A Proposal for an Agent Architecture
for Proactive Persistent Non Player Characters

Brian Mac Namee, Pádraig Cunningham

Department of Computer Science,
Trinity College, Dublin 2, Ireland.

Abstract

In the past, games development has been driven by the need to achieve more realistic graphics. This has resulted in a situation today with games set in visually stunning, and almost photo realistic worlds being the norm. As game developers seek something new to make their games stand out from the rest, artificial intelligence is playing a far more important role in game design than before. The most obvious application of AI to games is in creating Non Player Characters (NPCs). Although visually stunning worlds have been created for computer games, the NPCs inhabiting these worlds are most often shallow and uninteresting. This paper will introduce the idea of Proactive Persistent NPCs, that is NPCs that are always modelled (at least to some extent) even when the human player is not in their vicinity. We will consider the issues involved in creating these persistent characters, and propose an intelligent agent architecture which will allow the creation of realistic, interesting and robust characters.

1. Introduction

The computer games industry is now bigger than the film industry **[16]**. Until recently, technology in games was driven by a desire to achieve real time photo realistic graphics. To a large extent, this has now been achieved. At least, it will no longer be huge leaps in graphics technology that make one game stand out in the manner that Doom (www.idsoftware.com) stood out when it was first released.

This leaves the stage open for another aspect of gaming technology to move to the forefront. One of the real contenders for this role is AI. Although graphics technology allows the creation of games in which the environment looks incredibly realistic, the behaviour of computer controlled characters (referred to as Non Player Characters (NPCs)) often leads to a shallow, and unfulfilling game experience. The application of sophisticated AI techniques to the control of NPCs could rectify this situation and create more immersive gaming experiences.

As part of a larger research effort seeking to achieve this application of sophisticated AI techniques, this work will identify a potential intelligent agent architecture for use in Interactive Entertainment (IE) applications, particularly computer games. The key goal in this piece of work is to devise an agent architecture which can be used to create more interesting NPCs for such applications.

This paper will proceed by introducing the concept of Proactive Persistent Agents (PPAs) and outlining how they differ from the techniques used to create NPCs in current games. Also, the motivations for using PPAs in IE applications will be discussed.

Following this, some of the challenges unique to designing agents for this area will be discussed and the most relevant fields of current agent research will be identified.

The next section will discuss the concept of intelligent agents and elaborate upon a range of possible agent architectures. These initially break down into three main streams, reactive agents, deliberative agents and hybrid agents (a mix between the two). Each of these will be outlined in some detail and a number of more exotic architectures will also be identified.

Finally, an architecture for agents based in IE environments will be proposed. Some of the advantages and future challenges of this proposed architecture will also be discussed.

2. Motivations

> *"don't let monsters outside of the player's PVS* [Potentially Visible Set] *act up, or most of the interesting things will happen before the player gets there"*
>
> **Code Comment from Half-Life SDK**

The above quote comes from a code comment in the Half-Life Systems Development Kit[1]. It represents an attitude behind a lot of current game design. The essence of this is that games are designed only to do interesting things in the areas of the game environment in which the player is currently active[2]. This approach is quite acceptable for a lot of games. In action games in particular, which are heavily scripted and event driven, it would be pointless to have too much going on before the player arrives at a location. However there are a range of games, particularly adventure games and Role Playing Games (RPGs), in which a virtual world, populated by agents which act autonomously - irrespective of the player's location - would greatly add to the sense of immersion.

One might well ask, do today's games not already offer this kind of environment? RPG and adventure games on the market today do have a vast range of NPCs with which the player can interact. However, these NPCs simply wait for the player to arrive at their location and then, either await the player's involving them in some interaction, or play through a scripted sequence. This kind of character could be considered a *static agent*, and will be denoted as such from now on.

What this work proposes is a virtual world populated by *Proactive Persistent Agents (PPAs)*. These are computer controlled characters with needs, beliefs and desires of their own. From these, would arise goals which the characters would pursue of their own accord and in real time. The persistence of these agents refers to the fact that they are always modelled, at least to some extent, regardless of the player's location in the game world.

To illustrate the need for PPAs, consider the situation in which a game world is made up of 4 rooms, inhabited by 3 NPCs and a player character. Figure 1 shows a plan of such a world in its initial state. Now consider that the player moves around the world for a period of time, as do some of the NPCs. Figure 2 shows how the positions of the characters change over time.

