Writing for video games

Viviane Gal, Cécile Le Prado, Stéphane Natkin, Liliana Vega.
CEDRIC/ CNAM
{gal, leprado, natkin, lvega} @cnam.fr

Abstract. This paper is a survey of the process and technology used for video games and tackles the problem of writing them. The first part of the paper addresses the general state of the art in the design of video games : market and technology constraints, production process. The second part is devoted to the writing of games practices and technology constraints.

1 Rationales

In 1999 the CNAM (Conservatoire National des Arts et Métiers), the Universities of La Rochelle and Poitiers, in collaboration with IRCAM (Institut de Recherche et Coordination Acoustique/Musique) and the CNBDI (Centre National de la Bande Dessinée et de l'Image) decided to create a new postgraduate training in video games design and development. To define the contents of this training, the authors of this paper interviewed representative of all the main activities involved in the game industry. This work has been completed by a bibliography analysis. This paper is a survey of the result of this work and in particular in the field of game design. To understand the specific features of game design in video games, it is necessary to presents general aspects of the game industry process and technology. The first section of this paper is devoted to this subject. In the second part we focused the presentation on game design. The last section discusses tools used to implement games.

2 Game industry state of the art

2.1 A few words about the game industry

The computer game industry is one of the most important fields of the interactive multimedia industry. A 21.1 Millions $ revenue is forecasted in 2003 [1]. The market is split between PC and console games (PS, PS2, Xbox, GameCube…) with two special cases : game for small console (Gameboy, Palm.) and Online persistent games (like Everquest), which are sold on a subscription basis.

In the last twenty years game editors and distributors were generally also producing most of their products and developing the software tools used for games. There is an increasing trend to separate, like in more classical audio visual fields, these three domains into editing company, studio and specialized software editors.

Console manufacturers are the great winners of the game industry: each time a console game is sold, the manufacturer gets royalties. Moreover the console manufacturer controls the game design and development.

2.2 Genre

What is a computer game? There are numerous answers to this question. In the purpose of this paper, we consider the word game from its classical meaning. As a consequence playing with computer games is mainly an entertainment, and the main goal of a game designer is to give fun to his public. This definition excludes, for example, computer education systems that may include some games as an artifact to teach.

Game writing style is related to the genre of the games considered. Numerous possible classifications of games according to various criteria have been proposed in the literature. In this paper we will derive our taxonomy from

the one of Rollins [2]. This classification relies on the main focus of the game. It is not the most widely used (in particular, it is not used by the games press), but it is rather clear and relates to the notion of genre in literature:

- **Action games** lead the player to push as fast as possible à lot of frantic buttons. Fighting games (Beat them all) are a good example of this genre.
- **Adventure games** are probably the ones which are the most related to classical audiovisual scenario. The player is the hero(in) of a complex scenario. Metal Gear Solid 2 is a typical adventure game.
- The main features of **Strategy games** are the complexity of the decisions taken by the player in a simulated or fantastic politic, economic or military universe. The Sims and Black and White are very elaborated strategy games.
- **Simulation games** lead the player to exercise himself on simulated sport or physical device (Plane, car, skateboard…).
- The aim of **Puzzle games** is to solve a hard analytic challenge. The computer versions of classical games (like chess) are the basis of Puzzle games. But Puzzle games lead the player to be in charge of an investigation. Myst games are an example of the genre.
- The main goal of **Discovery games** is to discover an historical, geographical… problematic embedded in a game, which looks like one of the previous genre. Discovery games are the equivalent of documentary in the audiovisual field. For example, Versailles game leads the player to discover the life of the king Louis XIV through a puzzle game.

According to Rollins we define the style as the way the game is executed. "Duke Nukem and Tomb Raider are both action/ adventure games, but their style make us perceive at different genre". RPG (Role Playing Games) is a mix of action/ adventure/ Strategy games.

3.3 The process

The creation of a game is carried out in four stages:

- Specification and planning
- Pre production
- Development
- Validation and testing.

The conception of a game starts with either an original idea proposed by a studio or a query submitted by an editor (design Tomb Raider 23 or a Donald Duck game). In both cases the first document produced is a short description (the synopsis) which includes:

- The public focussed (hard core gamers…)
- The plat-forms (PS2, PC…)
- The genre (s)
- References (game, books…)
- A first draft of the planning

Starting from this document a studio will produce a specification that includes the main original features of the game and a cost evaluation. It may include also a first prototype of the game. At the end of this first specification phase (called for example Concepting in the Ubi Soft terminology) a decision to carry on is taken. It is mainly based on marketing consideration.

