Institute of Information Studies and Librarianship, Charles University in Prague, U Krize 8, Prague 5, Czech Republic, *vsisler@gmail.com*

‡ Department of Informatics and Telecommunications, Czech Technical University in Prague, Konviktska 20, Prague 1, Czech republic, *galambos@fd.cvut.cz*

Abstract

This article describes a novel technique for representing plots of stories in computer games and for unfolding stories according to these plots. The technique is based on Petri Nets. Its main advantages are that 1) it copes well with branching stories, which 2) can evolve in parallel in 3) large virtual worlds. While the issue of branching stories has been already studied in literature, the latter two issues have not been sufficiently addressed yet. The technique has been evaluated on a prototype implementation, and subsequently used in two serious games Europe 2045 and Karo. Europe 2045 is an on-line multi-player strategy game aimed at education of high-school students in economics, politics, and media studies, which is presently evaluated at high-schools in the Czech Republic. Karo is a storytelling simulation that is intended to engage students in decision making concerning social relations, and work on it still continues. While Europe 2045 simulates Europe in a coarse-grained way, Karo features a detailed virtual world inhabited by virtual humans. Besides presenting the Petri Nets modifications used in the prototype and in both of the games, the article details several scenarios from the games, and on a general level, it discusses the strengths and weaknesses of implementation of Petri Nets in virtual storytelling applications.

1 Introduction

The idea of using computer games to support training and learning objectives is more than 30 years old (de Freitas, 2006). Recent work has explored the potentialities of commercial strategy games and simulations in formal education and their alleged advantages over classical e-learning and edutainment tools, e.g. Egenfeldt-Nielsen (2005). Indeed, many of such games have been experimentally integrated to formal curricula in the last four years. Perhaps the most prominent case studies have been conducted with The Sims 2, Civilization III, and Europe Universalis II (Egenfeldt-Nielsen et al., 2006; Sandford et al., 2007; Squire, 2004), but other attempts exist as well. The results from these pilots are promising, but also ambiguous in some aspects, e.g. Squire (2004). Hence, so called "serious" or "educational" games are starting to achieve increasing amount of attention. These games are, contrary to commercial games, intentionally developed as educational tools, which makes their integration into formal education easier. For example, a role-playing game prototype Global Conflicts: Palestine has been recently evaluated in a Denmark high-school with positive outcome (Egenfeldt-Nielsen et al., 2006). Another studies are being conducted, including FearNot!, an anti-bullying educational game (Aylett et al., 2005), and Revolution, a multi-player educational role-playing game concerning American War of Independence (Revolution RPG, 2007; Francis, unpublished).

As a part of European funded project "Integration of IT Tools into Education of Humanities" we have developed an educational game Europe 2045 (Brom et al., 2007b), and are developing a game Karo. Europe 2045 is likely the first on-line multi-player strategy game worldwide aimed at education of high-school students in economics, politics, and media studies. Its core constitutes a social-economic simulation, i.e. it models Europe in a coarse-grained way. Karo is a storytelling simulation that is intended to engage students in decision making concerning social relations. Contrary to Europe 2045, Karo features a virtual world inhabited by virtual humans, that is autonomous agents (Wooldridge, 2002) imitating behaviour of humans embedded into the environment. We emphasise that Europe 2045 is a fully developed game, which is presently being evaluated by high-school students, while programming work on Karo still continues.

Importantly, narrative aspects are strong in the both games. The reason for that should be clear: storytelling has played an important role in humanities education since the advent of formal schooling (de Freitas, 2006). Stories help to build a learning context, students can better understand the problematic through them, stories increase their involvement, and consequently their motivation.

Specifying plots of stories and controlling the course of a game in accordance with these plots is a well known problem (e.g., Aylett, 2000; Louchart and Aylett, 2003). It was indeed one of the most challenging goals we faced during the development. Essentially, both games had to be designed in order to meet the following requirements:

a) The story plots to be branching.

b) The episodes to be triggered by various initial conditions depending on the time and state of the word (including virtual humans in the case of Karo).

c) The virtual world to be very large.

d) The episodes to can happen in parallel. Europe 2045 features more than 20 countries, which could be played simultaneously, each having defined different episodes. A prototype scenario developed recently for Karo features about 10 different virtual humans, which are simulated all the time, being located in different parts of the virtual world.

e) The technique for specification of the plots to be intuitive enough for a high school teacher or another user (typically an undergraduate university student of humanities) to be able to design new scenarios for the game.

In our previous work, we investigated Petri Nets as a plot specification technique (Brom and Abonyi, 2006). We found this technique extremely user-friendly and very fitting for large virtual words and stories evolving in parallel, contrary to other approaches described in literature (which have, it must be acknowledged, other advantages for other kinds of virtual worlds). More or less, advantages of Petri Nets match the requirements mentioned above. Hence, we have adopted Petri Nets for these games.

This article is a report on this work. The overall goal is to introduce Petri Nets as a new technique for representing plots in storytelling applications, and to discuss their applicability. In the whole text, the different perspectives of a designer, who use Petri Nets as a specification tool, and a programmer, who use it as an architectural underpinning of the story manager, are emphasised.

We start with introducing Petri Nets in general in Section 2. Section 3 analyses various methods for representing plots and controlling stories, contrasting them with Petri Nets. Section 4 provides a formal account of a Petri Nets modification we developed formerly in Brom et al. (2006). This section is aimed at explaining in depth a particular refinement of Petri Nets that can be used for storytelling purposes. An example of a prototyped story is given as well.

Section 5 introduces Europe 2045 and Petri Nets modification used in there, which differ a bit from the formal model described in Section 4. This part of this article is based on our work reported in Brom et al. (2007b). In Section 6, Karo is overviewed and its Petri Nets modification sketched. Since Karo should be regarded rather as a prototype at present moment, we will visit it relatively briefly here. Nevertheless, the results we gained so far allow us to conclude that Petri Nets can be scaled for applications featuring virtual humans. Section 7 discusses the strengths and weaknesses of our method and concludes.

2 Petri Nets

This section gives a brief description of Petri Nets in general. Petri Nets is a specification technique frequently used in software engineering. A basic variant of it consists of *containers* (or places, represented by a circle: ○), *tokens* ("the pellets": ●), *actions* (or transitions, ▭), and *transition function* (→). The containers contain the tokens. If a sufficient number of tokens is contained in specific containers, an action is triggered. After firing an action, the tokens that helped to fire this action are removed, and some new tokens are generated (see Fig. 1a). Which tokens fire which action and which action generates tokens to which containers is specified by the transition function (and depicted by arrows). At one instant, several containers can contain tokens, which allows for concurrent triggering of actions. Obviously, a conflict between two applications of the transition function may appear (see Fig. 1b). Such a conflict can be solved in various ways. For example, we can choose one of the actions non-deterministically.

This basic kind of Petri Nets can be extended by introducing different types of containers, tokens, and transition functions. For example, tokens can have a state: such modification is typically called *coloured* Petri Nets (the colour meaning the state). For more thorough introduction to Petri Nets, we recommend the reader to consult Petri Nets World (2007).

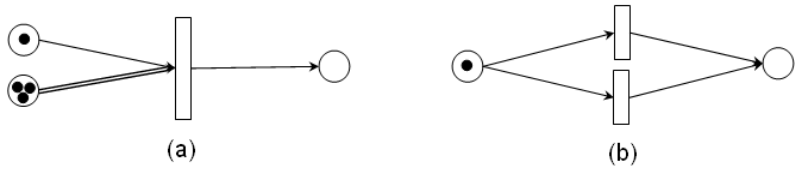

Figure 1: Petri Nets examples. a) The action generates one token if there is one token in the upper container and two in the lower container. b) The two actions are in conflict.

2.1 Petri Nets and a Story Manager

In this article, we discuss a particular kind of storytelling applications, that having plots of stories specified in advance by a designer. Actually, one may also be interested in automatic generation of stories based only on some pre-specified constraints, see e.g. Reidl and Young (2006), or in emergence of stories from interaction of autonomous human-like agents, as discussed in Aylett (2000), but this is not our case for our stories must precisely fit into the formal curricula (Brom et al. (2007b); note, however, that they still are branching and can evolve in parallel).

For explanatory reasons, let us view the architecture of the kind of applications we are discussing as being three-layered. The individual layers stand for a GUI, a simulator of the physical world, which presents the setting for the story (and possibly is inhabited by virtual humans), and a story manager (see Fig. 2). Each of the layers is more or less an autonomous component. The purpose of the GUI and the simulator are obvious. The story manager holds the representations of definite plot scenarios and a state of the story being unfolded. The state of the story can be changed based on the plots, and on the state of the simulated world. Conversely, state of the world can be changed based on its previous state and the state of the story. Importantly, not all changes of the state of the world are caused by the story manager! For example, in an application featuring virtual humans, the virtual humans can be controlled by the simulator most of the time (or they can be autonomous to some extent), and the story manager can only alter their high-level goals from time to time, i.e. only influence the virtual world, which is driven by the simulator primarily. Note, that also a user interaction changes the world (by definition of interaction). Hence, we can describe the interaction among the user, the world and the story manager by the following functions:

$$\text{story\_next}: \quad W \times S \rightarrow S \tag{1}$$

$$\text{world\_next}: \quad S \times W \times I \rightarrow W \tag{2}$$

Here, S is a set of possible states of the story, W is a set of possible states of the world, and I is a set of possible interactions of the user. The results of the function $story\_next$ is "computed" by the story manager, while those of the $world\_next$ by the simulator.

