A Survey of Agent Programming and Adaptive Serious Games

Nadia Hocine, Abdelkader Gouaich

Agent programming and adaptive serious games: A survey of the state of the art

ABSTRACT
This paper provides a survey of software agents in adaptive serious games. It outlines the most important challenges of the game design such as adaptation. The study (i) overviews the state of the art of behavior programming in agent-based games (ii) provides our further reflection about the agent programming model dedicated to therapeutic serious game (iii) and finally introduces some major challenges to be addressed by the community in the medium term.

Categories and Subject Descriptors
A.1 [**Introductory and survey**]

General Terms
Design, Human Factors.

Keywords
Adaptive serious game, agents, state of the art.

1. INTRODUCTION
Serious games are currently attracting the interest of researchers and professionals in various fields such as education, health, vocational training, governance and defense.

Currently, there is no formally accepted definition of serious games. However, one can state that serious games are commonly based on a fundamental principle which is reflected by taking into account simultaneously and consistently: (i) serious aspects that determine the pedagogical objectives such as the transmission and/or acquisition of knowledge, know-how, or information; (ii) and fun aspects, which focus on the motivation and the management of end users' frustration. In the rest of the paper, we will refer to end users as *players-learners*.

The answer concerning players-learners motivation has been analyzed by research works from different angles:

- The use of appropriate and attractive human-machine interfaces in order to facilitate the acceptance of player-learners. This implies, for instance the choice between visual and auditory interaction modalities, display frequency and the level of graphics rendering details in the game e.g [24].

-The use of general principles and good practices of game design in order to create an immersion or a flow as described by Czikszentmihalyi [10].

- Allow a dynamic adaptation of the serious game in order to individualize and contextualize the game experience for each player e.g [28]

We focus in this paper on the last two points. We are interested, particularly in review of works that exploit the advantages of agent-oriented programming to design adaptive serious games. In fact, we study the behavioral programming of software agents that can adapt a serious game according to pedagogical objectives and observed skills of the player-learner.

This paper is organized as follows: we describe in the second section our motivation, identify the challenges we may confront during the design process of an adaptive game. We also discuss the use of agent-based approach to overcome these challenges. The following section aims to describe the main approaches of behavior modeling in agent-based game. We present in the fourth section the state of art of behavior programming in agent-based game. Finally, we conclude this paper by: (i) analyzing and discussing these works (ii) and describing our outlook concerning agents programming dedicated to therapeutic serious games.

2. MOTIVATIONS
We are interested in this paper in adaptation problem in serious games. In this context, we focus on the agent approach as a means of behavioral modeling and programming of adaptive games.

Thus, the objective of this paper is to describe the main work on behavioral programming in adaptive agent-based games. The purpose of this section is to: (i) specify the challenges posed by adaptive games and their influence on the behavioral programming process (ii) and discuss why an agent-based approach may be effective in this context.

2.1 Adaptive game design challenges
Among the challenges that may be confronted in the design process of an adaptive game, we can mention:

(i) Taking into account the assessment of players' capacity and skills during the game session. This challenge is also due to the heterogeneity of players who can be for example children, students, elderly people or patients with disabilities. We are interested in systems that rely on a game design process which is centered on the user (or player-centered game design) [35]. In this case, the game must have adaptive behaviors to cognitive and motor skills of users and to their performances in the game.

(ii) Compliance to real-time constraints. Time constraints occur at various stages such as the computing of game entities behaviors, the control and adaptation overhead. These elements can deeply affect the game performance in terms of response time and interaction quality.

(iii) Creating a flow or an immersion in the game. The goal is to introduce the elements allowing involvement of the player in the game world. The aim is to keep his motivation and commitment during the game experience. This has been inspired from works on game flow and immersion [10] [6].

Cite as: Title, Author(s), *Proc. of 10th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2011)*, Yolum, Tumer, Stone and Sonenberg (eds.), May, 2–6, 2011, Taipei, Taiwan, pp. XXX-XXX.

According to our vision concerning the case of serious games, immersion can be verified by responding to control process challenges which concern: (i) the game environment including the management of its interactions with the user (ii) and the virtual game world that includes the control of its entities such as the virtual characters. We can mention in this case the management of:

- Difficulty level in the game: The difficulty level should be adapted to the player progression and to the game experience goals.