The pre-production phase is mainly devoted to the game design, the production of a significant prototype, and the refinement of cost and planning evaluation. A second carry on decision is taken at the end of this step.

The development phase includes the creation of all the elements of the game (images, sounds, video, programs) and their integration.

The first step of the validation is a functional testing (Alpha tests), which is an evaluation of the quality of the game (principles, gameplay, man machine interface, aesthetic qualities, and ability to respond to the market needs). If this first validation is passed, the second step is a debugging process (Beta tests) done by a specific team.

All this process is performed as iterations between prototypes enhancement and evaluation. At any time the project can be aborted, in particular at the end of the Alpha tests, when the economical goals may not be reached.

The whole process takes from six months (small Gameboy games) to three years, and cost up to 1, 5 millions dollars.

3 Writing for games

3.1 Introduction

Writing for games is a rather difficult task. Of course it is an interactive composition and, as in other fields of open work, the author must leave a controlled freedom to the player. But, in the opposite of the art installation field or interactive music composition, marketing goals drives the game industry. Game is mainly entertainment, hence the player must solve non-trivial but not too complex problems, leading to a succession of goals in a reasonable amount of time. The player must feel in an open interactive world, but should be driven to the game solution. To solve this paradox the game industry has invented several techniques derived from game theory and object oriented specification, which are summarized in this section.

3.2 Game and level design

A game is first and foremost an imaginary universe. Then the first step of the game specification is to define the main aspects of this universe: Epoch and style, context of the game, goal to be reached, main types of objects involved, user perception of the game… This part of the game definition is called the game design.

Next steps of the game specification are called levels design. A level of the game is a mix of a virtual space, a set of puzzle to be solved in this space and the main actions to be done by the player to reach a given goal.

3.3 Scenario

There is an open discussion in the world of game design about story telling and scenario. The notion of scenario comes from the movie world and is related in one hand to the idea of story telling and in the other to a sequence (and time driven) of scenes. A game can not be only a scenario, as the player must always be the main actor of the scene. Rollins argue that a game designed as a story should better be developed as a movie. The importance of the scenario is related to the genre of the game. For example a scenario often drives adventure games when in action or simulation games the scenario is mainly a piece of the context. But there is always a scenario in a game. It may be reduced to a sequence of goals to be reached or a path, which must be followed. This is even true in a car game where during a race the player must follow the circuit.

But, even in a given genre, the importance of the scenario depends on the style of the designer. Consider two action/adventure games: Tomb Raider and Metal Gear. In the first one the scenario is reduced to almost nothing. The story is anecdotic and the characters (in particular the heroin Lara Croft) are as fat as the paper of a cigarette. Tom Raider is mainly an action game. Metal Gear is also an action game. But one of the implicit goals

of the game is to understand the complexity of the universe and story embedded in the game. Metal Gear includes long non-interactive animations (cinematics), able to drive the player in this universe and define the milestones of the game. Moreover, in Metal Gear solid 2, the player is in the first part of the game a mythic hero (Solid Snake). He becomes, in the second part of the game, a new "white beginner" character (Raven). The designer, Ideo Kogima, explains, in the making off, that this choice allows the player to understand the complexity of Snake character, taking a third person point of view. In the same making off, Kogima says that he has always wanted to make movies…

It is out of the goals of this paper to take an aesthetic point of view on this discussion. In the sequel we define a scenario as a specification, which forces a partial, or total ordering in the game execution. It may be a classical description of a sequence of scenes. But, in general, the game design scenario is a description of the main game phases and the navigation between phases. The level design scenario is a positioning of objects in a maze and an associated puzzle to be solved. It induces a partially ordered set of actions that the player must perform to end the level.