This notion of a separate storytelling component is actually not new. Various authors employ it, prescribing it various roles based on the intended features of the final application - see e.g. Magerko (2006). Following this architectural metaphor, Petri Nets are exploited in two ways. First, they present the mechanism for representing story plots, i.e. the function $story\_next$, a specification tool for the purposes of a designer. Second, in run-time, they also represent the state of the story being unfolded, i.e. $s \in S$. This means, that they are also the technical underpinnings of the story manager.

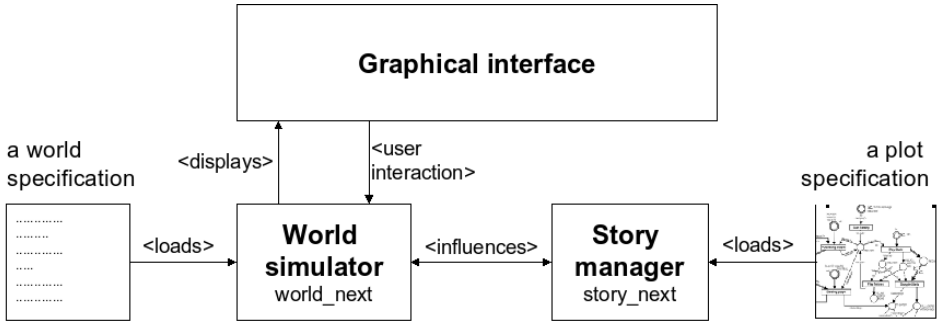

Figure 2: Architecture of a "typical" storytelling application that features a virtual world simulator and a story manager, which works with pre-specified plots of stories. Note that by virtual world, we mean both a world inhabited by virtual humans, which is the case of Karo, and that not inhabited by them, which is the case of Europe 2045.

3 Analyse of Plots Representational Techniques

The issue of generating/controlling stories in games and storytelling applications is notoriously known. Most techniques come from games and experimental simulations featuring human-like actors. This is also a case of Karo, but not Europe 2045. In this game, the story events are more abstract; they deal with whole populations, with a country economy etc. However, formally, the problem is very similar in both cases. Also the architectures of both application matches the schema at Fig. 2. In this section, we analyse several plot representational techniques, contrasting them with Petri Nets. We remind that our aim is evolving a story based on pre-specified plots with several requirements detailed in Sec. 1.

3.1 Rule Based Systems

A story manager viewed in an abstract manner as a component computing the function $story\_next$ (1) is actually a reactive component that responds to some events by triggering new events that change the state of the virtual world. Once we have this metaphor, we can easily come up with an idea of computing this by if-then rules matching a state of the story and of the world and executing an action in the world. Because of the requirement on large worlds (c), also large story plots, we argue that using just a list of such if then rules as a method for specifying plots is not a good idea. The reason is that for large plots hundreds of rules are needed. A designer that has to write down the rules needs a methodology, i.e. constraints that would guide him or her in the space of possible sets of rules that can be written down (see req. e)). A pure rule based system does not posses any such methodology.

Fortunately, there are several techniques that are a sort of rule based system, which present such a methodology. One of them is Petri Nets. Another one, which will be discussed here because it is already used in storytelling, is deterministic finite-state machines.

3.2 Finite State Machines

A well known branch of techniques for specifying plots are deterministic finite-state machines (dFSMs) (e.g., Sheldon, 2004; Silva, 2003). The formal specification of a dFSM

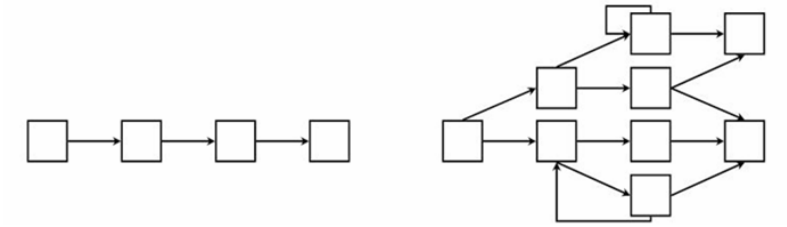

Figure 3: Story plots as dFSMs. The linear plot is on the left, the branching on the right.

constrains a manner of specifying the if-then rules, i.e. it presents the desired methodology. Each state of a dFSM represents a story episode, and a transition is a trigger that detects the end of the episode and starts a next one. Natural advantage of dFSMs is that they are formal, and yet graphical (Fig. 3), which makes them easily intelligible.

However, a classical dFSMs was not suitable for our purposes, since they cannot cope with the issues of parallelism (req. d)). On the other hand, non-deterministic FSMs can cope with it, but they are not easily comprehensible (e)).

Sometimes, so-called branching graphs or branching trees are used in storytelling, e.g. Sheldon (2004); Gordon (2004). We view them, more or less, as a sort of dFSMs, with similar advantages and disadvantages. Literally, they are not dFSMs, but describing subtle differences would be unnecessary for the purposes of this article.

3.3 Beat approach

Both for Europe 2045 and Karo, our aim was to author plots and to keep an unfolding story as close to one of "optimal stories" as possible. This objective is similar to this of Mateas (2002). However, his beat approach fits better to the domains of small worlds featuring virtual humans. We could not find out how to scale his solution to cope with the parallelism issue, and the large world issue (reqs c, d)).

3.4 HTN planning

In the field of emergent narrative and narrative generation, a planning formalism is frequently used, hierarchical task network planning being the most prominent (Aylett et al., 2005; Cavazza, 2002; Reidl and Stern, 2006). This technique can cope well with the requirements a), b), c), and d). Actually, it goes far beyond a pure reactive rule based story evolving in that a planner can perform a look-ahead search. However, HTN planning is not too friendly for a non-AI expert (req. e)).

To tackle e), one could introduce a "presentation layer" for a formal planning system to disguise the underlying representation and develop an authoring interface; however, this is time-consuming activity. In fact, Petri Nets (and dFSMs as well) naturally feature this presentation layer: as said above, they are a sort of rule based system, and it is this underlying rule based system, that they innately present in a graphical form.

Since we were interested in pre-specified plots but neither in a pure emergent narrative, nor in automatic story construction, and because of the unintelligibility disadvantage of the HTN planning, we decided not to use this approach. This does not mean, however, that for other domains, planning may not be the best option.

3.5 Petri Nets

Petri Nets have natural comprehensibility advantage of the dFSMs. They are formal, and yet allow for graphical depiction, which mirrors all of their (or most of their, depending on the complexity of a particular Petri Nets variant) formal features. Hence, they fit well as a specification interface between a game designer and a programmer. Indeed, we were able to use them both in Europe 2045 and Karo. First, they were employed in an informal manner as a specification tool for the designer, i.e. a tool for specifying the function $story\_next$ (1). Second, a rigorous counterpart of this informal specification tool presented the architectural underpinnings of the story manager developed by our programmer, i.e. a mechanism for computing the results of this function.

For the story manager, it is important to hold the state of the story, i.e. $s \in S$. How s and S are represented by Petri Nets? Let us contrast it with the representation of s and S in a dFSM. For a dFSM, S is a set of its states, and s is the state the machine is currently in. This actually holds for a Petri Net as well. However, a state of a Petri Net, i.e. s, is represented by which tokens are in which containers, and hence S is a set of all possible combinations of allocations of tokens to containers. This is much more flexible representation, and finally allows us to cope with the parallelism issue.

Petri Nets have been already employed in storytelling. Natkin (2003) used them to a retrospective analysis of a computer game story. They were also used for prototyping purposes (Brom and Abonyi, 2006), and for plot monitoring in a simple game (Delmas et al., 2007). However, to our knowledge, none of these work implemented a story manager for a real full-fledged game.

4 Petri Nets for Representation of Story Plots

As said in Section 2, Petri Nets are a whole class of specification techniques; many modifications exist. Not surprisingly, we have found that for purposes of each particular storytelling application, it is beneficial to develop a particular modification that fits best for what the application demands. For the explanatory reasons, we proceed now as follows. We first describe in detail a particular refinement of Petri Nets, based on so-called timed coloured Petri Nets. In fact, we used this refinement for prototyping purposes formerly, which allowed us to conclude that Petri Nets are a fruitful technique in the storytelling domain (Brom and Abonyi, 2006). In Section 5, we describe Europe 2045, and overview the Petri Nets modification it uses. Note that although our technique was intentionally designed for storytelling applications featuring virtual humans, neither the prototype, nor Europe 2045 actually exploit them. However, in Section 6, we introduce Karo, where the method is really used for controlling a story employing virtual humans.