- Feedback: the game must respond in a coherent and intelligible manner to the players' actions. We can cite for examples: encouragement messages, congratulations after success and help after failure or obstruction.

- Playability: we mean by playability the conditions that the system must check in order to satisfy the usability of the game. We can cite for example: an easy game interface to use, graphics and sound management, and anticipation of illusion that the system gives the choice to the player of his next state in the game.

2.2 Why an agent-based approach?

The realization of adaptive serious games can be achieved through an agent approach. As a matter of fact, the idea of using agent concept is not new in itself, as it has already been used in: (i) games based on artificial intelligence (or Game AI) [23] [40] (ii) and simulation systems and game-based virtual reality [29].

In the first category, we can refer also to the use of agent concept in commercial games such as Half-Life2 [17] (for more discussion see [12]) Civilization IV [8] (for analysis see [1]) and Black & White [5] (for a discussion see [39]). These games use agents to model virtual characters with intelligent behaviors.

The second category of works deals with the concept of virtual fidelity. In these virtual reality systems, the agent concept is used to model the virtual character representing the user in the game (human-like character) eg. [26]. The agent must therefore have the same behavior as a user. In this case, the representation of the system behavior and its states is closely related to users' behavior. We focus in our analysis on the adaptation of game behavior that is related to the first category of these works.

In general, the design of agent-based adaptive games allows:

- A simple game structure: games often use software entities or virtual characters of the game world, such as PCs (Player Character) or NPCs (Non Player Character), as individualities that have a decision process.

- Facilitate management and control process in the game environment: the use of agent-based approach allows the distribution of control in the game.

- Overcome the most challenges in the management of real-time interactions, notifications and events in the game. This can be realized by exploiting the software agents' properties such as autonomy, reactivity and pro-activity, and by using agents that have a limited cycle of execution in the game session.

- Deals with the adaptation challenge in the context of serious games. This can be achieved by benefiting from the expressive and flexible decision-making models. These models are used to customize and contextualize the behavior of each game entity.

3. MODELING BEHAVIOR IN AN AGENT-BASED GAME

The purpose of this section is to describe the main approaches used to model the behavior of game characters so-called "agents". The used characters are usually the NPCs ones.

We can bring out two categories of works that concern behavior modeling in agent-based games. The first category deals with path finding [41] so-called 'navigation' in game's virtual world. This category is based on artificial intelligence algorithms in order to plan paths of virtual characters in the game. We can cite for instance the A* algorithm [32] and real-time path planning algorithms such as RTA-RG [18].

We focus in our study on the second category. This latter refers to the main behavior modeling approaches of games and its entities. These works are usually based on Finite State Machines (FSM), rule-based systems, Goal-Oriented Action Planning (GOAP) and Machine Learning (ML) approach.

3.1 Finite state machines (FSM)

This approach uses a graph which has a finite number of states in order to model game entities behavior [13]. Each node in the graph represents a specific context for the virtual character, and the transitions between the states provide the conditions that are used to switch the behavior from one state to another.

FSM is considered as a traditional and a simple means to be used during the design process of a video game. For instance, FSMs have been already used in video games such as Age of Empires [15].

Using FSM requires a prior planning of all game states by the programmer. This technique is therefore expensive in terms of game development and gives a limitation to the possible behaviors that a game should include.

3.2 Rule-based system

In this approach, the state of NPC depends on the rules that have been associated to it. This model has been proposed to facilitate the control of NPCs in complex games such as RPGs (Role-Playing Game). We can refer for instance to [7] and [30] works. Behavior programming of the game-world entities was often performed via XML scripts. These scripts contain all events that reflect the NPC behavior at a given time. For instance, Baldur's Gate [4] and Virtual Fighter 2 [34] games use the rules-based systems approach. In these systems, rules are used to control the NPCs' behavior.

As in the case of FSM, prior planning of all game rules should only cover some behaviors expected during the game design phase without incorporating other behaviors. Introducing a new behavior to the system or a simple change in its rules may require a global checking of rules consistency to avoid conflicts among rules and behaviors.

3.3 Goal-oriented action planning (GOAP)

In this model, the graph nodes are considered as goals that are planned for a virtual character. Each character chooses its behavior according to its current goal. These goals are dynamically determined during the game session [25]. Among the first games that have used this approach, one can mention FEAR that is a first-person shooter game [19].