3.4 Components of the game design

The main components of the Game Design are:

- The context of the game (epoch, style, historical or mythical references)
- The global scenario (Topology, global navigation graph, main characters, nature and hierarchy of the levels)
- Main features of the game, that is to say what makes the game unique. Features must be classified as fundamental aspects of the design and chroma.
- The principles of the gameplay : modalities, goals, rules, main strategic choices
- The image and sounds charts (look and feel)
- The ergonomic principles: Interface, game learning, saving and loading options…
- The classes of objects in the game. The concept of object in the game design must be understood as in object oriented specifications. Classes of objects are defined by their names, properties and interface specified as the actions (methods) which can be performed on an object. Classes can be constructed using (more or less formally) heritage and polymorphism operators. Objects can be active: It may include an automaton, which is used to specify the autonomous life of the object.

We will consider to opposite example: The Sims and Gran Turismo.

Sim city, designed by Will Wright, started as a simulation based on the urban planning simulation derived from Jay Forrester system theory. W. Wright took the idea of architecture being a functional solution to life and coupled it with his fascination for 3D home architecture products. Objects in Sims are all the main things you can find in a life in a town: buildings, the people classified by sex and profession, the animals, the materials to built houses and monuments, TV sets, phones, birds…. Each object is associated with all the main actions (methods) which can be found in the real life. Make a phone call, buy a car, built a house, fall in love… Ambient objects are used to define things like a changing weather or ambient sounds. Other objects are used as elements of the staging, for example virtual camera allowing several point of views of the same scene. The main features of the Sims are the variety and the number of actions that a player can do and the complexity of the embedded simulation model which take into account all theses actions.

On the other hand consider a car racing game such as Gran Turismo 3, in this case, objects are the set of cars, that can be used as the elements of possible racing circuit. The main features on Gran Turismo are also the variety of choice allowed in the context of a car game (cars, circuits, kind of races…), the quality of the physical simulation and the real time graphics.

In both cases objects have numerous attributes: geometry and appearance, variable parameters (strength, speed, robustness), type of actions allowed on the object (move right, jump, ring, explode…)…

3.5 Level Design

Generally the level is first defined by the geometry of the space: a given maze, a race circuit. Then the level designer chooses the positions and actions associated with the objects in this level. The goal is either implicit (win the game on this circuit) or explicit (find three elements of a totem to open a gate). In both cases the player is conduct by an implicit scenario, which limit the number of possible effective actions.

To keep the sensation of freedom, several solutions are used: first, a set of independent actions can be performed in any order, in more complex games the player can pursue, in the same space, several goals in parallel. This is the case in Sims, where the player is able to take a lot of political and social decisions, but some message inform him, when the people or the city is going wrong as a consequence of these decisions. In the car game, you can drive, as you want as long as…you follow the circuit until the end of the race!

The puzzles and levels must be designed in a way such that the player doesn't feel trapped in a cycle. The degree of difficulty increases gradually. As a consequence levels are generally designed and developed in an increasing order of complexity.

Cinematics are placed in between levels to present information needed at the next step, while the player relax, enjoys his accomplishments and recovers for the next challenge.

3.6 Immersion and Interactivity

Introduction

The feeling of immersion either on a perceptual emotional or intellectual point of view is one of the key factors of a game:

"To enter a game, I must be catch by, the image, the touch and finally the gameplay",

explains Frederic Raynal, the game designer of Alone in the dark. Hence the "look and feel" is a main feature of the game. This is the art peace of the game, which is out of the scope of this paper. But the feeling of immersion and the technology constraints are fundamental element of the game design. In this section we discussed first the gameplay analysis. We consider then several important ergonomic factors: learning process, saving and loading options. At least, we point out some aspects of the peripherals.

Gameplay

A gameplay is first defined by a hierarchy of goals given to the player. Goals can be explicit: in GTA 3 the player is a gangster and he must perform a sequence of missions (kill somebody, destroy a bar…). In the game Black and White, the player is a God of a small world. His main goal is to be recognized and honored by his people. To reach this goal he can be a good God, helping people in their life or a kind of devil, known for his cruelty. This goal is implicit. The player discovers it when he understands that his power increases with the number of believers. But in the same game each level is defined by an explicit goal written on a message (help a shepherd to find his sheep). To reach the goal the player must take some decisions that are the core of a gameplay.

Decisions are classified according to their long or short time effects (strategic or tactic). The player must understand quickly (intuitively) which decisions are inefficient. In counter part, a decision must not be trivial. It must have some positive aspects and some non predictable counter parts: Building a castle needs to spend some resources which may be useful in the sequel, choosing a devastating big gun may slowdown the player for the gun is heavy.