[1] The Half Life Systems Development Kit can be found at http://www.planethalflife.com/half-life/files/

[2] This is an interesting version of the old philosophical question "Does a tree falling in the forest make a sound if there is no-one there to hear it?" In reality most people would believe it would, whereas in current computer games it most definitely would not!

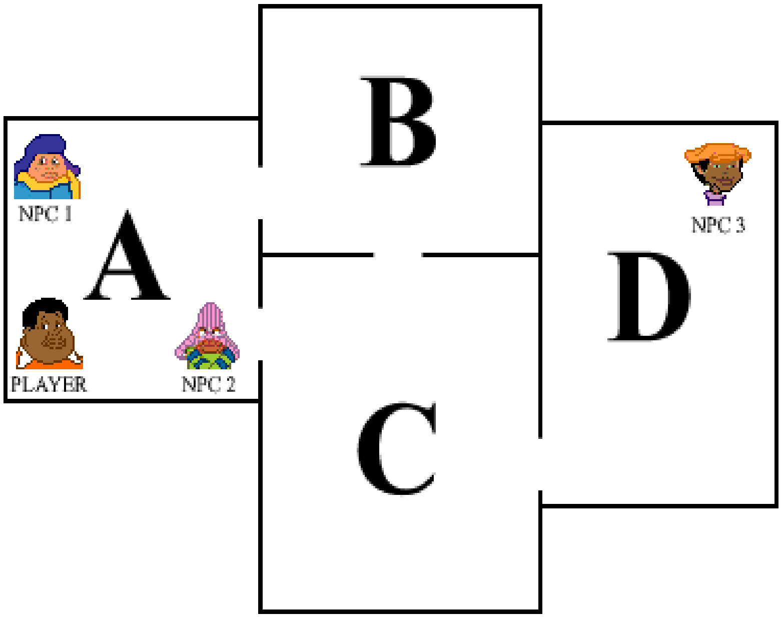
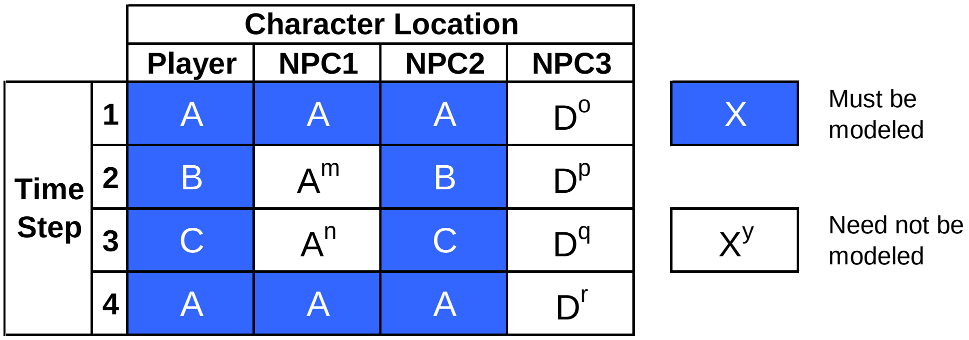


Figure 1 An example game world containing 4 rooms - A, B, C and D - inhabited by four characters - PLAYER, NPC1, NPC2, and NPC3.

If this world were to be implemented using static agents then only the NPCs that are in the same room as the player character need be modelled. These are shown as the shaded squares in Figure 2.



		Character Location			
		Player	NPC1	NPC2	NPC3
Time Step	1	A	A	A	D^o
	2	B	A^m	B	D^p
	3	C	A^n	C	D^q
	4	A	A	A	D^r

X	Must be modeled
X^y	Need not be modeled

Figure 2 The positions of the characters in the game world as they change over time. Situations in which an NPC must be modelled, under a static agent scheme, are shaded.

However a difficulty arises with this scheme if we consider the case of NPC1. During the first time period NPC1 is in the same room as the player character, room A, and thus is modelled. However, during the second and third time periods the player character moves on to other rooms while NPC1 remains in room A. For these time periods then, NPC1 would not be modelled.