[16] is another work that uses this model for an agent-based game architecture using dynamic generation of game goals. Agents in this architecture use a teleo-reactive program-- composed of a set of pairs (actions, conditions)--, to determine or calculate their plans.

However, the real-time planning of agents' behavior requires a significant processing time. This can influence the performance of the game in terms of responsiveness time [41].

3.4 Machine learning (ML)

Machine Learning gained the interest of many academic game projects on and commercial video games [31] [23] [1]. These studies aims at creating game worlds that are composed of virtual characters that behave in an intelligent and adaptable manner. This can be realized using artificial intelligence algorithms.

This approach consists in applying learning algorithms to model and calculate the behavior of agents. In this case, agents represent usually adversary characters of the game world. We can mention as an example of works that use this approach, the techniques of artificial neural networks (or Neuronal Networks: NN), evolutionary algorithms [11] and the reinforcement learning algorithms (or RL). This implies, for example the works of [36] and [1]. These works are often based on the classical modeling aspect of game's virtual characters, and use dynamic programming of the agents' mental state to describe their reasoning.

4. SURVEY OF BEHAVIOR PROGRAMMING IN ADAPTIVE AGENT-BASED GAME

In this section we discuss the problem of designing adaptive agent-based games. A system is adaptive when it is able to change its behavior or the behavior of its entities, in response to certain number of well identified events [2]. In the case of games, the system can be adapted by modifying: (i) the behavior of PCs or NPCs in the virtual game world (ii) and the game environment and its various interactions with the player [28]. Therefore, we are interested in this section in the principal works that deal with these two points. Our main goal is to study the manner through which the agent approach has been exploited to adapt the game behavior.

4.1 Analysis criteria

In order to analyze the works using agents to adapt the game behavior, we propose an evaluation framework that contains the following analysis criteria:

- Adaptation perimeter that can be:

(i) Game agents: This consists in checking whether the adaptation concerns the virtual characters behavior in the game world.

(ii) Game environment: In this case the aim is to adapt global game's parameter such as objectives and quests, speed, enemies' appearance intervals, game rules and so on.

-Adaptation parameter: The user parameter that can be used by the adaptation process which can be:

(i) User performance: the player performance can be determined for instance by his score, success or failure rate and time to complete some tasks in the game.

(ii) User ability: This means that the adaptation process takes into account the cognitive and motor skills of the player.

-In-line/off-line adaptation: This criterion specifies whether the adaptation process is performed during the game session (or inline mode) or outside it (or offline mode). In the case of off-line adaptation, the management of the game state is carried out in an implicit manner i,e game behavior is pre-planned by the game programmer. However, in the case of in-line adaptation, the game state is determined during the game session and the behavior depends on this new state.
-Game's behavior modeling: This criterion specifies the modeling approach which is used in the game design process.

-Single/multi player adaptation: In this case we specify whether the agent approach also takes into account the adaptation of the players' collective behavior.

- Adaptation of individual /collective behavior: This criterion specifies whether the adaptation process takes into account the adaptation of individual and/or collective behavior of game agents.

- Open-loop/closed-loop system: The system uses an open-loop controller (or non-feedback controller) when it does not use a feedback or return to determine whether its results (or outputs) have been accomplished. This system focuses on time factor because it supposes fast treatments that do not take into account errors' handling. However, in the case of closed-loop controller, feedback is used to ensure the adaptation control. This can be realized for instance through a decision tree or a learning algorithm.

4.2 Presentation of the state of the art

In this section we describe the main works on behavior programming of adaptive games that are based on an agent approach.

4.2.1 Adaptive behavior programming using personality-based adaptation technique [36]

This work is based on a player-centered game design. It proposes an agent-oriented framework implemented in the Truevision3D 6.2 game engine. The action scenario is built on a zombie's game. In this game, the player is represented by an avatar that possesses some weapons. The goal of this latter is to kill all the zombies in the game world.

In this work, game's behavior is programmed using a personality adaptation module encapsulated in a reinforcement learning framework. The parameter used for adaptation is the performance of the user in successive game sessions. The adaptation perimeter in this case is the virtual character so-called agent. This agent

must be in compliance with the individual style and strategies of each player through the game session (i,e in-line mode). Each new personality of the player-character (PC) depends on the other personalities that have been created before.

This work uses a behavior's modeling based on the agents' goals GOAP [25]. The proposed game uses also a rule-based system to determine these NPCs' behavior. In addition, the learning process used by the framework is based on a reinforcement learning algorithm (RL [33]). Therefore, this allows each agent to determine its personality according to the existing ones.