To set the terminology, we say that a *PNA-model* is the type of Petri Nets formalised in this section, and a *PNA-plot* is a specification of an individual plot by means of the PNA-model. Sections 5 and 6 then introduce *PNB-model* and *PNC-model* respectively.

Every PNA-plot can be loaded from an xml-file by an application called TEST we developed, which features an abstract story manager for running and verifying PNA-plots. TEST works as follows: After a PNA-plot is loaded and the simulation started, events start to occur according to the PNA-plot in an abstract manner. That means that the TEST computes the $story\_next$ function (1), but there is no simulator of a virtual world to compute the $world\_next$ function (2). The result of this function is to be determined manually by a user through a check-box like interface in TEST; this mimics the changes that would be imposed on the world by the simulator or a user interacting with the world

in case of a full-fledged application.

4.1 Story example

For explanatory reasons, we first introduce a dummy story, whose plot was formalised as a PNA-plot and evaluated in TEST. The story is a simple fantasy narrative set in a village in an evening. It is a rather small story, just an episode from a larger tale. There are the following actors and groups of actors:

- MAGICIAN: a user-actor. She needs a puppet (for whatever reason).
- GUARDS: four bum-bailiffs. One of them is a friend of the magician; he has given her a note that there will be a puppet theatre coming this evening.
- PUPPET THEATRE: four artists. One of them is a twin of a guy who used to steal in the village several years ago, was arrested but managed to escape.
- ROBBERS: a band having pilfered in the village for a few weeks.
- VILLAGERS: citizens of the village. Some of them will mistakenly recognise the twin as his robber-brother.

Two things are going to happen in parallel. First, the robbers are going to pilfer this evening and the guards will try to capture them. Second, the troupe is going to perform a piece in a pub, and in the course of the play, some villagers will mistake the twin as his robber-brother and a brawl will flare up in the pub. Consequently, a fire might start, that would destroy the theatre and all the puppets.

Story can evolve in several ways. All events occur in a probabilistic manner: it is not sure whether or not the guards capture the robbers, or even notice them; whether or not somebody mistakes the twin; and whether or not the fire breaks out.

The magician is allowed to influence the story in the following ways: She can buy a puppet (before the theatre is reduced to ashes), or steal one in the course of the brawl. If she helps the artists in the brawl, she will be given a puppet for free, provided that no fire has broken out.

She can call the guards during the brawl using a spell. In this case, the guards interrupt the chasing and rush to the pub. This reduces the probability of the fire breaking out significantly, especially if guards have just caught the robbers. She can also put out the fire immediately by casting a spell.

A part of the PNA-plot for this story is depicted in Figure 4. The most important features of the portrayal are described in Section 4.2.

4.2 PNA-model

This section gives the formal description of the PNA-model and describes the most important features of the graphical representation of the PNA-plot depicted in Figure 4. An algorithm for driving a story according to a PNA-plot is also presented.

Notice that PNA-plot's portrayals are informative only; what is important is a background formal specification. Nevertheless, a designer works only with the graphical specification and informal textual descriptions; there is no need for him or her to know the formal delicacies, which must be known, on the other hand, to the programmer. As already said, it is the programmer whose job is to convert designer's plot specifications into the code, and clarify ambiguities that may have appeared.

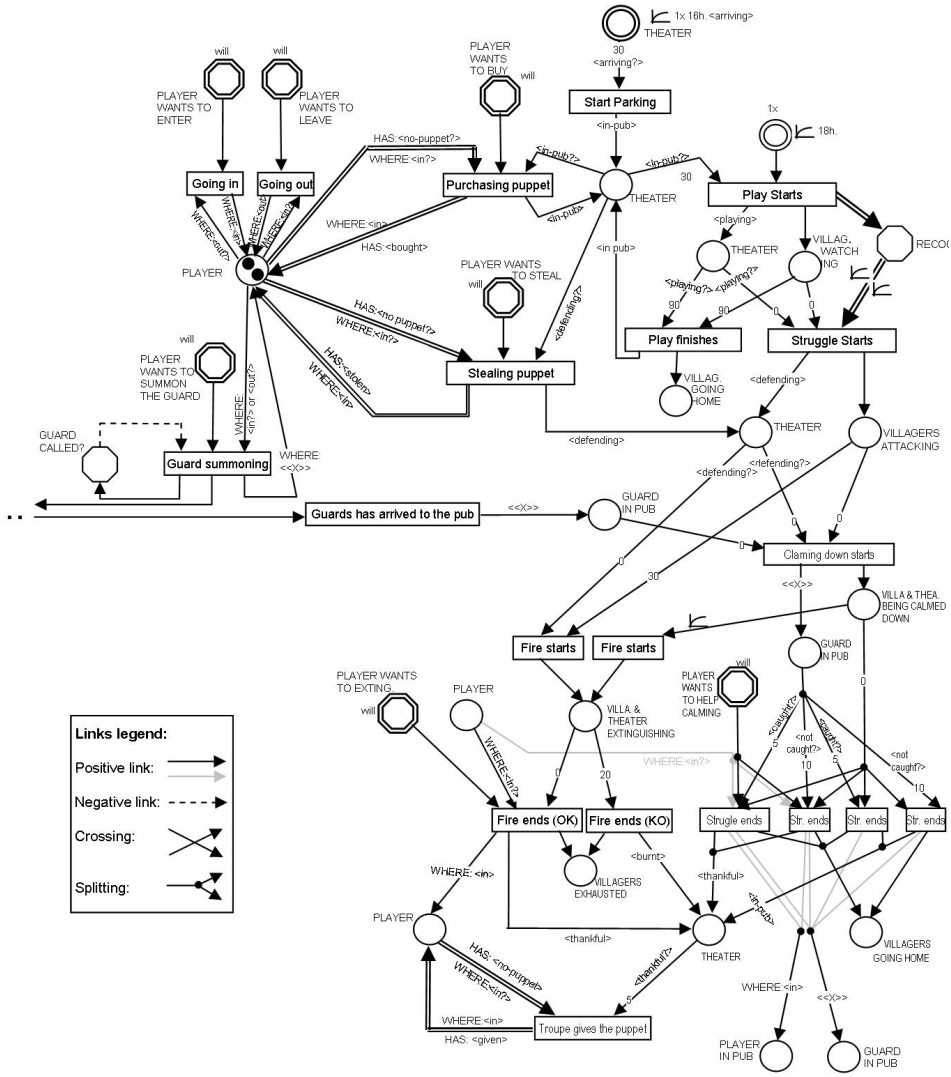

Figure 4: The "pub-plot" in the initial state is depicted. Notice that a plot can be much larger and several PNs can run in parallel. In particular, in our case, the "catching-plot" is connected to the "pub-plot" on the left (indicated by "…"). This part is not described here for brevity.

The PNA-model features following entities: *containers* (which are places – "the circles": ○, ◎, ▢, ▣), *tokens* (which are "the pellets": ●), *actions* (which are transitions – "the rectangles": ▭) and *triggers* (more or less, they correspond to a transition function – to "the arrows": →, ⇢). The purpose of a trigger is to fire an action according to tokens' location, or to add or to remove tokens from containers.

Tokens. Every *token* has a name, a colour, age, and a state. For every colour, there is a set of corresponding states. Actually, a colour is like a type of the token, but for conventional reasons, we keep the designation colour. Let us denote T_{ALL} a set of all possible tokens, N_{ALL} a set of all possible names, B_{ALL} a set of all possible colours, and S_b a set of all possible states for a colour $b \in B_{ALL}$.

Then, we say that *colouring* is a function: $\beta : T_{ALL} \rightarrow B_{ALL}$ – it gives a token's colour – and *token–status* is a function (defined for each colour): $\sigma_b : T_{ALL} \rightarrow N_{ALL} \times S_b \times \mathbb{N}$ – it gives a token's name, state, and age; and nothing if the token in question does not have the colour b. If an S_b is empty, we say that the token is stateless. A token can be located in a container. The age stands for how long the token is located there. In Fig. 4 only the initial tokens are shown. After the start of the simulation tokens would begin to appear or could be removed.

Containers. Every *container* has a name, a type and can be associated with a set of triggers, which are actually if-then rules. Let us denote C_{ALL} a set of all possible containers, Y_{ALL} a set of all possible types and I_{ALL} a set of all possible triggers. Then, we say that *container–status* is a function: $\gamma : C_{ALL} \rightarrow N_{ALL} \times Y_{ALL} \times \mathcal{P}(I_{ALL})$ – it gives a container's name, its type, and triggers. A container can contain more than one token in a given simulation time. We say that *containing* is a function $\kappa : C_{ALL} \rightarrow \mathcal{P}(T_{ALL})$ – it provides all tokens located in a given container in a given instant.