For the fourth time period the player returns to room A and so NPC1 is modelled again. An anomaly arises in that there will be no model of what NPC1 was doing during the time periods two and three - A^m and A^n. Script-writers consider the moment before, and the moment after a scene crucial. Motivations bringing a character to the point at which they appear in a scene must be established, as must the character’s purpose for leaving any scene. The static agent model ignores these two notions completely. Maintaining PPA models, on the other hand, will allow for continuous modelling of NPCs and so solve this problem.

Similarly, even though in the situation shown the player never encounters NPC3, some model must be maintained so as to maintain realism. The player may well encounter NPC3 in later time periods and a model will be required. The key goal here then is determining how much must be modelled in order to maintain realism and believability.

To further illustrate the difference between NPCs implemented as static agents and those implemented as PPAs one can consider the following properties of agency as given by **[33]**:

- *autonomy*: agents operate without the direct intervention of humans or others and have some kind of control over their actions and internal state.
- *social ability*: agents interact with each other (and possibly humans) via some kind of agent communication language.
- *reactivity*: agents perceive their environment, and respond in a timely fashion to changes that occur in it.
- *pro*-activeness: agents do not simply act in response to their environment, they are able to exhibit goal-directed behaviour by taking the initiative.

NPCs modelled as static agents can be said to fulfil the first three[3] of these properties. It is the property of pro-activeness which is unique to PPAs, the key difference being their ability to "take the initiative".

A number of attempts have been made to design NPCs for computer games using agent based techniques – both in the academic sphere and in the gaming industry. Some of the more interesting academic projects include the Excalibur Project **[20],** the Oz Project **[18]** and work being undertaken at the Institute for Learning Sciences, Northwestern University, IL, USA **[7]**, **[13]**. Interesting commercial games in which interesting NPCs have been used include The Sims (www.thesims.com), Sentient (www.psygnosis.com) and Black and White (www.lionhead.co.uk).

One property of agency which has been, to a large extent, ignored by these systems is that of social ability. It is suggested in **[26]** that, as they are made today, computer games are quite limited in their appeal. For this to change a shift will have to occur towards what are referred to as "character centric" games. Such games place a much stronger emphasis on character interactions as the main focus of game play.

It is important to note that when discussing IE applications the concept of social ability is quite different to what is implied by this property in most other spheres of agent research. While in most agent research social ability refers to the ability of agents to communicate with other agents, usually in an attempt to share knowledge or to create multi-agent plans; in IE applications the property of social ability is more aligned with what is considered social ability in the real world. Agents in IE applications must be able to engage the user (or player) in interesting and, obviously, entertaining interactions. Also, NPCs must appear to engage other NPCs in

[3] NPCs' fulfilment of the *social ability* criteria is somewhat debatable, however the use of teams in a number of recent games could be looked upon as agents using such an ability.

interesting social interactions. To achieve this, relationship modelling must be used and this will be expanded upon in the section on the proposed architecture. The need for relationship modelling is very well illustrated by the following quote from **[6]**:

> *"The fact is, in most computer role-playing games, the NPCs don't give any semblance of intelligence," he continues. "Say you're in a bar and you throw your beer at the bartender one day. The next day you go back, and he's just as happy to see you. That shouldn't happen."*

3. Issues Unique to Agents for IE Applications

Intelligent agents have been used for a vast range of applications, from information assistants which sort e-mail and trawl the web for articles of interest to the user, to systems which act as air traffic controllers. This disparity of application areas has led agent based research to become somewhat fragmented. For this reason, it is worth identifying the application areas which are of most relevance to work on designing agents for interactive entertainment purposes. Two such areas have been identified - agents involved in robotics and agents which are designed to facilitate Human Computer Interaction (HCI). What follows will discuss the similarities between these two fields and then identify issues unique to the design of agents for IE applications.

The similarities between agents for robotics applications and those for IE stem from the fact that both are situated in complex and dynamic environments and are expected to perform continuously, in a robust and useful manner. For IE agents the environment is the virtual world in which the interaction takes place - worlds which are becoming ever more complex - while for robotics based agents the environment is the real world. The use of robot architectures for computer game NPCs is considered in **[13]**.

It should not be surprising that techniques used for agents in robotics can be borrowed for agents in IE. Over the development of AI in IE there have been a number of techniques borrowed from robotics, the most obvious being the A* path finding algorithm which is now widely used in IE applications **[27]**.

The differences between these two application areas come from differences in what is meant by "useful" performance. This will be elaborated upon below.