The control system used is closed-loop. In fact, the game loop starts with the execution of the adaptation process. During this process, each agent that has its own adaptation module activates this latter to generate its personality in the game. This module uses a neural network algorithm (NN). Indeed, the system calculates the new weights of the NN to generate new parameters in order to create the agent's personality. This is realized while taking into account the errors diffused by the execution module in the game. After this reinforced learning, the personality of the character can be created.

This work is based on adapting the collective behavior of agents. It applies the same behavior policy for the entire agents during the game session. Agents in this framework represent virtual characters that have a low autonomy and social ability. An improvement of this Framework has been proposed in [37]. Their contribution is to introduce tactics and strategy concepts during the generation of characters' personalities. In addition, they use the concept of multi-agent systems to incorporate social ability to agents in game.

Finally, we can note that the adaptation in this work concerns only the case of single player.

4.2.2 Adaptive behavior programming using case-based reasoning [28]

In this work, the adaptation of game's behavior is founded on a case based reasoning algorithm (or CBR). The game's behavior is based on a prior planning of all possible game actions (or goal-oriented action planning). In addition, behavior modification is related to the different cases that are anticipated in the game design steps.

The adaptation perimeter in this work concerns the game environment. It is about the adaptation of the game experience that depends on the players' style and performance through a user model.

The *behavior transformation* is performed via specific system architecture of the game. This architecture contains two layers: a reactive one that allows the management of real-time interactions, and the other one is the reasoning layer to manage the virtual characters' states.

In the reactive layer, the developer uses a behavior programming language ABL (Adaptive Behavior Language) based on a reinforcement learning algorithm [22]. This language is based on the partial programming. This is a programming paradigm that assumes that the programmer must only specify some behaviors of the system and give the task to its execution module to create other more complex behaviors. The control system used is the open-loop one. The character or the agent in ABL is composed of a behaviors' library. The purpose of the use of this language is to create at the reactive layer credible and reactive agents.

The reasoning layer is responsible for behavioral adaptation. Changing behavior of the characters is based on an emotional user-model.

The adaptation is realized in off-line mode. The management of the game state is performed in an implicit manner as it implies the pre-planning of the game's actions.

Finally, this work did not address the case of multiplayer adaptation. It implies a process of user-centered game design and use reactive agents in order to satisfy the real-time requirement in the game.

4.2.3 Adaptive behavior programming using neuronal networks [31]

This work is part of the Nero video game research project. This project proposed a machine learning based on an artificial intelligence algorithm called Neuroevolution [23]. This technique was originally used by the agent simulation work in robotics field.

The adaptation perimeter concerns agents that adapt the game by learning from user actions during the game experience. NERO game uses the same adaptation policy, through the learning algorithm, for all game's agents. This is done while taking into account the user's performance as an adaptation parameter.

Behavior modeling is based on a ML approach that uses a neural network (NN, Neural Network). It applies rtNEAT learning method (realtime NeuroEvolution of Augmenting Topology, NEAT) [31] which is invented under the Nero project.

The system controller is closed-loop as it uses a machine learning based on NEAT or neuroevolution method. The latter is based on the *complexification* of the neural network by adding nodes and connections to create complex behaviors for agents during the game session.

Adaptation in this case is in-line and involves the modification of agents' behavior and its base knowledge during the game session.

Finally, the proposed adaptive behavior programming technique is intended for games and simulation environments that deal with the single user adaptation.

4.2.4 Adaptive behavior programming based on evolutionary algorithm [3]

This work is introduced in the context of games which are based on virtual opponents' team (Team-oriented games). In these games, the adaptation concerns the behavior of the groups of virtual characters (or NPCs groups).

TEAM (Team-Oriented Evolutionary, Adaptability Mechanism) defined in this work, uses an evolutionary learning algorithm. The agents' learning process is realized through the cooperation of evolutionary algorithms of each NPC that belong to the same opponents' group. Thus, the adaptation perimeter concerns virtual characters of the game world.

Adaptation parameter in the case of programming TEAM agents is also the player's performance. In addition, the control system

used is open-loop as it does not control the generated agent team's behavior.