This dominant strategy problem is well known in game theory. To avoid dominant strategy, the effect of choices must be non transitive. A classical example of this kind of choices is the paper/scissors/ stone game. Paper wins against stone, stone wins against scissors, and scissors wins against paper. A way to construct non-dominant strategy is to define decisions (made by the player and the computer) as a vector of criteria. A decisions A dominates B, if the vector associated with A is, component by component, greater than the vector associated with B. Numerous games (in particular RPG games) use explicitly this representation of decision. Numerous other simple ideas of game theory can be used to design a good gameplay (MinMax analysis, optimal mixed strategy, game tree.).

The other problem of gameplay is to allow the player to reach its goal in a fair and not too complex way. As the computer knows everything about the state of the universe, it can easily win in all situations. Moreover it can decide at any time that the game is over. This kind of gameplay will be perceived as unfair. The gameplay must be associated with a learning process: when the player loose, he must be able, from the information given by the game, to understand his mistakes.

Ergonomic principles and learning process

As almost all virtual reality system, a game is an implicit learning system. Understanding the laws of the game universe is a part of the fun. It must not be too simple to keep some mystery. Excessive information leads the player to feel guided. But the learning process must not be too complex to allow a discovery by practice. This has already been point out from the gameplay point of view, but it is also true from a man machine interface point of view.

Each time a player executes an action, he is expecting something to happen as a consequence. Some game designers suggest that every interaction with the game system should have an answer [3]. Most of the player actions must have an influence in the game universe. In counter part, the consequence must not be always entirely revealed. But, as the learning process must be implicit, the consequence on an action must be logic. Hence all classical ergonomics principles of a computer interface (the same action lead to the same consequence, reflex actions must be intuitive,…) apply to games.

When a player open a manual to understand a game, It can be considered as a failure of the design. The training must be an integrated part of the game and must not disturb the player too much from its feeling of immersion. Hence several levels of on line training are included in the game design. For all of them the goal is to teach the user how to interact with the game or give him additional information. Some of them, the more classical, use contextual menus, other might appear as icons or characters activated by certain actions and the latest ones use some kind of sound effects to interact with the user. For example in Black and White two little characters (good and bad consciences) provides spoken contextual advice. There are also a lot of little notes written all around the scenery.

The game and level design scenario define a second level of training. The first levels of the game are built to teach and introduce the player in the game. In contrast, they are also training spaces (isolated rooms, training islands, etc) where the player can go at any time to exercises him self. In the same way, actions and simulation game provides also a particular playing training mode. It is a special purpose level where the goal of the player allows him and the game play to measure his abilities. The game execution can be altered according to the results of these training sessions.

Loading and saving options

As a consequence of the hardware architecture, in particular console architecture, all operations on persistent memory can slowdown the game. In order to keep the feeling of immersion several techniques are taken into account for saving, backups, loading, and the exchange of CD operations. CD exchange and saving options are mainly game play driven. Different politics are used: some games have “choke point” which controls the game progression. To save the game the player must reach "physically" a choke point. Some choke points induce a commitment: data are saved and roll back before this point is not allowed. In present games, several choke points can be kept. In all cases all the work done between choke points is lost. This option is mainly due to the lack of space on WROM memories. Hence the state variables must have well defined values at saving points. As a consequence the time to save the game is small enough and the player includes his race to a choke point in the game challenge. On PC and Xbox games (that have a hard disk) games may have more elaborated saving options (automatic or controlled saving at any time). For example in the Sims the interface and saving menu offers several options. The player can save at any time and, as a consequence quit the game when he wants**.**
For the same reasons there are no good solutions to the loading process on consoles like the PS2. The lack of memory and the bus characteristics induce a delay each time the player is moving from a scene to one other. This is unfortunately the case for action and adventure games evolving in a real time 3D world which descriptions include numerous textures. The loading delays are particularly disturbing for this king of games. In this case the game design play on the scenario (loading at still point of the story, at the end of a fight) and a transition based on fixed images, environmental sounds and music.