A semantic meaning of a token in a container is a denotation either of a state of a group of actors (i.e., an actor-token) or of a satisfied general precondition (i.e., a prec-token). In a drawing, containers are denoted according to their triggers and tokens they can contain as follows:

- ○: an actor-token & without any trigger,
- ◎: an actor-token & with triggers,
- ▢: a prec-token & without any trigger,
- ▣: a prec-tokens & with triggers.

There are following main token types in the PNA-plot from Fig. 4; italic denotes the initial state:

- "magician has" (colour m); $S_m = \{noPuppet, \text{bought}, \text{stolen}, \text{given}\}$
- "magician where" (colour i); $S_i = \{inPub, \text{outPub}\}$
- "guard" (colour g); $S_g = \{notCaught, \text{caught}\}$
- "theatre troupe" (colour t); $S_t = \{\text{arriving}, \text{inPub}, \text{playing}, \text{defending}, \text{burnt}, \text{thankful}\}$
- "villagers", "robbers" (colour o); $S_o = \{\}$
- a prec-token: "has-called", "has-recognised" (colour o)

The meaning is obvious: for example, every "magician has" token represents that the magician has stolen, or has bought, or has been given a puppet, or does not have it, respectively. Similarly, the state of "magician where" stands for the magician in the pub or out of the pub. Notice, that these tokens can be located only in PLAYER container. Notice also, that a token is not a virtual human itself, it is just a representation of its state for the purpose of the story management!

Triggers. The most important primitive of the PNA-model is a *trigger*. A trigger can be associated with an action (an *action trigger*) or a *container* (a *container trigger*). Basically, a trigger is an *if-p-then-c* rule, where p is a precondition and c a consequence, which is to be performed when p holds. There are four types of triggers (both action triggers and container triggers).

- A *token-generating* trigger is a trigger that has a consequence of always adding some tokens to some containers while not removing any token.
- A *token-consuming* trigger has a consequence of always removing at least one token and possibly adding tokens to containers.
- An *action-firing* trigger neither generates nor removes any tokens, but fires an action.
- A *conflict-resolving* trigger's precondition tests whether two or more conflicting actions are to be fired at the same time, and its consequence resolves the conflict. What actually means a conflict between actions will be explained later.

Actions and triggers. Every *action* has a name, one action-firing, one token-generating and a token-consuming trigger, a "ready to fire" flag, and an effect. Essentially, firing of an action has two steps: first, the "ready to fire" flag is set, second, the action is really fired. The purpose of this separation is that there may be a conflict between several actions, and not all of them can fire.

The precondition of every token-generating trigger and every token-consuming trigger tests whether or not the flag is set. Let us denote A_{ALL} a set of all possible actions. Then, we say that *action–status* is a function $\alpha : A_{ALL} \rightarrow N_{ALL} \times \{0, 1\} \times I^3_{ALL}$ – it returns an action's name, whether or not its flag is set, and its triggers.

If an action-firing trigger holds, its consequence sets "ready to fire" flag. Then, the other two triggers can be triggered. Notice the word "can". In a given instant, more actions can be ready to fire, but not all of them can be allowed to fire for there can be a conflict between their token-consuming triggers: it is not possible to remove a token that has been just removed by another trigger. We decided to solve this "conflict issue" by introducing conflict-resolving triggers. A purpose of a conflict-resolving trigger is to detect such a conflict, its consequence is to simply unset the flag of some actions. Finally, when an action is really allowed to fire, its effect is performed, and its token-generating and token-consuming triggers are triggered. See Algorithm 1 below for details.[1]

There is no graphical primitive corresponding to a trigger directly. However, a container with a trigger is denoted as ◎ or ◎, and an action trigger (action-firing, token-generating, and token-consuming) is depicted as a set of arrows, each from a container to

[1]We remark that conflict resolving triggers present a relatively complicated element. We noticed that designers of real scenarios for Europe 2045 and Karo typically specified plots avoiding the conflicts at the first place; they almost never used these triggers. Hence, we regard the conflict resolving trigger rather as a mean for having the model theoretically coherent than a way for really solving possible conflicts among actions.

an action, or vice versa. For example, ⊙→□ denotes both an action-firing trigger and a token-consuming trigger at the same time, meaning "try to fire the action if there is at least one token in the container" and "consume one token when the action is fired". An example of a token generating trigger is □→○ (it means "generate one token when the action is fired").

Details of a precondition or a consequence of a trigger are indicated schematically next to an arrow, next to a container, or by a changed shape of the arrow. Precisely, a precondition can:

- test κ: i.e., whether some containers contain ($\rightarrow$) or do not contain ($\dashrightarrow$) some tokens; test β and σ_β: i.e., colouring ($< col? >$) and token-status ($< state? >$),
- test simulation time (HH hours),
- compare age (MM minutes) or simulation time (HH hours) to a random value generated using a given probabilistic distribution (📈),
- test whether a user will alter the simulation somehow ($will$),
- test whether a trigger has fired n times ($n\times$),
- test whether two or more actions are in a conflict ($=\!=\!=\!=\!=\!=$).

A consequence of a token-generating trigger can generate a token of a specific colour and a state. This is indicated as $< state >$, $<< X >>$ or $< col >$ above the arrow. $<< X >>$ means that the generated token has the same state as the token that has been just consumed (see, for example, GUARD IN PUB container).

Notice that not all conflicts are always solved in Step 4 – there might exist a conflict not recognised by any trigger. In this case, the action to execute will be chosen randomly from the set of the actions in the conflict (Step 6).

4.3 Examples

Now, consider several examples taken from Figure 4. First, focus on the container THEATRE and the action *Start parking* (in the pub), which is redrawn at Figure 5a. There are three triggers indicated in the drawing. The first one generates a token "theatre" with *arriving* state into the container once at about 4 p.m. (Step 7, 8 of Algorithm 1). The second one is an action-firing trigger. It sets "ready to fire" flag when the token in the container is at least 30 minutes old (Step 1, 2). The third one is a token-consuming trigger, which removes "theatre" token when the action fires (Step 5, 6).

The next example concerns with the episode of the theatrical performance (Fig. 5b). The token-generating trigger of the *Play starts* action has generated four tokens; two "has-recognised" into the RECOGNISED? container, one "villagers" into the VILLAGERS WATCHING container, and one "theatre" with the state $playing$ into the THEATRE container. The action-firing trigger of the *Struggle starts* action tests whether or not there is a token with the state $playing$ in the container THEATRE, and a token in the container VILLAGERS, and two tokens in the container RECOGNISED? with the appropriate age. It is determined randomly for the both tokens, whether or not the token is old enough. If the action is fired, the next trigger – the token-consuming one of *Struggle starts* – consumes all four tokens. Notice that all durative episodes are represented in a similar way using the age of the tokens.

Fig. 5c represents the stealing of a puppet. If the magician is in the pub (denoted as WHERE: $< in? >$), and if she does not have a puppet (denoted as HAS: $< noPuppet? >$),

Algorithm 1 Unfolding a story in TEST using a PNA-plot.

Input: a PNA-plot.
The algorithm is performed in every time step t.

1. $I_{action,t} \leftarrow$ all action-firing triggers that fire in time t
2. Perform a consequence for all $i \in I_{action,t}$ in a random order (i.e., mark the "ready to fire" flag of the respective actions)
3. $I_{conflict,t} \leftarrow$ all conflict-resolving triggers that fire in time t
4. Perform a consequence for all $i \in I_{conflict,t}$ in a random order (it unsets the "ready to fire" flag of some of the actions being in conflict; it may include asking a user for selecting such actions)
5. $I_{remove,t} \leftarrow$ all token-consuming triggers that fire in time t (they test for "ready to fire")
6. for each $i \in I_{remove,t}$ (take i in a random order) do
   - if all desired tokens can be still removed do:
     - perform the consequence of i (it includes removing the tokens)
     - if i is an action trigger do perform the effect of the action
   - otherwise, if i is an action trigger do unset the "ready to fire" flag
7. $I_{generate,t} \leftarrow$ all token generating triggers that fire in time t
8. for each $i \in I_{generate,t}$ (take i in a random order) do
   - perform the consequence of i
   - if i is an action trigger do unset "ready to fire" flag

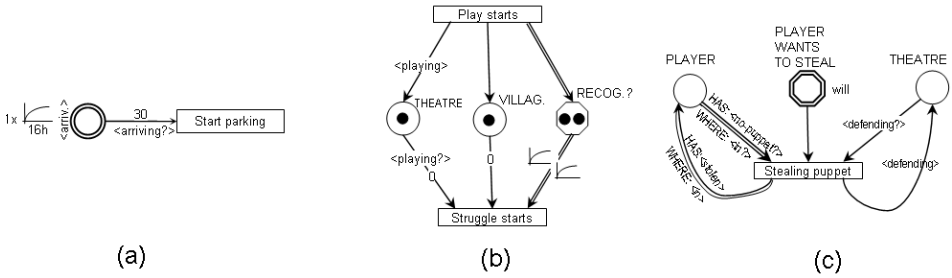

Figure 5: a) The theatre container example. b) The theatrical performance example. c) The stealing of the puppet example. Notice that the container THEATRE is depicted several times for convenience in 4.

and if she wants to steal, and if the brawl has already flared up (the state of "theatre" is defending), she can steal the puppet. Notice that the token "player wants to steal" is generated according to the choice of the user – this feature is achieved by checking a check box in the application TEST. Notice also, that from the perspective of the high-level plot, the action of stealing is instantaneous: it takes no time.