Agents' behaviors are programmed using the action planification based on an evolutionary algorithm. Moreover, a finite state machine is implemented as a modeling technique for each agent group behaviors. For each state in the FSM, representing the behavior of the agents that belong to the opponents' team, an evolutionary algorithm is associated. The goal of each algorithm is to develop or learn new behaviors of the opponent group for a given state.

This work deals with the adaptation of collective agents' behavior intended for the real-time strategy games, it provides a single player adaptation process.

The experimentation of this work use Quake III commercial game [27] in order to develop new tactics of the game. Finally, we note that the adaptation is used in off-line mode.

4.2.5 Adaptive behavior programming using a reinforcement learning algorithm [1]

This work introduces an agent-based adaptation dedicated to the popular Civilization [8] strategy game. In this game the player should build cities and then defend them while forming an army in the game virtual world. The idea is to integrate a dynamic generation of strategies through a learning algorithm.

The adaptation parameter, used to generate the new strategy during the game session, is the player's performance and style. Thus, adaptation in this case is performed in in-line mode.

The adaptation perimeter in question is the game agents. This concerns the adaptation of these agents' individual and collective behavior. Indeed, agents represent the NPCs in the game and only one of them resonates using a reinforcement learning algorithm (or RL: Reinforcement Learning). In fact, it determines strategies through the Q-learning algorithm [38]. The control system is closed-loop because the feedback is used by the machine learning in order to compute new strategies in the game.

The RL-based agent is used in this work in order to provide adaptive behaviors. The modeling agents' behavior use a stochastic process represented by a hidden Markov automaton (or Markov chain). The behavior is based on the dynamic programming of the agents' mental state which is deduced from the reasoning of RL-agent.

Finally, we note that this technique has only treated the case of adaptation to individual user behavior (or single-player adaptation).

4.2.6 Adaptive behavior programming based on theory of mind [21]

This work uses the theory of mind for agent's behavior programming. In this type of work, adapting behavior of agents is based on the description of the agents' mental state. The experiment uses an agent called KGBot which is realized via a game simulation environment of UT (Epic's Unreal Tournament) [14].

Adaptation in this case concerns the player avatar (or agent PC). This system communicates the game UT via a module allowing the control of the characters in the game world.

Modeling behavior defined in this work focuses on a GOAP approach. The internal state of the agent KGBot was realized using the BDI model which is base on the description of beliefs, goals and intentions of the agent game. The implementation of the internal state of KGBot uses, for its decision-making process, the BDI-agent engine called UM-PRS [20] and the real-time path planning algorithm RTA- RG [18].In fact, UM-PRS is a system based on traditional goal-directed reasoning and a reactive behavior. The control system is open-loop; agents do not use their feedback in order to determine their actions in the game.

The adaptation is performed in in-line mode. This means that the management of the state of the game depends on the real-time planning of agents' behaviors. Moreover, this work is based on the adaptation of collective agents' behavior as it requires behavior modification of KGBot agent. Finally, this work was also intended for single-player adaptation.

4.2.7 Adaptive behavior programming based on user-model [9]

Various researches have been aborted on the adaptation of game behaviors by introducing system architectures that are based on a user model. Among these studies we choose to describe [9] work.

This work is based on designing an educational game called Prime/Climb. This is a game where the player should pass certain obstacles during his ascent of a mountain and the game offer him a contextualized help. The goal of adaptation is to generate appropriate feedback to the player's learning situation during the game session (or in-line mode). The adaptation parameter is the player performance. Programming behaviors in this case is based on a learning model. This latter allows the planning of the various system behaviors associated to a learner during the game experience.

The adaptation perimeter concerns agents that adapt their behavior according to the interactions' result and player progression in the game. Indeed, the game proposed represent the help associated to the player using a software entity called a *pedagogical agent*. Adaptation technique is based on behavioral programming of pedagogical agent in the game.

Behavior modeling is based on GOAP. In fact, these plans will be used by software agents to adapt the user's feedback.

Finally, the game uses an open-loop controller system. It uses a single agent in order to adapt the assistance to the user in a single-player game context.