The similarities between agents involved in IE and those used in HCI are quite obvious. Agents involved in IE need to be capable of maintaining interesting, and obviously, entertaining interactions with the player, while agents involved in HCI must relate sympathetically with the user. These kinds of agents are often denoted affective agents **[22]**.

Agents for IE give rise to a number of issues not found in other systems. The key to understanding these is to realise that the goal in designing agents for IE is not to design the most efficient or effective agent, but rather to design the most interesting agent. IE is concerned with maintaining illusions of *believability*. The player must be able to believe that the agents involved in an IE system are living, breathing characters with goals, believes, desires etc of their own. Thus, it is not so important for an IE agent to choose the most efficient, cost effective option available to it, but rather to choose a reasonable plan of action and respond realistically to the success or failure of this plan.

Similarly, when designing an architecture for IE applications it is important that it be as easily configurable as possible. This requirement stems from the fact that these agents are to act as characters in entertainment experiences. For this reason, it must be possible for an author to configure the agent in order to achieve a range of different characters.

Finally, when considering the design of agents for IE, and this is particularly true for computer games, there are a number of practical restrictions which must be taken into account. For example, while it is a noble goal for all NPCs in a virtual world to be implemented as PPAs, this is not realistically achievable based on the hardware available to system designers. For a typical computer game AI processing receives approximately 10% of the processor cycles **[30]**. For this reason, it is important for an architecture to be scalable. It is expected that when using an agent based system such as that proposed, a game designer might make some characters which are central to the game fully proactive and persistent, while relying on traditional static techniques for most of the characters in the world. Similarly, it may be desirable for a designer to be able to dynamically switch an NPC from a simply static character to a fully proactive and from then on persistent one as the game requires. This kind of technique is referred to as *level-of-detail* AI **[35]** and arose from similar work conducted in graphics research.

4. Current Agent Architectures

The definition of an intelligent agent has been touched upon in the previous sections, however a deeper exposition is worthwhile. Following this, this section will explore some of the major trends in agent design in the past number of years. The field breaks into three main areas, reactive agents, deliberative agents and hybrid agents (a combination of the reactive and deliberative systems). Also a number of novel architectures which fit into none of the proceeding categories will be presented. Each architecture will be considered in turn illustrating the advantages and drawbacks of such systems for the IE application area. A number of relevant examples of past applications will also be given.

What is an Intelligent Agent?

The question “What is an intelligent agent?” is disturbingly nebulous. In **[33]** it is compared to the question *what is intelligence?* The reason for this is that agency is more of a software design philosophy than any particular set of techniques. Also, agent techniques are used, to varying extents, in a wide range of disciplines. Applications range from information assistants which sort e-mail and trawl the web for articles of interest to the user, to air traffic control systems. This makes devising a robust universally accepted definition of agency very difficult. **[33]** considers the problem briefly and proposes a definition worth considering. Two possible definitions arise, proposing the notions of weak and strong agency. A weak agent is considered anything that fulfils the four properties of agency given above.

The notion of strong agency is a little more controversial. A strong agent fulfils the four properties of agency but also is either conceptualised or implemented using concepts which would more usually be applied to humans. Such concepts include *mentalistic* notions (the Belief Desire Intention **[10], [24]** model of agency being an example of the application of such notions), emotional abilities **[5], [11], [29], [34]** and the representation of the agent through visual means, perhaps through cartoon or an animated face **[17]**. Agents for IE applications fit quite comfortably into the strong notion of agency.

Reactive Agents

Reactive, or behaviour based, agents are the simplest form of intelligent agent architecture. The most well known architecture arises from work by Brooks **[2]** which resulted in the subsumption architecture **[3]**. The key to a reactive agent is that they work in a hard wired stimulous response manner. Certain sensor information always results in a specific action being taken. This can be most simply encoded as if-then rules.

This very simple architecture has a number of compelling advantages. These include:

- Reactive systems are very fast.
- Reactive systems are simple, both to program and understand.
- Reactive agents are deterministic and so completely predictable.
- Reactive systems require little support infrastructure.

In fact, it is the most widely used architecture in IE applications today **[18]**, **[30]**. In IE applications reactive agents are most often implemented using Finite State Machines (FSMs).