4.4 Summary

The purpose of this section was an explanatory one. We introduced in detail the PNA-model, on which we evaluated fruitfulness of Petri Nets in storytelling in general. The evaluation took place on the application TEST we developed that allowed for running PNA-plots in an abstract manner, i.e. without any virtual world connected to it (Brom and Abonyi, 2006).

Several things ought to be commented. First, notice that a story can really evolve in parallel. For example, the skipped "catching plot" part at Fig. 4 is actually unfolded at the same time as the "theatre in pub" part. Second, notice the level of abstraction of the events generated from the PNA-plot in Fig. 4. Apparently, the PNA-plot presents a high-level narrative structure; many low-level events are assumed to be generated by the simulator of the virtual world, possibly based on the user interaction (see Fig. 2 again). This gives substantial degree of freedom in possible ways of evolving the story, and this also presents advantage from the design point of view, because splitting the design part into specifying a high-level plot and low-level behaviour of virtual humans and objects allows ones to concentrate their thinking on one problem, without being distracted by the second one. Additionally, the simulator can use a different representation than the story manager, not necessarily based on Petri Nets. For example, behaviour of virtual humans can be represented by means of hierarchical if-then rules (Bryson, 2001; Brom, 2005), or BDI (Bratman, 1987). The BDI is actually the case of Karo.

Third, there is no conflict-resolving trigger in the PNA-plot discussed: the reason for this is that conflicts were intentionally avoided during the design of the plot. Forth, all the properties that are only indicated in the portrayal, such as probabilistic distributions, must be specified precisely in the xml-file corresponding to the plot. It is the job of the programmer to create this xml-file, and not the designer's. This precise xml representation is to be created from a Petri Nets drawing of a designer and based on his or her supplementary text.

In the next two sections, we overview the modifications of Petri Nets used in Europe 2045 and Karo. In Section 7, a more general discussion follows.

5 Europe 2045

Europe 2045 is an on-line multiplayer game aimed at education of high-school students in economics, politics, and media studies (Brom et al., 2007b). The developmental phase is completed and the game is presently being evaluated. Seven preliminary studies have been already carried out, each involving about 10 high-school or undergraduate university students. A pilot evaluation has been conducted on two groups of 34 high-school students (19 females, 15 males) in Prague, the Czech Republic, in January 2008. The game is intended to be fully applied in spring 2008.

Europe 2045 contains economical and social model, which simulates population aging, migration, evolution of the market, transfers of industry and services, changes in environment, moods of citizens, and a substantial number of other variables describing

particular states and European Union as a whole (e.g. culture, infrastructure, education, etc.). The game features three layers of game-play. Each student (1) represents one EU member state in the game and is responsible for its governmental policies, economical development, and social issues. Basically, the player governs its state by deciding on the budget's priorities. Additionally, in cooperation with other players, (2) he or she is engaged in setting politics of the whole EU. Nevertheless, the essential feature of the game is that (3) each player faces various simulated scenarios and crises addressing contemporary key issues of the unified Europe, including migration, international relations, and energy independence.

The narrative structure of Europe 2045 serves for three purposes. First, it introduces new topics and agenda for students' discussions. Second, unfolding new events in accordance with players' previous decisions, it serves as a global feedback for the students and as a method for sharpening the discussion. Both these kinds of events are global, i.e. they are common for all the players and concern EU and international issues (e.g. conflict in Darfur has intensified). The third class of events provides individual players with a feedback about the results of their previous actions concerning their own states; hence, these events are local (e.g. citizens in France protest against university fees, or unemployment in Czech Republic has reached 15%).

The game proceeds in rounds, one round is one game year. An atomic "beat" of a scenario is called an *affair*. It is an event that takes place in one round and can be triggered by players' actions or results from the economical and social model or affairs from previous rounds. An affair is communicated to the player via a textual description in the game newspaper or via a short animation in TV, which is being displayed at the beginning of every round (*news item*). In some cases, an affair also has an impact on the economical and social model, i.e. it influences state of a country or the whole EU. Typically, an affair can result in increasing the EU budget, increasing the level of pollution in particular states, crippling agriculture production, etc.

Some affairs introduce issues that require decision to be taken by the players (e.g. accepting another state's proposal, sending humanitarian mission to the area of a conflict, etc.). These decisions are intended to be taken during a discussion, typically in the class under the teacher's supervision, and voted through a *ballot*. One affair often triggers more ballots, each constituting precisely formulated question ("Do you vote for sending European humanitarian mission to Darfur area?") with three possible answers (yes/no/abstain). The ballots chosen by the game designers aim to cover all the main possible solutions usually proposed by real politics in similar cases. When the answers can not be schematised to the yes or no option, the ballot contains number (3-4) of more detailed solutions. The decision chosen by the players influences the economical and social model and the affairs to be triggered in the next round.

The game offers more different campaigns to be played, each of them focusing on different problematic (e.g. energy independence, international relations, environment). For each campaign, specific affairs and a scenario describing relations between them have to be designed. New campaign also comprises distinctive animations for the TV, articles for the newspaper, items for the in-game encyclopaedia, teachers manual, and handouts for students.

The aim of this section is to describe how Petri Nets were modified for the storytelling purposes in Europe 2045, i.e. to introduce the PNB-model. Also an example of a particular plot (i.e. a PNB-plot) concerning with "Darfur crisis" scenario will be introduced.

Figure 6: Screenshots from Europe 2045. left) The interface, through which the player governs its country. right) The balloting interface.

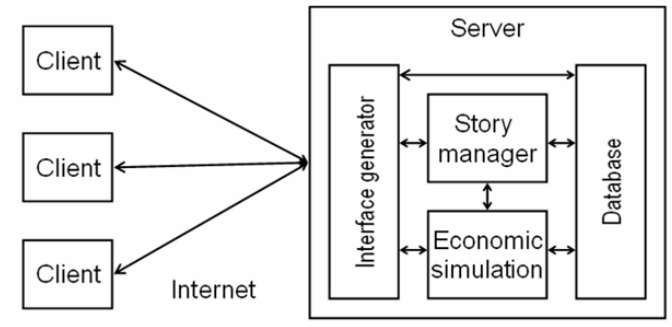

Figure 7: Architecture of Europe 2045. Notice, how it mimics the abstract architecture from Fig. 2.

5.1 Technical background of Europe 2045

Technically, the game is a client-server application; the students play the game via the Internet (Fig. 7). The server part comprises PHP scripts generating the game interface, the story manager, which is written in PHP as well, and the social-economical simulation, which is written in Java. Almost all parts of the interface are programmed in Flash (see Fig. 6). The social-economical simulation features a simplified model of EU economy. Basically, at the end of each round, this component computes the next state of each country (i.e. will the quality of infrastructure increase or decrease?), and carries out decisions in which country to build new factories, mines etc. based on particular variables of the countries (e.g. environmental tax, quality of environment, how well the infrastructure is developed etc.). We remark that this simulation does not feature human-like agents.

5.2 PNB-model

The PNB-model is stemming from the PNA-model. It was designed to mirror the features of stories of Europe 2045 described above. Particularly, the model must have seized the round-based nature of the game, the affairs, the ballots, and the presentation of news items. We now describe individual components of the model. Then, we describe how they are integrated together and introduce the story manager algorithm.

Actions. The model works with two types of actions: *affairs* (▭), and *news items* (⋯). Both of them can be started between two rounds by a trigger, as described later. When an affair is triggered, it can influence the game by its *game impact* (this is similar to an effect of an action of the PNA-model). An impact can be for example: "migration to EU decreases in this round by x". The result of the impact is computed by the economic simulator before the next round starts (see Algorithm 2). Contrary to affairs, news items never have a game impact.

Both kinds of actions can communicate to the player several news items in the next round, using either the game newspaper, or the game TV. Only an affair can invoke (immediately) one or more ballots about a proposal related to the affair.

When the next round finishes, both affairs and news items can generate a new token to a specific container (this was a job for token-generating triggers in the PNA-model; both approaches are equivalent). Fig. 8a depicts an action, which generates one token. In Fig. 8b, there is depicted an affair "Darfur conflict" invoking a ballot about "sending an EU mission to Darfur" proposal (which in case of agreement generates two tokens, each to a particular container – see next).