4.3 Analysis

Table 1. Overview of the state of the art analysis

	Adaptation perimeter	**In-line/off-line adaptation**	**Adaptation of individual / collective behavior**	**Single/ multi player adaptation**
	Adaptation parameter	**Game's behavior modeling**	**Open/closed-loop system**	
[36]	PC and NPC agents	In-line	Collective behavior	Single-player adaptation
	User performance	Hybrid : GOAP and ML (RL)	Closed-loop	
[28]	Game environment	Off-line	-	Single-player adaptation
	User performance	Hybrid : GOAP and ML (CBR)	Open-loop	
[31]	Agent	In-line	Collective behavior	Single-player adaptation
	User performance	ML (NN)	Closed-loop	
[3]	NPC agent	Off-line	Collective behavior	Single-player adaptation
	User performance	FSM, evolutionary algorithm	Closed-loop	
[1]	RL-agent	In-line	Collective behavior	Single-player adaptation
	User performance	Hybrid: ML: RL, HMM	Closed-loop	
[21]	PC agent	In-line	Collective behavior	Single-player adaptation
	User performance	GOAP	Open-loop	
[9]	Pedagogical agent	In-line	Individual behavior	Single-player adaptation
	User performance	GOAP	Open-loop	

Table 1 gives an overview about the analysis of the principal works discussed in this paper. This analysis is based on the evaluation framework proposed before:

Adaptation parameter

According to Table 1, we deduce that the adaptation parameter in the reviewed works concerns either (i): the entities or virtual characters that represent usually the NPCs [3] and the player's avatars in some cases [21]. This can be performed through a dynamic change of their behavior in the game [31] [9] (ii) or only the game environment without taking into account the behavior of its virtual characters [28]. However, we note a lack of the techniques that deals with the two perimeters at once. In fact, we bring out on the basis of this analysis two axis of works:

(i) the first one focuses on the adaptation of virtual characters' behavior, generally called agent game, which is analyzed in the context of game AI. In this case we distinguish between works that use a learning process or an agent based-reasoning [31] [28] [1], and works using agent communication [3] [36].

(ii) the second axis focuses on the adaptation which is based on a user-centered game design. It uses the player model and/or his profile in order to adapt the game's behavior. We also find in this axis works deals with educational games that are based on learners' models [9]. However, in the context of serious games both perimeters may be involved. Adaptation in this case must be introduced at game design level. This requires: (i) taking into account the players-learners' actions (ii) and the control of the game environment and its components including the virtual characters of the game world.

Behavior modeling

Behavior modeling used in game design is usually focused on the prior planning of the game behavior. We can mention for instance, the use of FSM or rule-based systems [3]. This implies the determination of all possible game actions before its realization.

Some works such as [31] [21] use machine learning or theory of mind concepts. These models assume that the agent: (i) may possess new states or unanticipated behavior that is usually not controlled by the system (ii) and has not a time-bounded execution cycles. This can affect the processing time and the game responsiveness. In addition, in the case of serious games, a new generated agent behavior, which is not controlled by the system, can have an undesirable effect on the player's motivation or on his training programs.

Adaptation of individual /collective behavior

The game adaptation in most works acts generally on the collective behavior of agents [1] [36] [37].

In-line/off-line adaptation

The adaptation technique is commonly used during the game session (i,e in-line adaptation) [21] [31] [36], and sometimes outside it (i,e off- line adaptation) [3] [28]. However, this may have an impact on the system's or the agent's responsiveness in the game. This can depend on the processing and reasoning time during the game session.

Open/closed-loop system

The game's state management is related to the behavioral modeling technique used during the game design. We use an open-loop controller system [21] [3] when we have a pre-planned behavior. In contrast, in closed-loop systems [1] [31], the agent uses its feedback to determine its future states or to learn from its experience [31]. This learning process can be fulfilled through machine learning or communication between agents [1] [36].

The use of closed-loop system through agent programming allows to the agent to learn from its previous experiences during the serious game session.

Parameter adaptation, mono / multi player adaptation

Finally, we can also deduce that the behavioral adaptation of agent-based games concerns only the individual player's behavior (i.e single-player adaptation). Moreover, the adaptation technique in these works use only players' performance as an adaptation parameter without taking into account the user's cognitive and motor abilities and his motivations.

In addition, in the case of serious games, some challenges such as the heterogeneity of users (eg. in terms of age, skills training, and health) and the training programs diversity, should be taken into account while adapting the game behavior.

5. DISCUSSION

Adaptation can play an important role in proving the effectiveness of serious games in various fields such as in the therapy context. The question of acceptance and effectiveness of these games is subject to the possibility of involving player in his training program via the game. This can be performed through the realization of games that can be adapted to the player needs, preferences and cognitive and motor abilities. In addition, the introduction of collaborative and social aspects in the game can also have a positive effect on the motivation of players-learners.