In spite of these advantages, however, reactive architectures have a couple of major drawbacks. Firstly, every possible situation the agent might find itself in must be encoded in the rules. This places a huge burden of responsibility on the designer as they must have allowed for every eventuality. Related to this is the fact that for complex environments the range of possible situations can be vast making the design of a reactive system very difficult, if not impossible. For example, a behaviour based dialog system presented in **[12]** required a separate behaviour for every possible utterance by every possible speaker.

Another major drawback, is that reactive systems have trouble forming all but the simplest long term plans. Reactive agents have no internal model of the world and so are incapable of reasoning about it in any abstract way. To reiterate, the agent simply receives inputs and reacts to these through simple rules.

Deliberative Agents

A more sophisticated agent architecture is that of the deliberative agent. Built upon classical symbol based AI techniques, these agents build internal models of the world that they can then use to formulate plans in order to achieve goal states.

$$StartingState + GoalState \Rightarrow Plan$$

A system such as first order logic based situation calculus is used in order to formulate the model and construct plans. This planning ability is the main advantage of such systems. They do however suffer from the complexity involved.

Due to the use of logic based inference, a deliberative system cannot make the real-time guarantees required by agents in the IE realm. Inference based reasoning is very processor intensive and so, it is not realistic to use completely deliberative agents in an IE system. Another serious drawback to the use of deliberative agents is that they require constant maintenance of a knowledge base. In fast moving dynamic environments such as those encountered in IE applications this can be a major issue. Also, consistency of the system must be maintained often involving updating any inferences made based on knowledge that may change.

Purely deliberative systems have not been used widely in IE applications. A few examples do exist; one such being **[9]** however, one must note that this did not run in real-time. Also the Excalibur **[20]** project is attempting to create a deliberative agent for IE applications. This is mainly being achieved by placing heavy restrictions on the potential search space for any inferences. From this project the concept of an *anytime agent* has emerged. This will be described below.

Hybrid Agents

Hybrid agents combine aspects of both reactive and deliberative agents. The philosophy behind hybrid agents is that they can make use of the best features of both architectures. For example, a reactive system can be used to deal with time critical behaviours such as collision avoidance, while long term planning can be achieved using a deliberative system. The inclusion of two sub-systems, however, leads to the introduction of another problem, that of mediating between the two systems. Examples of Hybrid agents include INTERAPP **[19]**, and being-in-the-world **[7]**.

Interesting Variations

A number of variations on the three core architectures have been developed. Some of those that are most relevant to IE applications will now be elaborated upon.

Broad/Believable Agents

More of a design philosophy than an architecture, designing broad believable agents involves designing agents that are reasonably proficient at as wide a variety of things as possible. The key here is to ensure that the agent remains at all times believable. Broad agents have been used to good effect in the Oz Project **[1]**.

Emotional Agents

Another requirement unique to games (and some applications in HCI) is the need for agents to be able to display and react to emotion – or at least give the illusion of doing so. A number of attempts have been made at this including **[11]**, **[5]**, **[34]**.

Social Agents

Finally, agents in a virtual world are required to communicate believably with both a human user and other agents. Again the differences between the general notion of social ability in agent research and the notion of sociability in terms of IE agents must remain as discussed previously. Social agents in terms of IE applications are agents with the ability to interact socially with both human users and other agents. A lot of research has been done on this area as a part of the larger HCI research effort. In particular, *chatterbots* (agents which engage a user in conversation) have been implemented extensively, **[31]**.

Anytime Agents

Emerging from the Excalibur project **[20]**, the concept of the anytime agent is quite simple. These agents are deliberative based with the added constraint that they form plans iteratively. For each slice of processing given to a particular agent as much of the current plan as possible is inferred. When the time slice runs out some action is taken. For subsequent processing time slices allotted to the agent, plans are re-evaluated and improved upon.

The advantages of this kind of systems is that it overcomes some of the processing restrictions associated with deliberative systems and also, increases the believability of the agent, as the agent is constantly pursuing some aspect of the current plan.