Ballots. In a typical *ballot* (⬭), each student has three possibilities: he or she can agree, disagree or abstain, but more complicated cases are also allowed. To determine how a result of a ballot influences the game (i.e. which game impact to apply), a *what-next function* is defined for each ballot. Basically, for each return value of a what-next function we define a *game impact* (this is again similar to an effect of an action of the PNA-model). Additionally, every ballot can generate one or more tokens similarly to actions based on on the result of the what-next function. More detailed description on what-next function element can be found in Brom et al. (2007b).

Tokens. We employ state-less aging tokens, in contrast to the PNA-model. A token starts to age after it is generated to a container, but not removed in the next round. The age is given in the number of rounds the token stays in the container.

Triggers & Containers. Triggers are evaluated at the beginning of each round. The PNB-model features three types of them. First, we have *token-generating* triggers. These are associated with containers, but not with actions as in the case of the PNA-model. Every time the condition of such a trigger holds, a new token is generated into the container. In Europe 2045, we use only the form of "when the round x starts $\rightarrow$ generate one token". In fact, these triggers start the game or schedule the episodes to particular rounds. A container with a trigger is depicted as ◎, while a container without a trigger as ○.

Second type of triggers is an *action-starting* trigger, that joins job of action-firing and token-removing triggers of the PNA-model (this was regarded as more intuitive for designers). As we already know, a conflict between two actions may occur (Fig. 8c). Hence, this trigger only marks its action to be started, and the respective tokens to be removed. A typical action-starting trigger has the following form: "if there are n tokens in a particular container & $f \rightarrow$ mark the desired action", where f is a function of game state returning a boolean value (e.g. f can be "is EU budget in this round bigger then x EUR?").

To cope with conflicts, we define *conflict resolving* triggers that unmark one of the marked actions in the same manner as in the PNA-model. In fact, they have not been used in real scenarios, remaining a mechanism making the PNB-model theoretically coherent.

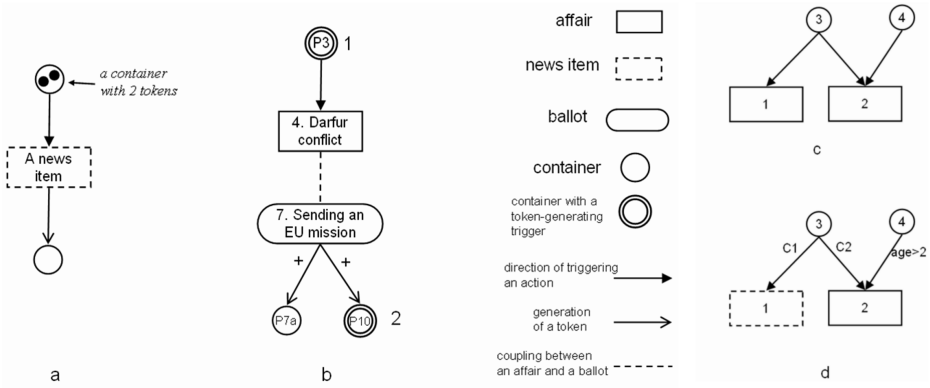

Figure 8: a) A simple net. One token is needed in the upper container to start the news item, which then generates a token to the bottom container. b) A token-generating trigger generates a token to Container P3 in the first round (notice 1 next to P3). In fact, this starts the scenario. Affair 4 invokes Ballot 7 immediately. Between round 1 and 2 and if the what-next function of Ballot 7 returns "+" (which is in this case when the ballot proposal has been agreed), two tokens are generated; one to P7a, one to P10. Additionally, a token-generating trigger generates one token into P10 in the second round regardless of the result of the ballot. c) If there is both one token in Container 3 and one token in 4, it is not clear whether to start Action 1 or 2. A conflict resolving trigger must be invoked. d) News item 1 is started when Condition C1 holds. Affair 2 is started when condition C2 holds and there is at least one token in Container 4 older than 2 rounds. Note, that in fact, the overall condition of trigger starting Affair 2 is "C2 & *age_of_token_in_Cont4* > 2". If there is a conflict between this trigger and the trigger starting News Item 1, the conflict-resolving trigger is invoked as in the case (c).

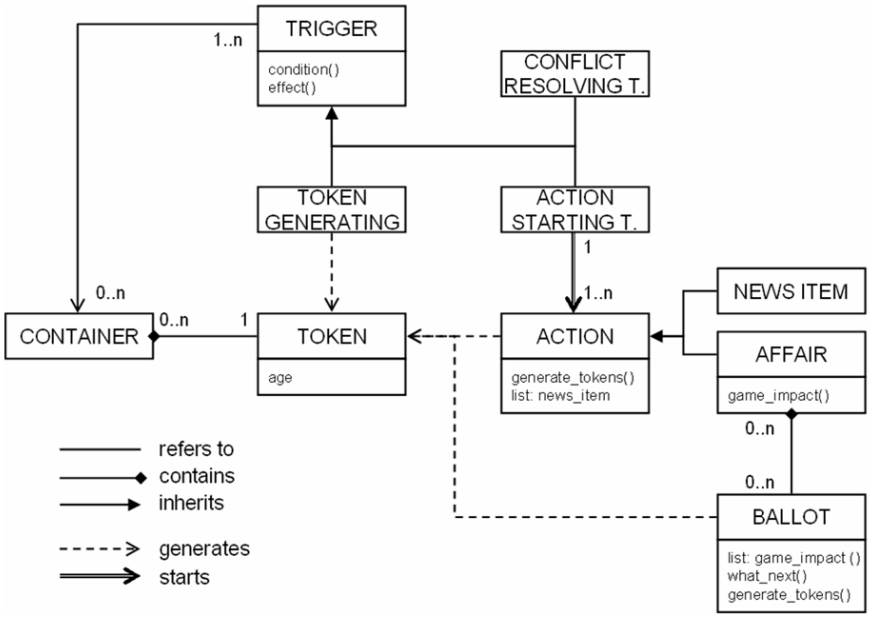

Figure 9: Overall description of PNB-model given in a UML-like class diagram. Based on this schema, the story manager in Europe 2045 works. Relations of the "conflict resolving trigger" are not depicted for clarity.

To sum up; in the PNB-model, we use tokens with aging, but without states (contrary to PNA-model), we use two types of actions (news items and affairs), and we can couple an affair with one or more ballots. We use three types of triggers, which can have relatively complicated conditions. Tokens can be generated by ballots, actions, and token-generating trigger.

A game designer specifies the plots graphically at the first place. Additionally, most entities of the model have a unique ID, which serves as a referencing mechanism to additional textual description (concerning e.g. conditions of triggers) as well as corresponding animations and news articles.

During a game, a particular PNB-plot is evaluated between each two consecutive rounds using Algorithm 2. Theoretically, some difficulties may be encountered in marking of tokens in Step 2 (for each trigger, we need to mark the tokens of appropriate age that has not been already marked, which may require searching for an appropriate ordering of triggers). Additionally, ordering of actions in Step 4b may matter (the total impact of "action A after action B" can differ from the total impact of "action B after action A"). Though these issues are of theoretical interest, we have not tackled them rigorously for they could be addressed easily in an *ad hoc* manner even for our largest scenarios (Fig. 10).

Fig. 9 overviews how all the components of the model are integrated together.

5.3 PNB-plot example: Darfur crisis

One of the largest and perhaps the most informative plots of Europe 2045 is Darfur scenario (Fig. 10). For brevity, we will not describe here the whole plot, rather we will illustrate how the building blocks of the PNB-model work on the plot.

Algorithm 2 The algorithm of the story manager in Europe 2045.

Notice, that the steps 1-5 are concerned with a round k, while 7-10 with the round $k + 1$.

1. Evaluate all token-generating triggers and generate tokens based on the results.
2. Evaluate all action-starting triggers and mark the respective actions to be started and the tokens to be removed.
3. For every token that is marked more than once: trigger the appropriate conflict-resolving trigger and consequently unmark one or more actions.
4. If:
   (a) there is no marked action, then end this scenario.
   (b) Otherwise, prepare all the marked actions to be run: generate newspaper & TV news based on the news items, consider the impact of the affairs, prepare the ballots of the affairs.
5. Remove all marked tokens, increase the age of remaining tokens.
6. New round:
   (a) Run new round. Display the news, and ballots, take the input from the user...
   (b) End the round.
7. Calculate the results of the ballots & their what-next functions.
8. Generate new tokens based on the run actions and on the what-next functions of the ballots.
9. Compute the game impacts of the ballots based in their what-next function.
10. Go to 1

This scenario starts in the first round by an affair communicating via TV that the crisis in Darfur has escalated and invoking four ballot proposals. Based on the results of the ballots, the crises further develop. The important point is that it can evolve in several branches at the same time. For example, if students agree both on a form of development aid (Ballot 8) and a humanitarian aid (Ballot 5), both the affairs "Development aid begins" (8a) as well as "Humanitarian aid begins" (5a) are triggered in the second round. Additionally, either Affair 10a "Big migration increase", or Affair 10b "Small migration increase" is started. Which affair is started depends on conditions 7A and 7B.