In order to meet these constraints during the design process of adaptive games, we have proposed in this paper to study the state of the art of behavioral programming in adaptive game context. This study which is based on an agent approach, seeks to answer to the key issues concerning the design of adaptive serious games.

As a perspective, we hope to develop this work by specifying an agent programming model dedicated to collaborative therapeutic games. The aim is to overcome most game design challenges which have been discussed in this paper.

The desired programming model should be reusable and flexible. In fact, it should be appropriate for different types and genres of therapeutic serious games. The aim is to reduce the cost of developing such games. In addition, the model should take into account the maintaining of responsiveness and process time performed by the adaptation process, using a time-bounded game agent. This is the purpose of guaranteeing the performance and service quality of the game.

6. CONCLUSION

This paper presented and analyzed the principal works that deals with behavioral programming in agent-based adaptive games. The analysis of the state of the art shows that the adaptation process is generally based on the player's performance and style. However, in the case of serious games, taking into consideration the user's cognitive and motor ability as well as his preferences may be important as it affects his motivation and consequently the usability and the effectiveness of the serious game.

Moreover, these studies deal only with the adaptation of individual behavior in a context of single-player game. Indeed, adaptation in a multi-player context raises important challenges such as managing the consistency of players-learners' views of the game. Thus, this leads us to identify the development of adaptive multiplayer serious games as a major challenge to be addressed by the community in the medium term.

7. REFERENCES

[1] Amato, C., Shani, G. 2010. High-level Reinforcement Learning in Strategy Games. In Proceedings of 9th International Conference on Autonomous Agents and Multiagent Systems AAMAS (Toronto, Canada, 2010), 75-82.

[2] Andresen, k., Gronau, N., 2005. Seeking Optimal IT Strategies by the Determination of Adaptability in Domain-Specific Software Applications. In Proceedings of IJCAI Workshop on Reasoning, Representation, and Learning in Computer Games (Brazil, 2005).

[3] Bakkes S., Spronck P. and Postma E. 2004. "Team: The team-oriented evolutionary adaptability mechanism", Entertainment Computing journal ICEC 2004, 135-148.

[4] BioWare. 1998. Baldur's Gate. published by Black Isle Studios and Interplay.

[5] Black & White. 2005. Official site, http://www.ea.com/uk/. June 2010.

[6] Brown, E., Cairns, P. 2004. A grounded investigation of game immersion. In CHI-2004 Conference.

[7] Christian, M. 2002. A simple inference engine for a rule-based architecture. In AI Game Programming Wisdom. Charles River Media Inc.

[8] Civilization, 2010. Official site, http://www.civ4-lejeu.com/home.htm. June 2010.

[9] Conati, C., Manske, M. 2009. Evaluating Adaptive Feedback in an Educational Computer Game. Proceedings of the 9th International Conference on Intelligent Virtual Agents (Amsterdam, the Netherlands).

[10] Csikszentmihalyi, M. 1991. Flow: The Psychology of Optimal Experience, New York : Harper Perennial.

[11] Eiben, A.E., Smith, J.E. 2003. Evolutionary Computing Tutorial Part 1: What is an Evolutionary Algorithm?

[12] Fairclough, C., Fagan, M., Namee, B. M. and Cunningham, P. 2001. Research directions for AI in computer games. In AICS Conference.

[13] Fu, D., Houlette, R. 2004. The ultimate guide to fsms in games. In AI Game Programming Wisdom 2. Harles River Media Inc.

[14] Gerstmann, J. 1999. Unreal Tournament: Action Game of the Year, GameSpot.

[15] Goodman, R., Shelley, B., and Sullivan, B. 1997. Age of empires, developed by Ensemble Studios, published by Microsoft Game Studios.

[16] Gordon, E., Logan, B. 2003. A goal processing architecture for game agents. In Proceedings of the second international joint conference on Autonomous agents and multi-agent systems.

[17] Half-Life. 2003. Official site. www.half-life2.com. June 2010

[18] Hernandez, C., Meseguer, P. 2005. LRTA*(k). In Proceedings of the 19th international joint conference on Artificial intelligence, 1238-1243.

[19] Hubbard, C. 2005. F.e.a.r. first encounter assault recon, developed by Monolith Productions, published by Vivendi Universal.

