Mario, Warcraft ou Pokemon ? La programmation de jeux vidéo comme motivation dans l'apprentissage des sciences informatiques

Tuyêt-Trâm Dang-Ngoc et Tianxiao Liu
dntt@u-cergy.fr et tliu@u-cergy.fr
Département des Sciences Informatiques de l'Université de Cergy-Pontoise
2, rue Adolphe Chauvin – 95302 Cergy-Pontoise CEDEX

RESUME : La programmation de jeux vidéo est une motivation souvent donnée par les étudiants désireux de suivre une formation en sciences informatiques. Or implémenter un jeu vidéo est complexe et fait appel à beaucoup de notions dont la plus visible est l'Interface Homme-Machine (IHM). Cette dernière fait souvent obstacle à la proposition d'un sujet de projet portant sur le jeu vidéo. Pourtant, les jeux vidéo font appel aux notions fondamentales (algorithmiques, bases de données, réseaux, intelligence artificielle, etc.) des sciences informatiques. Nous montrons dans cet article que la plupart des jeux vidéo peuvent néanmoins être implémentés avec une IHM minimale qui pourra ensuite être étendue suivant la motivation des étudiants. Dans cet article, nous avons catégorisé les jeux existants suivant leur genre et les avons classés en fonction des compétences visées tout au long de la formation informatique à l'université de Cergy-Pontoise. Nous montrons comment nous avons mise en place des solutions pour encadrer les étudiants dans leur réalisation de projets de jeux vidéo. Les résultats sur 14 ans d'enseignements en projets de jeux vidéo montrent l'efficacité de cette approche dans l'apprentissage des sciences informatiques.

Mots clés : jeux vidéo, apprentissage des sciences informatiques, apprentissage par projet

1 INTRODUCTION

« Programmer un jeu vidéo ! », voilà la motivation souvent donnée par les collégiens qui veulent « *apprendre à programmer* ». Entre console portable, console de salon, PC et maintenant smartphone, 79,5 % des jeunes (15-24 ans) jouent aux jeux vidéo. Et pour 68,2 % plus d'une heure par jour [1]**[2]**. Le jeu vidéo fait partie des activités quotidiennes des jeunes. Mais entre « jouer aux jeux vidéo » et « apprendre l'informatique », la réalité rattrape beaucoup d'étudiants qui se heurtent à la discipline des sciences informatiques avec toute sa rigueur et ses nombreux concepts théoriques.

Implémenter un jeu vidéo est complexe. Comme tout logiciel interactif, le jeu vidéo comporte au minimum un ***moteur*** et une *IHM*. Le *moteur* représente le cœur (algorithmique) de l'application : traitements des données, interactions avec la base de données et les autres programmes, réseaux, intelligence artificielle, etc. C'est aussi ici que sont appliqués les concepts théoriques fondamentaux sur lesquels portent l'essentiel de l'enseignement des sciences informatiques. La deuxième partie d'un jeu est *l'Interface Homme-Machine (**IHM**)*. L'IHM n'effectue pas de traitement, elle se contente d'afficher les résultats des traitements effectués par le moteur et d'interagir avec l'utilisateur. C'est dans cette partie que le programmeur conçoit souvent les interfaces graphiques réalisant : a) les entrées : claviers, mouvements de souris/joystick, mesures de capteurs divers (accéléromètre de smartphone, images webcam, micro, etc.) b) les sorties: affichage du texte et des images, gestion des animations, son et musique, contrôle des actionneurs divers (servo-moteur, activation de LED, etc.)

De par sa fonction, c'est souvent l'IHM qui représente la majeure partie de l'attrait du jeu vidéo pour les jeunes. Malheureusement, sa programmation est souvent complexe et longue pour un intérêt limité en termes d'apprentissage de sciences informatiques. De ce fait, la programmation d'IHM est en général considérée comme secondaire dans l'enseignement de l'informatique et se limitent habituellement à des IHM textuels.

Il est parfois proposé en TP de programmation, des jeux de programmer des jeux simples sans IHM graphique comme les jeux de morpion, de démineur, de sudoku ou de master mind. Par leur aspect ludique, la programmation de ces jeux sont souvent perçus comme un peu plus motivant par les étudiants par rapport aux exercices de programmation traditionnels. Mais les limitations restes évidentes : a) ces jeux laissent peu de place à la créativité de l'étudiant. On ne peut guère diverger du jeu initial. b) les programmes de ces jeux se retrouvent en abondance sur le net, les livres, etc. et sont abondamment commentés. De ce fait, ces jeux ne peuvent faire l'objet de projets, mais juste d'exercices d'entraînement ou d'auto-évaluation.