4 Tools

4.1 Images, Sounds, Touch

Numerous manufacturers have developed low cost immersion peripherals (goggle, guns, steering wheels, guns, and elaborated paddle for PC, dancing carpet…). Out of Japan it has been generally a commercial failure. There are several reasons to this fact. First there is a lack of interface standardization. Hence a peripheral can be used only with one or two games. Even if it is relatively cheap such hardware cost a quarter or a third of the console price which is too expensive. The use of goggle and headphone is not adapted to play during hours, like hard core games do.
From a visual and audio point of view the development of the home theater may change the situation, at least for console games. If a huge screen and a 5.1 audio system replace the TV set, the feeling of immersion will be greatly facilitated.

In term of interface PC games sensors and actuators have a less privileged place than console games. Of course a PC can use a wide monitor, good and well-positioned HP's and a fine paddle. But the game must be designed for a PC with a 15 inch monitor, an old graphic accelerator and sounds card, one HP and a mouse. On console the interface is the TV set, the hardware architecture is determined so as the paddle. It is interesting to point out that console manufacturers lay down ergonomics rules related to the use of the paddle to the editors. Immersion must rely on subtle relationships between images, sounds and game play.

Most of actions/adventure games developed for PC and the last generation of consoles used real time 3D animations (using several virtual cameras). This is considered in a highly interactive or a scenario oriented game as an essential feature. This is the consequence of marketing constraints and a technical goal: the frame refreshment rate must be at least 30 frames/s. Whatever game you edit for a console, its manufacturer requires a frame rate up to 60 frames/s.
The current generation of graphic accelerators can handle images composed with several thousands of polygons in the 60 frame/s throughput. Several millions of polygons are expected in the next generation. When a PC game

runs on a smaller configuration, game developers have adapted from VR technology scalability techniques: the game adapts the image definition according to the configuration. On strategy games 3D isometric view is still the main option, for it allows a simpler panoramic consideration of the situation. But this may evolve in the next years.
3D sound is considered as an interesting ability to increase the feeling of immersion. 3D technology and the corresponding development tools are already available (EAX/EAGLE, Sensora Sound Engine, OpenAL and DirectX library) for PC and the last generation of console. It has been successfully experimented on several games. The HP problem is the limiting factor to this trend. An other factor is the repartition of the CPU allocated to the different functions. Image synthesis is the most expensive function. This limits the audio, simulation and AI features that can be used.

Platform	**Graphics**	**Sound**	**IA and other**
PC (game)	60	30	10
Console	75	15	10

Fig. 1 Typical percentage of relative CPU consumption by functions in a game

4.2 Artificial Intelligence

Video games developers use to call "Artificial Intelligence" (AI) any kind of techniques that gives the feeling that the computer is, as a player, not too stupid. A well-known example showing the lack of "AI" is the memory less behaviour of a soldier in one of the first versions of "Return to Castle Wolfenstein". When the player opens a door, he sees the back of an enemy and he can shoot him. But if he closes the door and then opens it twice, the enemy is still waiting to be killed. Hence the basic AI in games is to used memory variable and more generally to associate to each situation a finite automaton. More elaborates AI techniques are still used only in a minority of games. They are two reasons to this situation. First, as we have already point out, a low percentage of the CPU time is allocated to AI functions [8] [9]. This explanation of the poor level of AI is less and less justified [8], as the power of consoles is increasing quickly and as graphic accelerators can execute complex graphic functions. The second reason is related to the game play. Remember first that a computer playing does not need to be intelligent (that is to say able to react to any situation), but has to look as intelligent. As all the game is designed to drive the player to the victory in a predictable time, situations can be predicted. The player may open the door behind the soldier once, twice at least three times, not more. An automaton with five or six possible states will give a sufficient feeling of intelligence. Moreover if a more sophisticated random algorithm is used to define the computer behaviour, it may become very difficult to control or to test.

One may say that very efficient AI algorithms have been developed in the field of classical games (chess, go…). But the knowledge of chess game is the result of thousands of years of thinking. The state of the art in computer chess is the result of thirty years of research on a universally known game. Most of video games have a lifetime of six months. Each video game has its own rules, always rather simple. The difficulty of a game is generally not related to the rules, but to the ignorance of these rules by the player. [11] [12]

As stated before the main "AI" in game is the use of automaton or finite state machine. It is used, for example to control the enemies in first person shooter games (Doom like).

Standard graphs and relations algorithms, like R*, are used to simulates realistic movements and paths finding.