Notice that all elements have their ID referring for further textual description. For example, it is specified in this way that Condition 7A, i.e. the condition of the trigger starting Action 10a "Big migration increase", is "if there are less than 3 tokens in P10", while the trigger starting 10b wants "at least 3 tokens" (Condition 7B). Notice also, that Container P10 has a token-generating trigger, which generates one token into P10 in the second round. This means that even if all the proposals are disagreed, large increase in migration to EU still occurs.

Similarly, it is specified textually that in the ballot "Lost reaction" (12), students have three possibilities: to reinforce the mission, to pull out the mission or to ask NATO for help. To which container a token will be generated depends on the result of the ballot as specified by Conditions 12A, 12B, and 12C. It is also specified that Ballot 12 has no game impact, but the affair "Mission failed" (20) has the game impact: "migration to EU is increased by x".

The scenario ends in the fourth round by a final series of affairs. However, notice, that other scenarios can be unfolded in parallel. In the basic campaign of Europe 2045, there are two additional scenarios of the size of "Darfur", and several dozens of small scenarios comprising from one to three actions. The whole campaign lasts 10 rounds (game years).

5.4 Summary

We extended the preliminary model described in Section 4 by presenting the PNB-model, which was used as a plot specification technique in the full-fledged serious game Europe 2045. Presently, this game features one 10-round, fully developed campaign, which comprises about 70 game events (affairs, or news items). This allows us to conclude that in this section, we demonstrated fruitfulness of Petri Nets in the domain of social-economical serious games.

The plot of the mentioned campaign was actually coded directly in PHP. This helped us to start quickly, but it also presents a limitation. To facilitate the development process, we would benefit from a graphical authoring tool, especially because we aimed at creating a second campaign and several undergraduate humanities students, who do not know PHP, develop other campaigns as a part of their university course. Developing this tool presents our future work.

6 Karo

Karo is a simulation featuring virtual humans intended as a serious game for civic. Particularly, it should help students to understand dynamic of social relationships. Presently, it should be regarded as a research prototype, not a full-fledged application. Contrary to Europe 2045, programming work on Karo still continues. Speaking in terms of the three-layered metaphor introduced in Sec 2.1, the GUI, the simulator as well as the story manager are finished. However, only one case-study scenario (i.e. the specification of the

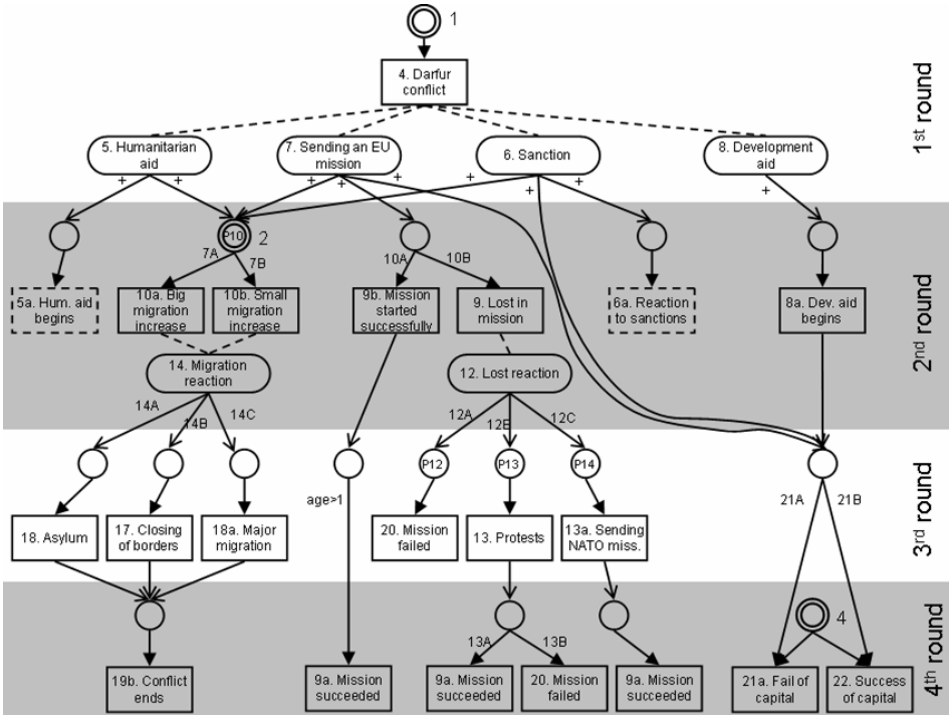

Figure 10: Darfur scenario. The elements of the plot are organised according rounds for clarity. Container's ids that are not referred from text above are not depicted. For clarity, several actions are depicted twice (9a, 20).

plot and its setting) is implemented. Moreover, this scenario has no educational aspect. Nevertheless, it showed that Petri Nets can be really scaled as a storytelling technique for a simulation employing virtual humans. After sketching technical background of Karo, we describe the case-study scenario and review the differences between the PNA-model (Sec. 4) and the Petri Nets modification used in Karo, the PNC-model.

6.1 Technical background of Karo

Technically, Karo is built upon Java framework IVE – intelligent virtual environment – we developed formerly (Brom et al., 2006), which serves as a simulator of virtual worlds. The IVE features grid worlds only; however, this has been regarded as sufficient for our purpose. The action selection mechanism of virtual humans is based on BDI architecture (Bratman, 1987), that is implemented using fuzzy if-then rules. The IVE was intentionally designed for simulations of large virtual environments inhabited by tens of virtual humans. It uses level-of-detail technique for simplifying space and behaviour of objects and virtual humans at the places unimportant at a given instant (Brom et al., 2007a).

6.2 Narrative in Karo

The narrative in Karo is partly emergent and parallel, very character-oriented and goes into the detail. Our case-study story contains many actors; some of them being background characters completely driven by underlying world simulator and never directly influencing the story (e.g. inhabitants of a village, who seem to have their own lives). On the other hand, behaviour of main characters can be influenced by the story manager.

We decided to solve "interactive–narrative" tension by limiting the degree of interactivity available to a user. We allow the user to participate only by changing emotions, moods, and relationships of the characters, and by modifying the environment slightly, e.g. by moving objects. Nevertheless, these small influences can change the outcome of the story radically. Our working hypothesis is that this approach has a strong educational potential because it allows a student to perceive that small, seemingly similar actions can have totally different outcome; however, this issue remains to be investigated.

The case study scenario is a dummy fantasy set taking place in a fictional medieval town, and in the surrounding rural area. It features seven main characters, and several background actors.

In the beginning, Anne Greyfox, a maiden, who is the daughter of a local baker, Bryant, is visited by her friend, Nerys Robertson. After a short conversation Anne and Nerys decide to go to the Lonely Tree, a romantic place outside the town. Possibly they are warned about the place by Bryant, which may influence their caution later.

The girls go through the town, finally coming to the house of Aunt Dawson, a good-hearted old woman. She asks them if they can take a lunch to her son Timmy, a city guard. After they agree, Jonas, Nerys' little brother, comes here. He is a spoiled child, fattish and rude. He tells the girls that he is coming with them. Nerys, depending on her mood, either ignores him or forbids him to follow them. He, however, heard where the girls are going, so if he is not allowed to follow them, he decides to go to the Lonely Tree on his own. He does not know that they are going to Timmy's watch post first, which is quite a detour, so he arrives to the Lonely Tree sooner.

Near the Lonely Tree, local villain and murderer, Rob The Robber, is hiding. When the girls arrive, they see him and naturally he decides to kill them. In the end, the girls are either alive, and Rob the Robber is dead, or other way round, and also Jonas is either alive or dead, depending on how the user was influencing the story.

6.3 PNC-model

The PNC-model is an extension of the PNA-model. Apart from minor modifications, there are two significant extensions. First, triggers can now test state of the world using a general pattern matching algorithm. This feature is actually similar to the function f introduced into the PNB-model (see Sec. 5.2, Triggers & Containers). Hence, a designer of a plot can specify a condition like "someone is near Jane" with the effect of "putting token to container X".

Second, because graphical representations of plots became quite large, larger than the ones on Fig. 4 and 10, we introduced nesting of Petri Nets. This has been achieved through adding opening and closing triggers, which based on the state of the world and on the state of an open net can open or close a subnet. A subnet can also contain opening and closing triggers, a method for having deep hierarchies. This mechanism not only simplifies design, but also speeds up evaluation, because the closed subnets are not evaluated. It also prevents a designer to do unintended mistakes by using entities that are not related with the current part of the plot.

6.4 Summary

We extended the PNA-model described in Section 4 by presenting the PNC-model, which was used for developing a case-study scenario in Karo game. The main innovation in the PNC-model is hierarchical nesting of Petri Nets.