Enfin, ces jeux ne sont pas considérés comme « si ludiques que ça » par les étudiants. En effet, les jeux auxquels jouent les étudiants actuels sont d'un tout autre genre : Candy Crush Saga, Angry Bird pour les jeux de puzzle ; Zelda, Final Fantasy pour des jeux d'aventures ; Pokémon, Advanced Wars, pour des jeux au tour par tour ; World of Warcraft, League of Legends, ou dans un autre style FIFA, NBA pour les jeux « stratégiques » ; Counter Strikes, GTA pour les jeux d'action en vue subjective. Ainsi, trouver une pédagogie permettant aux étudiants de réaliser de tels jeux concilierait objectif de la formation et motivation étudiante.

Cet article s'organise comme suit : Après une comparaison sur les approches entre l'informatique et le jeu en section 2, nous décrirons en section 3 notre catégorisation les jeux vidéos par type et les relierons en fonction des niveaux de programmation et des compé-

Article publié par EDP Sciences et disponible sur le site http://www.j3ea.org ou http://dx.doi.org/10.1051/j3ea/2015002

tences à acquérir tout au long de la formation. Nous montrerons aussi dans cette même section comment ces jeux sont introduits sous forme de projets. En section 4, nous illustrons la mise en place de la pédagogie sur l'ingénierie logicielle et la gestion de projet pour mieux encadrer les étudiants durant les projets de jeux vidéo. Après un bref bilan pédagogique dans la section 5, nous conclurons enfin en section 6.

2 L'ENSEIGNEMENT DES SCIENCES INFORMATIQUES ET LE JEU

L'enseignement par le jeu a montré son efficacité depuis longtemps [3][4][5] pour l'apprentissage auprès de jeunes enfants et auprès des adultes [13]. Afin de motiver et faciliter l'apprentissage des étudiants en sciences informatiques, trois approches existent permettant de lier jeu et enseignement informatique.

1) Une première approche est l'enseignement de l'informatique de façon ludique à l'aide d'outils simples, intuitifs et ludiques. On peut citer : **a)** les kits d'apprentissage du raisonnement informatique, sans programmation : par des méthodes de type drag'n drop d'icônes ou de composants graphiques représentant des conditions ou des boucles. Les fameux Lego, jouet ludique par excellence, ont dans la version Lego Mindstorm [6] un logiciel de ce type permettant d'interagir et de programmer les constructions réalisées. **b)** Les langages spécifiques relativement simples présentés de façon ludique comme le LOGO [7] /StarLogo [8] (la tortue et l'intuitivité du langage en font un langage accessible aux plus petits), de même que Scratch [9] et Cleogo [10]. Alice2 [11] avec son monde 3D et sa programmation en drag'n drop est également très apprécié. **c)** Des environnements complexes mêlant objets communicants (avec actionneurs, capteurs) et monde virtuel comme dans le projet WISE [12].

Bien que très utiles pour initier l'apprenant à la programmation, ces outils sont soit peu « utilisables » dans un contexte professionnel (les langages ne sont pas utilisables pour des « vrais » logiciels en terme de diffusion et de performance), soient peu évolutifs car limités par les possibilités du langage, soient nécessitent un environnement complexe spécifique [12].

2) Dans une seconde approche reliant enseignement de l'informatique et jeu, on citera les *serious games* [13] dont l'objectif d'apprentissage est de permettre au joueur d'acquérir le raisonnement de la programmation informatique [14]. Ils se basent en général sur une histoire interactive. Lors du récit, le joueur est amené à programmer sous un langage spécifique ou pas afin d'interagir avec son environnement et l'histoire. On cite parmi ce type de *serious games* : LightBot [16], Code Hero [17], Code Spells [18], Codemancer [19], COLOBOT [15], et le mod « Kernel Panic » [20] de Spring. Dans ce dernier, l'étudiant doit réussir à programmer ce qui lui est demandé pour accomplir une mission dans le jeu. Il acquiert donc les compétences en programmation nécessaires, sait programmer dans les principaux langages de programmation et s'initie à certaines bases algorithmiques. En revanche, comme pour tous ces *serious game*, il est lié au moteur du jeu et est constamment guidé par le jeu qu'on lui demande de réaliser (peu de liberté). Dans cette approche, l'apprenant ne programme pas un jeu, mais pour le jeu.