5. Final Architecture

Before discussing the agent architecture proposed by this work the design goals will be reiterated. The key goals which the proposed design seeks to satisfy are:

- *autonomy*
- *social ability*
- *reactivity*
- *pro-activeness*
- *persistence*
- *scalability*
- *extendibility*
- *believability*
- *configureability*

In order to satisfy all of these goals a modular architecture is proposed. The key building blocks of this architecture are given in **Figure 3**. The system is composed of four major sub systems:

- *behaviour system*
- *social system*
- *goal based planner*
- *schedule*

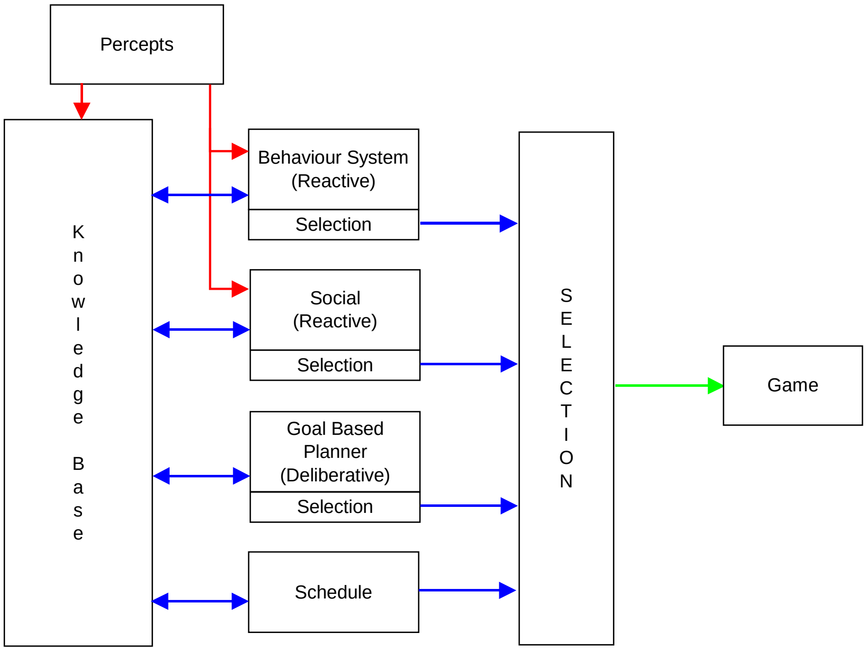


Figure 3 The modular design of the proposed agent architecture.

Precepts come from the game and are translated into a common knowledge base to which each of the above systems has access. Based on these, each system passes proposed courses of action into a selection module. Each of the sub-systems will now be elaborated upon in some detail.

The behaviour-based system is considered a classic reactive system. It will be made up of pre-programmed behaviours each of which arises from a particular set of world conditions. This allows very simple, robust and processor efficient agent behaviour. Also, as part of the behaviour system will be a model which will use a number of measures to describe agent personality. A body of research in this field exists and a number of avenues are being explored **[22]**, **[25]**.

The social system is also a reactive system and so is similar in many respects to the behaviour system. The main task of this module is to deal with interactions between the agent and other characters/players. A distinction must be made between what we refer to as a social system, and what classic agent research would refer to as a social system. In the sense in which we talk about a social system we refer to actual social interactions such as talking, arguing, dancing etc. This kind of social requirement is unique to IE applications and so not as obvious as the more traditional agent based sense of the word.

The system will be responsible for taking social action in the world. Also, it will maintain a model of the agent's relationships with other agents/players (friend, enemy etc. The main reason for separating the social system from the behaviour-based system is to stay true to the modular philosophy behind the architecture. These two systems have quite separate roles, and it is conceivable that some applications may have no need for a social system. For this reason they have been separated. However, due to their similarities these two modules will be more tightly coupled than any of the other modules in the system.

The goal-based planner will be just that. A deliberative system which has the ability to make long-term plans based on the contents of the knowledge base. This system will use some form of first order logic in order to reason about the world.

Finally, the schedule is a module unique to the needs of the IE domain. Often, as a means to maintaining the illusion of believability it is necessary for agents to appear to go about their daily business such as going to work, coming home, or going out to socialise. This can be achieved with the smallest processor hit through the use of a schedule giving a task for an agent for any particular time of the day. This module is particularly useful for the IE domain as it achieves a high level of believability through a very small amount of processing.

Wrapped around these modules, is a selection mechanism that mediates between the different modules and chooses a course of action based upon those suggested.

It is believed that the architecture proposed satisfies all of the design goals specified previously. Many challenges still remain, particularly to remain true to the design goals throughout the implementation. Some of these will now be outlined.

Implementation Challenges

The proposed architecture poses a number of significant challenges to any implementation. The key will be to remain as true to the design goals outlined as possible. For this reason, every effort will be made to make the system as modular, extendible, scalable and easily configurable as possible.