While this work is still in progress concerning its educational side and particularly the claims that "small influences can change the outcome of the story radically" and that this "has an educational potential", the experience we gained so far allows as to state that Petri Nets can be and in fact have been scaled for unfolding stories in an application featuring virtual humans.

7 Discussion & Conclusion

In this article, we have described a method for specifying branching plots of scenarios that can evolve in parallel, i.e. in several different places at the same time. The method is specifically designed to cope with stories being unfolded in large virtual worlds. The method exploits Petri Nets, which are a technique relatively novel in the field of storytelling (but see Natkin (2003); Delmas et al. (2007)).

The article proceeded as follows: after discussing related work, we have presented a formal model for representation of plots in Section 4, so called the PNA-model, and have introduced an application that allows for prototyping of these plots and evolving stories in an abstract manner, i.e. without any virtual world simulator. Then, in Section 5 and 6, we have extended this work by presenting the PNB-model and the PNC-model, which have been used and are being used in serious games Europe 2045 and Karo respectively. While the PNB-model have demonstrated the practical usefulness of Petri Nets as a plot representational technique in the domain of serious games, the PNC-model have showed their scalability for an application featuring virtual humans.

Apart from the ability to represent branching plots that can evolve in parallel, the strength of Petri Nets is that they are formal and yet can be presented graphically. We have found them as easily comprehensible by non-IT experts; not only the chief designer of Europe 2045 does not have IT background, but the technique was also explained to college students of humanities during a course (during about an hour and half) and they

were subsequently able to use it to specify their own "toy" campaigns. (However, we have not carried out a "control test" with another technique; the HTN planning might be a good candidate.)

Since the Petri Nets have all these strengths at the same time, we favoured them over deterministic finite state machines (Sheldon, 2004), beat-approach of Mateas (2002), and HTN planning formalism (e.g., Cavazza, 2002).

The question remains, what are the limitations of the technique. At the beginning, it must be acknowledged that Petri Nets fit well only for stories that are preset. We also think that the beat approach is better for controlling stories featuring relatively small virtual worlds inhabited by (a few) virtual humans – we regard this approach as more flexible in this domain. Additionally, if one needs a story based on a preset plot, which however do not evolve in parallel, a deterministic FSM would likely be sufficient.

It must be also stressed that Petri Nets is a branching technique: after all, the author must specify all the branches in advance. The potential risk of combinatorial explosion of branches must be avoided by manual "cutting" by the author. It is intriguing to think whether this problem can be overcame, perhaps by generating parts of Petri Nets automatically, hence empowering the technique with some ideas being exploited in the planning domain (e.g., Reidl and Young, 2006). This is, however, a pie in the sky.

Acknowledgements

The research on usage of Petri Nets in storytelling was partially supported by the Program "Information Society" under project 1ET100300517, and by the Ministry of Education of the Czech Republic (Res. Project MSM0021620838). The project "Integration of IT Tools into Education of Humanities" is financed by the European Social Fund, the state budget of the Czech Republic, and by the budget of Municipal House Prague. The authors would like to thank to all the partners of the project: Generation Europe, Ciant, gymnasium Sázavská, and Association for International Affairs, and to all the people who helped, most notably to Petr, Edita, Jakub, Lenka, Ondřej, Martin, and Michal.

References

Aylett, R. S. 2000, 'Emergent Narrative, Social Immersion and Storification' in *Proceedings of 1st Narrative Interaction for Learning Environments*, Edinburgh.

Aylett R.S., Louchart S., Dias J., Paiva A., Vala M. 2005, 'FearNot! – An Experiment in Emergent Narrative' in *Proceedings of Intelligent Virtual Agents*, LNAI 3661, Springer, pp 305–316.

Bratman, M. E. 1987, *Intention, plans, and practical reason*, Cambridge, Mass: Harvard University Press.

Brom, C. 2005, 'Hierarchical Reactive Planning: Where is its limit?' in *Proceedings of Modelling Natural Action Selection*, Edinburgh, Scotland, pp 235 – 242.

Brom, C., Abonyi A. 2006, 'Petri-Nets for Game Plot' in *Proceedings of AISB Artificial Intelligence and Simulation Behaviour Convention*, Bristol, Vol. 3, pp 6–13.

Brom, C., Lukavský J., Šerý O., Poch T., Šafrata P. 2006, 'Affordances and level-of-detail AI for virtual humans', in *Proceedings of Game Set and Match 2*, The Netherlands, Delft.

Brom, C., Šerý, O., Poch, T. 2007a, 'Simulation Level of Detail for Virtual Humans' in *Proceedings of 7th International Conference on Inteligent Virtual Humans*, LNCS Vol. 4722. Paris, France. Springer-Verlag, Berlin, pp 1–14.

Brom, C., Šisler, V., Holan, T. 2007b, 'Story Manager in Europe 2045 Uses Petri Nets' in *Proc. of 4th International Conference on Virtual Sotrytelling*, Springer-Verlag (to appear).

Bryson, J. 2001, *Intelligence by Design: Principles of Modularity and Coordination for Engineering Complex Adaptive Agents*, PhD thesis, Massachusetts Institute of Technology.

Cavazza M., Charles F., Mead S. J. 2002, 'Planning Characters' Behaviour in Interactive Storytelling' in *The Journal of Visualization and Computer Animation*, Vol. 13, pp 121–131.

de Freitas, S. 2006, 'Learning in Immersive worlds: A review of game-based learning', *JISC (Joint Informational Systems Committee) report*, [Online] Available at: `http://www.jisc.ac.uk/eli_outcomes.html`

Delmas, G., Champagnat, R., Augeraud, M. 2007, 'Plot Monitoring for Interactive Narrative Games' in *Proc. of ACE 07*, Salzburg, Austria, pp 17–20.

The Education Arcade: Revolution, a role-playing game 2007, [Online] Available at: `http://www.educationarcade.org/revolution`

Egenfeldt-Nielsen, S. 2005, *Beyond Edutainment: Exploring the Educational Potential of Computer Games*, PhD Thesis, University of Copehagen.

Egenfeldt-Nielsen, S., Buch, T. 2006, 'The learning effect of 'Global Conflicts: Middle East" in *Gaming Realities: A Challenge for Digital Culture*, eds. M. Santorineos, N. Dimitriadi, Athens: Fournos, pp 93–97.

Francis, R. 2008, *Revolution: Student's experiences of virtual role play within a virtual reconstruction of 18th century colonial Williamsburg*, an unpublished manuscript.

Generation Europe: Europe 2045, a multi-player strategy game 2007, [Online] Available at: `http://europe2045.eu`

Gordon, A. 2004, 'Authoring branching storylines for training applications' in *Proceedings of the 6th international conference on Learning sciences table of contents*, pp 230–237.

Louchart, S., Aylett, R. 2003, 'Solving the narrative paradox in VEs - lessons from RPGs' in *Intelligent Virtual Agents*, eds. T. Rist, R. Aylett, D. Ballin, 4th International Workshop IVA 2003, LNAI 2792 Springer 2003, ISBN: 3-540-20003-7, pp 244–248.

Magerko, B. 2006, 'Intelligent Story Direction in the Interactive Drama Architecture' in *AI Game Wisdom III*, pp 583–596.

Mateas, M. 2002, *Interactive Drama, Art and Artificial Intelligence*, Ph.D. Dissertation. Department of Computer Science, Carnegie Mellon University.

Natkin, S., Vega, L. 2003, 'Petri Net Modelling for the Analysis of the Ordering of Actions in Computer Games' in *Proceedings of Game-ON*, pp 82–92.

Petri Nets World: Petri Nets World, a web collection 2007, [Online] Available at: `http://www.informatik.uni-hamburg.de/TGI/PetriNets/`

Reidl, M. O., Stern, A. 2006, 'Believable agents and Intelligent Story Adaptation for Interactive Storytelling' in *Proceedings of TIDSE*, LNCS 4326, Darmstadt, Germany, Springer-Verlag, pp 1–12.

Riedl, M., and Young, R. M. 2006, 'From Linear Story Generation to Branching Story Graphs' in *The IEEE Journal of Computer Graphics and Applications*, May/June 2006, pp 23–31.

Sandford, R., Ulicsak, M., Facer, K., Rudd, T. 2007, *Teaching with Games. Using commercial off-the-shelf computer games in formal education*, Futurelab, Bristol, UK, [Online] Available at: `www.futurelab.org.uk/download/pdfs/research/TWG_report.pdf`

Sheldon, L. 2004, *Character Development and Storytelling*, Chapters 7 and 14, Thompson Course Technology.

Silva, A., Raimundo, G., Paiva, A. 2003, 'Tell Me That Bit Again... Bringing Interactivity to a Virtual Storyteller' in *The Proceedings of Virtual Storytelling II*, Springer-Verlag, pp 146–155.

Squire, K. 2004, *Replaying history: Learning World History through playing Civilization III*, PhD thesis, Indiana University.

Wooldridge, M. 2002, *An Introduction to MultiAgent Systems*, John Wiley & Sons.