3) Enfin, une troisième et dernière approche est la programmation d'un jeu vidéo pour apprendre l'informatique. En utilisant un framework existant, les étudiants doivent programmer en C++ [21] un jeu de type arcade EEClone [22] ou encore des variantes d'un jeu de type puzzle où il faut aligner des symboles [23].

Ici encore, le langage est imposé, un jeu spécifique est proposé et la programmation s'appuie sur un framework spécifique. Ces approches sont très orientées apprentissage de la programmation et ne permettent pas de mettre en pratique les autres concepts des sciences informatiques (BD, réseaux, IA, algorithmique des graphes, etc.). Enfin, ces approches sont orientées vers un seul type de jeu (voire un jeu bien cadré en particulier) et n'exploitent pas l'étendue des types de jeu vidéo par rapport aux concepts informatique à acquérir.

Nous proposons d'utiliser la programmation d'un jeu vidéo différent à chaque étape de l'apprentissage d'une formation informatique basée sur 5 ans (licence et master). Nous ne modifions en rien l'organisation des modules informatiques, mais nous introduisons des projets consistant à programmer un type de jeu vidéo en fonction des pré-requis et des compétences à acquérir pour le semestre en cours. Nous détaillons la mise en place de notre approche dans les sections suivantes.

3 APPRENTISSAGE PAR PROJET DE JEUX VIDEO

Catégorisation des jeux vidéo

Il existe des milliers de jeux vidéos. Il est donc bien hors de question de tous les répertorier. En retraçant l'historique des jeux vidéo, quelques grands genres communément admis s'en dégagent : les jeux de puzzle/réflexion (Tétris, Angry Birds, Sudoku, Candy Crush Saga), les jeux d'aventures (Zelda, Pokemon mais aussi Diablo et Titan Quest également considéré dans la catégorie Hack'n slash), les jeux de gestion d'animaux virtuels (tamagotchi, nintendogs), urbains (Sim City), de stratégie (Advance Wars, les *Tower Defense*) et mêlant gestion et stratégie dans les jeux à Stratégie Temps-Réel (STR) comme Warcraft, Age of Empire, ou pas (Clash of Clans, Travian). D'autres jeux nécessitant plus de réflexes comme les jeux de plateforme (sonic, Mario) hack'n slash (Sacred), de rythme (Dance Dance Revolution), de tir (Counter Strike), de simulation (Flight Simulator, Need for Speed) et de vision à la première personne : les FPS (*First Person Shooting*) (Counter Strike, Crysis, Half-Life).

Dans le cadre de cet article, nous mettons en relation les genres de jeu vidéo par rapport aux compétences acquises tout au long de la formation. Nous identifions les genre de jeux basés sur des tableaux (figure 1) et les genres de jeux à base de moteurs physiques (figure 2). Nous hiérarchisons les compétences

acquises. Pour chacun des genres, des exemples de jeux emblématiques sont cités et illustrés.