As a means to implement the reactive systems, particularly the social system, the use of role passing systems is being considered. Role passing is slightly more sophisticated than typical reactive systems. Presented in **[14]** role passing involves implementing inference rules to code different roles which an agent might take. For example chasing an enemy, defending a point or bargaining. Each role has a number of variables that are bound to the different objects such as target, enemy, bargaining partner etc. Role passing offers slightly more sophistication to typical implementations of reactive systems (e.g. FSMs) while still retaining the main benefits, such as speed and determinism.

Deliberative systems often pose a problem for implementation. Mainly this is due to the processing time and complexity involved. However, implementations have been achieved and those presented in **[8]** and **[21]** are being considered in particular.

Infrastructure for a system populated by agents such as those proposed is also a serious challenge. The main issue here is in prioritising computation so that agents immediately visible to the player/user receive more processing time than those not immediately visible to the player and a number of scheduling techniques are being investigated in order to achieve this. There are also a number of advantages to be gleaned from the use of level of detail AI, a technique whereby certain demands (such as accurate path finding) are only made of those agents in the immediate vicinity of the player.

6. Conclusion

An agent architecture for IE applications has been presented. To reiterate, the key goal is to create an architecture that will allow the construction of believable and interesting agents for entertainment purposes. Although there are still many challenges ahead, we believe that the proposed architecture, by building on previous architectures, will satisfy this.

7. Bibliography

[1] Joseph Bates, A. Bryan Loyall, W. Scott Reilly, *Broad Agents*, Proceedings of the AAAI Spring Symposium on Integrated Intelligent Architectures, Stanford University, March 1991.

[2] Rodney Brooks, Intelligence without reason, In Proc. of IJCAI-91. Morgan Kaufmann, San Mateo, 1991.

[3] Brooks, Rodney A. "A Robust Layered Control System for a Mobile Robot,", MIT AI Lab Memo 864, September 1985.

[5] Alastair Burt. Emotionally intelligent agents: The outline of a resource oriented approach. In Dolores Ca~namero, editor, Emotional and Intelligent: The Tangled Knot of Cognition, pages 43-48, 1998.

[6] Donna Coco, Creating Intelligent Creatures, *Computer Graphics World* July, 1997 http://cgw.pennnet.com/Articles/Article_Display.cfm?ARTICLE_ID=49169

[7] M.A. DePristo, R. Zubek , being-in-the-world, *Proceedings of the 2001 AAAI Spring Symposium on Artificial Intelligence and Interactive Entertainment (to appear)* http://www.cs.northwestern.edu/~rob/publications/

[8] John Funge, Representing Knowledge Within the Situation Calculus Using Interval-Valued Epistemic Fluents, Journal of Reliable Computing, Kluwer, issue 1, volume 5, 1999.

[9] J. Funge, *Making Them Behave: Cognitive Models for Computer Animation*, Ph.D. Thesis, University of Toronto, 1998

[10] Michael Georgeff, Barny Pell, Martha Pollack, Milind Tambe, Michael Wooldridge**,** The Belief-Desire-Intention Model of Agency, Springer Publishers, 1998.

[11] Piotr J. Gmytrasiewicz and Christine L. Lisetti, Modelling Users' Emotions During Interactive Entertainment Sessions, In *Working Notes of the AAAI Spring Symposium on Artificial Intelligence and Interactive Entertainment.* Menlo Park, CA: AAAI Press. (Stanford, CA, March 2000)

[12] Hasegawa, T., Nakano, Y.I., Kato, T. 1997. A Collaborative Dialogue Model Based on Interaction Between Reactivity and Deliberation. In Proceedings of the First International Conference on Autonomous Agents, 83 – 87. Marina Del Ray, CA: ACM Press.

[13] I.D. Horswill, R. Zubek , Robot Architectures for Believable Game Agents, *Proceedings of the 1999 AAAI Spring Symposium on Artificial Intelligence and Computer Games,* AAAI Technical Report SS-99-02
http://www.cs.northwestern.edu/~rob/publications/

[14] I. Horswill, How Far Can Behavior-Based Systems Go? To appear in *SAB-96*.

[15] N. Jennings, K. Sycara, and M. Wooldridge. A roadmap of agent research and development. Journal of Autonomous Agents and Multi-Agent Systems, 1:275--306, 1998.