Random automata or fuzzy logic is used to avoid predictable or recurrent behaviour. The main use is to choose randomly between several strategies in a given situation. It can be combined with game theory mixed strategy point of view to built nice game play. This can be used in fight games (Beat them all) and in action games in order to manage the behaviour of the enemies. The game Civilization, real time strategy game, uses this technique. It is also used in collective sport games to specify the variations around the collective behaviour of a team.

The more elaborate games that use this classical point of view are strategic simulation games. The core of the gameplay is an event driven simulation model, which is a simplified version of a well-understood scientific simulation. The Sim City game was derived from works on urban simulations. Hence, even if the model is complex, it is always under control.
Several military games, among them Conflict Zone by MASA group use a layered structure as AI architecture. Each layer corresponds to a level of decision (General, Captain, Troops). They are most commonly named strategic, operational and tactical levels [9]. Decision priorities are ordered according to the levels. These complex genre of AI are derived from models developed for military applications.
During the last years more sophisticated AI principles has been used :

The artificial life allows splitting the specification of a complex global simulation into smaller ones. Each "object" of the game is specified as a random automaton sending and receiving stimuli from other objects. The global behaviour is the results of the interactions between local ones. The game "The Sims" implements these techniques. Artificial life is used to simulate phenomena such as the movements and behaviours of characters at home, in town, in holidays, ... Artificial life does not need more processing power than a global simulation, it is well suited for object oriented specification and distributed games. It has the same limitation than the global simulation technique : Only experimented artificial life model can be used in a commercial game.

The neural networks can be used in order to update the AI model to take into account the player's progress and to adapt the difficulty to the player's style. The game rules seem to change continuously. "Fields of Battle" is a game that implements the neural network technique.

Genetic algorithms (a part of evolutionary algorithms) are the fashion in the video game world. It seems to be a promising way to create dynamically new types of objects (monster, weapons, sounds…) in the game. They are not used in practice. They are difficult to control (evaluation functions, mutation and cross over probabilities) and they still require too much processing power [8].

4.3 Architectures

Several companies (Infograme, Darkworks, Virtools…) have built a programming environment, which allows game development following the previous steps. Objects specified in the game design are implemented by the programmers and the team of graphic artists as classes in an object oriented library using all the facilities of object oriented programming (heritage, polymorphism…). The level designer defines the geometry using a standard 3D tool such as 3DSMax or Maya. A scripting language is then used to specify the level in term of objects in the space.
Virtools Dev 2.0 offers a development environment for interactive applications. It has a graphic user interface where objects can be placed and manipulated easily. 3D objects are treated independent from their data and behavior. Other game creation systems are available in the market : *The Game Factory* by Click Team, with a storyboard, a level and a event editor similar to the one in Director of Macromedia and *Pie 3D GCS* by Pie in the Sky with a layout editor, a paint program and an engine to run the game.

The implementation of game relies generally on software called a game engine. A game engine is first a set of software libraries. Each library performs a set of functions needed to code the dynamic and interactive behavior of the game. General purpose game engine includes a 3D graphic engine used to code interactive animation, a sound engine which includes sound synthesis, effect and spatialization functions and more specific libraries like Artificial Intelligence engines used to code the gameplay and physic engines used to simulate physical systems (like cars for example).

<table>
<tr><th colspan="4">Architecture Levels</th><th>Examples</th></tr>
<tr><td colspan="4">Level design scripts editors</td><td>God.move(right, 2);Wait_Event;
On button.click God_Anger:=new(thunder)
God_Anger.lightning, God_Anger.sound</td></tr>
<tr><td colspan="4">Game classes</td><td>class thunder
methods: lightning , sound</td></tr>
<tr><td colspan="4">Game engine library: general purposes game oriented libraries</td><td rowspan="2"></td></tr>
<tr><td>Graphic engine</td><td>Sound engine</td><td>Physic Engine</td><td>IA engine</td></tr>
<tr><td colspan="4">General purpose multimedia libraries
(Direct X, Open GL, Open AL…)</td><td>GlMatrixMode(); alsourceplay(source1)</td></tr>
<tr><td colspan="4">Operating system</td><td>Windows, PS2 Monitor…</td></tr>
<tr><td colspan="4">Hardware</td><td></td></tr>
<tr><td>Central Processor, memory…</td><td>Graphic accelerator</td><td>Sound card</td><td></td><td>PC, PS, GameCube, XBox</td></tr>
</table>

Fig. 2: Software architecture for games

When a collection of game relies on the same universe (Ubi Soft Ray Man games, for example), the studio can define a specific game engine at the game classes level.