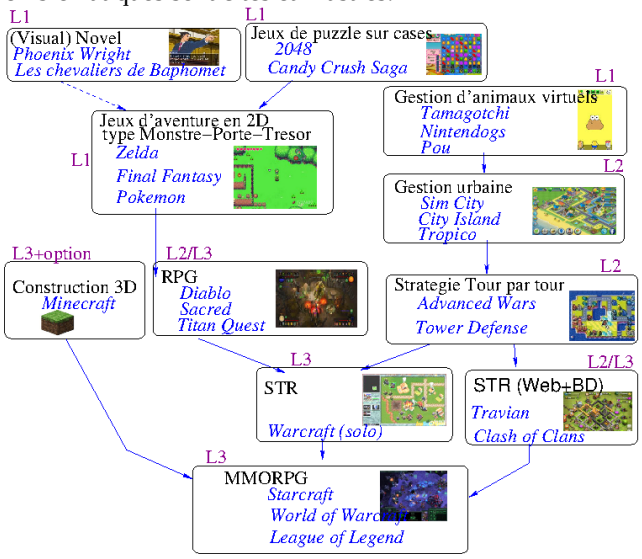


Fig 1 Jeux à base de tableaux

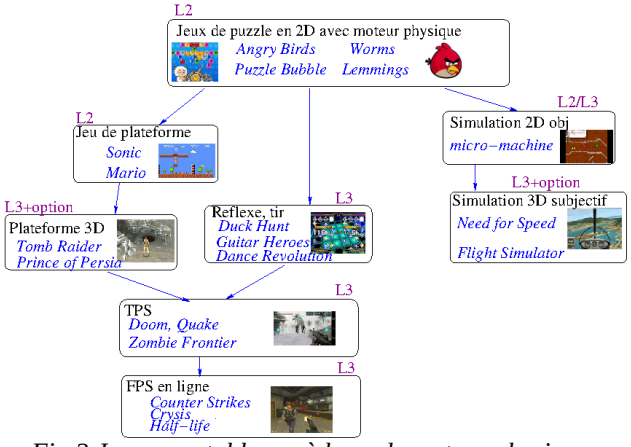


Fig 2 Jeux sans tableaux à base de moteur physique

Cinq ans de formation par projets

Comme dans toute formation LMD, les 5 années menant de la licence au master professionnel de l'université de Cergy-Pontoise se compose de 3 années de licence (donc 6 semestres que nous nommerons **S1, S2, ...S6**) et 2 années de master (**S7 à S10**) dans la suite de cet article.

Dans notre formation, nous attachons une importance particulière aux projets, que ce soit des projets de fin de module ou des projets de synthèse.

Un *projet de fin de module* est un projet dans lequel chaque équipe d'étudiant (de 2 à 6 suivant le niveau ou le module) travaille à la conception et l'implémentation d'un logiciel, qui peut être un jeu vidéo, et faisant intervenir les compétences travaillées dans le module. Le sujet est en général distribué en début ou au milieu du module et un certain nombre de créneaux libres encadrés, semi-encadrés ou non-encadrés sont réservés pour la réalisation de ce projet. La validation se fait par un rapport, une soutenance orale et une démonstration du logiciel.

Un *projet de synthèse* se fait au minimum sur un semestre (une année en L3 et jusqu'à deux années en master) en dehors de modules spécifiques. Des créneaux sont également réservés dans ce but. Ces projets de synthèse font appel à des compétences transversales entre tous les modules dispensés et laissent une grande ouverture pour l'acquisition de compétences additionnelles sur lesquelles l'étudiant se formera par lui-même suivant les besoins qu'il aura lui-même identifiés

Par la suite, nous décrivons les genres de jeu vidéo adaptés au module enseigné. La partie graphique est en général peu vue en cours et est laissé à la discrétion de l'étudiant motivé (et il y en a beaucoup lorsque l'on parle du jeu vidéo !).

Projets de jeux vidéo de module

1. Structure de données, programmation structurée (S1). En première année de licence, les étudiants s'initient à la programmation en C. Ils découvrent les structures de boucles (for, while) et de contrôles (if else, switch case). Les entrées/sorties sont simples : lecture de l'entrée clavier (scanf) et affichage en mode texte à l'écran (printf). Bien que simple, on peut citer le jeux :

▲ Histoires interactives, et « visual » novels. Basés sur le principe des livres dont vous êtes le héros et les devinettes, on citera le jeu Phoenix Wright .

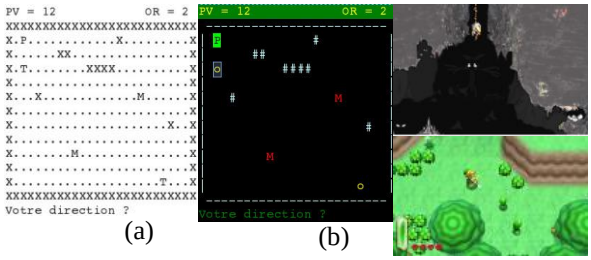
▲ Jeux d'élevage d'animaux virtuels. Il s'agit de s'occuper d'un animal possédant certaines caractéristiques quantifiées (niveau de satiété, de propreté, de sommeil, etc.). Le joueur doit s'occuper de son animal en réalisant des actions influant sur ces caractéristiques (par exemple en donnant de la nourriture pour augmenter le niveau de satiété). On citera dans ce genre, le jeu-modèle Tamagotchi, qui plus récemment ont inspiré les nintendogs (NDS), Pou (android) ainsi que tous les nombreux jeux d'élevage sur le web (Kochonland, Terato)