[20] Jaeho, L., Huber, M. J., Durfee, E.H. and Kenny, P.G. 1994. UM-PRS: An Implementation of The Procedural Reasoning System for Multirobot Applications. In Proceedings of CIRFFSS-94, 842-849.

[21] Kim, I.C. 2004. “Constructing a BDI Agent to Deploy in an Interactive Computer Game Environment”. Journal of Artificial Intelligence: Methodology, Systems, and Applications, 381-388, 2004.

[22] Mateas, M., Stern, A. 2004, “Life-like Characters. Tools, Affective Functions and Applications”. Springer. Chapter A Behavior Language: Joint Action and Behavioral Idioms.

[23] Miikkulainen, R. Bryant, B.D., Cornelius, R., Karpov, I.V., Stanley, K.O. and Yong, C.H. 2006. “Computational intelligence in games”. Computational intelligence: principles and practice. In Yen GY, Fogel DB edition. IEEE Computational Intelligence Society, (Piscataway, 155–191).

[24] Neerincx, M.A., Cremers, A.H.M., Kessens, J.M., Leeuwen D.A. van and Truong, K.P. 2009. Attuning speech-enabled interfaces to user and context for inclusive design: technology, methodology and practice, Universal Access in the Information Society, 109-122.

[25] Orkin, J. 2004. Symbolic representation of game world state: Toward real-time planning in games. Technical report, AAAI -Workshop on Challenges in Game AI.

[26] Perlin K., Goldberg A. 1996. Improv: A System for Scripting Interactive Actors in Virtual Worlds. In Proceedings of the ACM Computer Graphics Annual Conference, 205–216.

[27] Quake. 2005. Official site, http://www.quake3arena.com/. June 2010.

[28] Ram, A. Ontanon, S. and Mehta, M. 2007. Artificial intelligence for adaptive computer games, Twentieth International FLAIRS Conference on Artificial Intelligence.

[29] Silverman, B.G., Johns, M., Cornwell, J., O’Brien, K. 2006. Human Behavior Models for Agents in Simulators and Games: Part I: Enabling Science with PMFserv. Presence Teleoper Virtual Environ, 139–162.

[30] Spronck, P., Sprinkhuizen-Kuyper, I. and Postma, E. 2004. “Online Adaptation of Game Opponent AI with Dynamic Scripting”. International Journal of Intelligent Games and Simulation, edition N.E. Gough and Q.H. Mehdi, Vol. 3, No. 1., 45-53.

[31] Stanley, K.O., Bryant, B.D., Karpov I., and Miikkulainen, R. 2006. Real-time evolution of neural networks in the NERO video game. In Proceedings of the national conference on artificial intelligence (Menlo Park, CA, Cambridge, MA., London, 2006). AAAI Press; MIT Press.

[32] Stout, B. 1996. Smart Moves: Intelligent Path Finding. Game Dev Mag.

[33] Sutton, R.S, Barto, A.G. 1998. Reinforcement Learning: An Introduction. (Cambridge, Massachusetts). The MIT Press.

[34] Suzuki, Y. 1994. Virtua fighter 2. Developed and published by Sega.

[35] Sykes, J. and Federoff, M. 2006. Player-centred game design. CHI'06 extended abstracts on Human factors in computing systems.

[36] Tan, C.T., Cheng, H. 2007. Personality-based Adaptation for Teamwork in Game Agents. In Proceedings of the Third Conference on Artificial Intelligence.

[37] Tan, C.T., Cheng, H. 2008. A combined tactical and strategic hierarchical learning framework in multi-agent games. In Proceedings of the 2008 ACM SIGGRAPH symposium on Video games (Los Angeles, California, 2008), 115-122.

[38] Watkins, C.J.C.H., Dayan, P. 1992. “Q-learning”. Journal of Machine learning, 279-292.

[39] Wexler, J. 2002. Artificial Intelligence in Games: A look at the smarts behind Lionhead Studio's "Black and White" and where it can go and will go in the future, University of Rochester.

[40] Yildirim, S., Stene, S.B. 2008. A survey on the need and use of AI in game agents. In Proceedings of the 2008 Spring simulation multi-conference. SpringSim '08 (Ottawa, Canada), 124-131.

[41] Yue, B., de-Byl, P. 2006. The state of the art in game AI standardization. In Proceedings of the 2006 international conference on Game research and development.