In several game engines there is also a software monitor, defined by the level design script, which schedules either on a synchronous mode or asynchronous mode. On console platform the synchronous monitor approach is almost always used. On PC (as Windows 95,98… are highly asynchronous operating systems) the asynchronous approach is possible.

The use of portable game engines [1] allows minimizing the work to be done to create multi platform games. The Criterion engine[2] is a typical example of game engine. It contains a 3D Graphic engine, a sound engine originated from Sensora, a Physic engine originated from Karma and an IA engine. A game developed with Criterion game engine can be, in principle, ported on PC, Xbox and PS2 consoles. In practice the differences between the plat-forms hardware induce to adapt or even to change, in certain cases, the design of games.

[1] http:// 3dgraphics.about.com/cs/gameengines/
[2] http://www.renderware.com/

6 Conclusion

There is a process used for writing games, which seems to come out from our analysis of the present practices. It relies on an object oriented game design and a scenario oriented level design. We think that it is important to better understand and formalize this process. This could lead first to an analysis method for games, to be compared to the scenario analysis techniques used in movies. From a training and critical point of view this is fundamental.

From a more technical point of view there is a lack of integrated writing tools, taken into account the game design step. We think that methods and tools used in object oriented design may be adapted to this goal. UML (unified method language) method and tools are good candidates. Work in similar fields (behavioral scenario) has been already carried out[3].The objects specified in the game design in an UML like specification could be in one hand coded and tested using software engineering techniques and on the other hand used on a level scripting tool like Virtools. Working on this field is the next step of our research.

References

[1] V. Gal, C. Le Prado, S. Natkin, L. Vega, Processus et outils utilisés pour la conception et la réalisation des jeux vidéo, Rapport CEDRIC, A paraître, juin 2002.

[2] A. Rollins, D. Morris, "Game Architecture and Design", Coriolis Ed, 2000

[3] Bob Bates, Game Design, The art and business of creating games. Ed Prima Publishing, 2001.

[4] Grigoire Burdea, Philippe Coiffet, La Réalité Virtuelle, Hermès 1993.

[5]Ariane Mallender, Ecriture pour le multimédia, Dunod 1999.

[6] J. Weske, "Digital Sound and Music in computer games",
http://tu-chemnitz.de/phil/hypertext/gamesound/
Web game developers site: www.gamasutra.com

[7] D. Johnson, . Wiles, “Computer games with intelligence”, 2002
http://www.infenv.csee.uq.edu.au/games.pdf

[8] S. Woodcock, “Game AI : The state of the industry”, 1999
http://www.gamasutra.com/features/19990820/game_ai_01.htm

[9] S. Woodcock, “Game AI : The state of the industry”, 2000
http://www.gamasutra.com/features/20001101/woodcock_pvf.htm

[10] Get artificial life !
http://ai.about.com/library/weekly/aa032700a.htm

[11] J.P. Ponsard, Logique de la négociation et théorie des jeux, Ed. de l’organisation, 1997

[12] B. Guerrier, La théorie des jeux, Economica, 2002

Games

Rayman, Ubisoft, 1995 (PS1)

Silent Hill 2, Komany 2001 (PS2)

Gran Turismo 3, Sony, Polyphony Digitals, 2001 (PS2)

Jack and Dexter, Sony, Naughty Dog, 2001 (PS2)

Metal Gear Solid 2, Komany, 2002 (PS2)

Black and White, Electronic Arts, Lionhead Studio, 2001 (PC)

Alien vs Predators, Fox Interactive, Sierra, 2002 (PC)

Sim city, Maxis, 1989 (PC)

The Sims and add ons, Electronic Arts, 2000, 2001, 2002 (PC)

Doom, ID Software, 1993 (PC)

Civilization, Coregames, 1989 (PC)

[3] http://www.irisa.fr/siames

Quake, ID Software, 1995 (PC)
Fields of Battle, 1995 (PC)
Return to Castle Wolfenstein, Activision, 2001 (PC)
Conflict Zone, Ubi Soft, 2001 (PC)