A ce niveau, tous les jeux basés sur tableaux à deux dimensions à interaction simple (le joueur effectue une action sur une case, le programme réagit, ceci en tour par tour) sont possibles. (Figure 2)

▲ Jeux d'aventure. Dans ce contexte peut être proposé un jeu se déroulant dans un univers (modélisé sous forme de tableau à deux dimensions) vu de dessus où l'on promène un personnage à l'aide des touches du clavier. Dans ce genre se rencontrent Rogue (en mode texte), Pac-man, Zelda, Final Fantasy (version 2D).

▲ Jeux de puzzle/réflexion à base de cases. Les classiques Morpion, Master Mind, et démineurs sont bien sûr représentés dans cette catégorie et accessibles sitôt les tableaux à deux dimensions introduits. On citera les jeux plus récents comme le Sudoku et 2048 qui sont orientés texte.

2. Algorithmique des graphes, automates et programmation orientée objet (S2, S3, S4). L'algorithmique en S1 et S2 permet aux étudiants de travailler sur les structures de données (listes chaînées, arbres, graphes) aux algorithmes de recherche et de parcours et aux automates. La programmation orientée objet avec Java enseignée en S3 donnent aux étudiants les moyens



(a) (b) c)

Fig 1: Jeux d'aventure réalisés en L1. (a) En mode texte, (b) en mode texte avec utilisation de code couleur, (c) en mode graphique avec la bibliothèque SDL.

de travailler sur des projets complexes en structurant hiérarchiquement les objets qu'ils manipulent et en utilisant des bibliothèques d'objets existants de façon transparente.

En particulier, les bibliothèques d'IHM et de manipulation graphique 2D. Par les projets proposés une fois le langage Java connu, une multitude de jeux vidéo possibles s'offre alors à l'étudiant. Les compétences visées sont de nature algorithmique (par exemple un plus court chemin par Dijkstra ou A* dans un graphe modélisant la carte dans lequel se déplace un personnage), d'automate (dans la gestion d'élevage, de certains puzzle) ou simplement de conception objet (définir efficacement et logiquement l'ensemble des objets représentant une ville).

En apprenant à construire des IHM simples avec les bibliothèques awt et swing lors du module de formation à la programmation orienté objet avec Java, les étudiants sont à même de construire des fenêtres graphiques avec boutons, barre de textes, menus, de gérer les clics de souris, de gérer les événement claviers sans bufferisation, etc.

▲ Jeux de puzzle/réflexion graphiques à base de cases. On citera dans cette catégorie l'ancêtre du genre : Tétris, mais de façon plus actuelle : Candy Crush Saga, Pet Rescue Saga ou encore Lumines.

Toujours dans ce module de programmation orienté objet en s'initiant au graphique 2D avec Java2D, les étudiants peuvent à présent afficher des images où tracer des figures où bon leur semble dans un conteneur graphique en s'affranchissant du tableau de cases.

▲ Jeux de puzzle/réflexion « enrichi »à base de physique. Les jeux à base de moteur physique (balistique, réfraction) comme lemmings, Worms, Angry Birds, Bad Piggies, Portal, Puzzle Bubble, Metroid Prime.

▲ Jeu de gestion urbaine. On citera dans cette catégorie Sim City, Tropico, City Island, Farmville

▲ Jeux de stratégie en temps-réel (STR). Reprenant ce principe, en permettant au joueur de contrôler plusieurs unités sans attendre son tour, en développant une petite intelligence artificielle basique à base de règles simples permettant de jouer contre l'ordinateur et en développant les algorithmes de recherche de chemin, l'étudiant peut à présent implémenter des jeux de type : Warcraft (solo), Starcraft (solo), et les Tower Defense.

3. Développement Web et Base de données (S4, S5)

Dans ce module, les étudiants peuvent stocker et récupérer des données dans une base de données par réseau. Outre l'aspect navigateur web avec lequel les étudiants sont familiers, ils peuvent à présent interagir avec les autres joueurs par l'intermédiaire de cette base centralisée. On retrouve les jeux d'élevage sur le web comme Kochonland, Terato, Nainwak, de gestion urbaine (City Island, Farmville) et les jeux de stratégie basés sur le web (Travian, Clash of Clans). Tout en programmant ces jeux, l'étudiant est amené à réfléchir à la conception de la base de données et de son dimensionnement.