[16] J. E. Laird, M. van Lent, Human-level AI's Killer Application: Interactive Computer Games, Proc. AAAI 2000, pp. 1171-1178. AAAI Press / The MIT Press, 2000.

[17] W. Lewis Johnson, Jeff W. Rickel, and James C. Lester, Animated Pedagogical Agents: Face-to-Face Interaction in Interactive Learning Environments, The International Journal of Artificial Intelligence in Education (2000) 11, 47-78. http://www.csc.ncsu.edu/eos/users/l/lester/Public/apa-ijaied-2000.pdf

[18] Michael Mateas, *An Oz-Centric Review of Interactive Drama and Believable Agents,* Technical Report CMU-CS-97-156, School of Computer Science, Carnegie Mellon University, Pittsburgh, PA. June 1997.

[19] J. P. Muller and M. Pischel. *The agent architecture inteRRaP: Concept and application*. Research Report RR- 93-26, Deutsches Forschungszentrum fur Kunstliche Intelligenz, Kaiserslautern, Germany, 1993.

[20] Nareyek, A. 2000, Intelligent Agents for Computer Games, *In Proceedings of the Second International Conference on Computers and Games (CG 2000),* to appear.

http://www.ai-center.com/projects/excalibur/publications.html

[21] Nareyek, A., Open World Planning as SCSP, In Papers from the AAAI-2000 Workshop on Constraints and AI Planning, Technical Report, WS-00-02, pages 35-46. AAAI Press, Menlo Park, California, 2000.

[22] Picard, R., *Affective Computing*, MIT Press, Cambridge, Mass, 1997.

[23] Pisan, Y., Character building: A form of knowledge acquisition, In Proceedings of AAAI 2000 Spring Symposium on AI and Interactive Games, 2000

[24] A. S. Rao and M. P. Georgeff, "*BDI agents: From theory to practice*," Tech. Rep. 56, Australian Artificial Intelligence Institute, Melbourne, Australia, April 1995.

[25] Rizzo, P., Veloso, M., Miceli, M., Cesta, A., Personality-Driven Social Behaviors in Believable Agents:, in Working notes of the AAAI Fall Symposium on "Socially Intelligent Agent" , Cambridge, MA, November 8-10, 1997.

[26] Andrew Stern, AI Beyond Computer Games, *Presented at AAAI AI and Interactive Entertainment Spring Symposium 2000*. http://www.cs.nwu.edu/~wolff/aicg99/stern.html

[27] Bryan Stout, “Smart Moves: Intelligent Path-Finding”, Game Developer Magazine, October 1996. http://www.gamasutra.com/features/19990212/sm_01.htm

[28] M. Tambe, W. L. Johnson, R. Jones, F. Koss, J. E. Laird, P. S. Rosenbloom, and K. Schwamb. Intelligent agents for interactive simulation environments. AI Magazine, 16(1), Spring 1995.

[29] Ian Wilson, The Artificial Emotion Engine, Driving Emotional Behaviour, In *Working Notes of the AAAI Spring Symposium on Artificial Intelligence and Interactive Entertainment.* Menlo Park, CA, March 2000

[30] Steven Woodcock, AI Roundtable Moderator’s Report, *Game Developer’s Conference 2000*, March 10-12 http://www.gameai.com/cgdc00notes.html

[31] J. Weizenbaum. ***ELIZA* -- *A computer program for the study of natural language communications between men and machines*.** Communications of the Association for Computing Machinery, 9:36--45, 1966.

[32] M. Wooldridge, N.R. Jennings, "Agent Theories, Architectures, and Languages: A Survey", in Intelligent Agents, Proceedings of the ECAI-94 Workshop on Agent Theories, Architectures, and Languages, Springer-Verlag, Lecture Notes in Artificial Intelligence, Vol. 890, Pages 1-39, 1995.

[33] M. Wooldridge and N. R. Jennings. Intelligent agents: Theory and practice. *The Knowledge Engineering Review*, 10(2):115--152, 1995.

[34] Wright, I., *Emotional Agents*. PhD thesis, University of Birmingham, 1997.

[35] Ian Wright and James Marshall, More AI in Less Processor Time: ’Egocentric’ AI, Gamasutra, June 19th 2000. http://www.gamasutra.com/features/20000619/wright_01.htm