4. Programmation réseau (S5)

Ce module ouvre la possibilité de réaliser la plupart des jeux en réseau. Outre les jeux d'échange (pokédex des Pokémon) et de combat en réseau (*Player vs Player*)

5. Programmation multi-processus et multi-thread (S6)

Avec les threads, la gestion multi-cliente est possible sur le serveur. Toutes les extensions multi-joueurs des jeux STR, des RPG avec équipes et/ou confrontation sont possibles. On cite alors :

▲ Les jeux en ligne (massivement) multi-joueurs de type RPG : (M)MORPG. C'est la catégorie de jeu la plus populaire parmi les étudiants, on y retrouve League of Legends, Starcraft (multijoueurs), World of Warcraft

6. Options et apprentissage complémentaire

Suivant les options, les étudiants peuvent enrichir leur connaissance afin de réaliser d'autres genres de jeux.

▲ First Person Shooting (FPS). En apprenant la programmation graphique en 3D, les jeux à tir subjectif initiés par Wolfenstein 3D (1992) FPS sont possibles. On parlera alors des jeux de type Doom, Counter Strikes, Zombie Frontier, Half-life, Crysis. L'apprentissage de la programmation sous Android permet enfin de porter tous les jeux sur smartphone, équipement de plus en plus utilisé par les étudiants pour leur jeu.

7. Les projets en master

Le master suivant la licence informatique à l'université de Cergy-Pontoise est le master ISIM SIC spécialisé en conception de Systèmes Intelligents et Communicants. Poursuivant la logique précédente, des projets consistant à programmer des jeux vidéo sont encore proposés (parmi d'autres projets plus « sérieux »). En licence, les étudiants ont pu de façon autonome développer leurs propres jeux en s'inspirant de modèle de jeux existants. On pousse à présent le modèle encore plus loin en proposant en master des jeux à IHM innovante et faisant interagir, certains des modules vus en master dont :

-Intelligence artificielle, apprentissage

-Informatique embarquée et programmation temps-réel sur carte embarquée

-Traitement d'image et de video, reconnaissance de forme, de visage, de geste, de lieu...

Pour des raisons de place, nous ne détaillons pas les projets de synthèse de jeux en Master.

4 ENCADREMENT EN INGENIERIE LOGICIEL ET EN GESTION DE PROJET

Les projets de module et de synthèse permettent aux étudiants non seulement de pratiquer les connaissances acquises dans un contexte de projet, mais aussi d'acquérir les compétences et de maîtriser les outils de développement des jeux vidéo. De plus, tout au long de la formation de 5 ans, des notions et de compétences de l'ingénierie logiciel et de la gestion de projet sont aussi abordées et mises en pratique durant ces projets. Cette section illustre concrètement la pédagogie mise en place pour attendre ces objectifs.

L'élaboration du programme pour le projet jeu vidéo est une procédure composée de 5 étapes entièrement encadrées par l'enseignant :

1) Une fois le sujet du projet de jeu vidéo connu, les étudiants forment d'abord des groupes de 3-4 personnes constituant des équipes de projet. Ensuite, pour chaque équipe, tous les membres se réunissent afin de discuter du contexte et de la portée du projet. Ceci conduira à l'élaboration d'un document important : le cahier des charges décrivant concrètement ce que le produit final doit fournir comme fonctionnalités ainsi que la façon dont l'utilisateur doit pouvoir interagir avec le système. Puisque le projet est souvent réalisé en utilisant un langage de programmation objet, la communication et la mise en forme des fonctionnalités sont assurées par les diagrammes de cas d'utilisation UML.

2) Une fois le cahier des charges validé, les membres de l'équipe commencent à collecter (sélectionner) les informations (données) pertinentes aux fonctionnalités définies. Non seulement la liste de toutes les données nécessaires doit être établie, l'organisation (ou la structuration) de ces données sera aussi analysée afin de trouver une solution flexible et extensible. L'utilisation de diagrammes de classes UML est primordiale lors de cette étape afin d'avoir une illustration claire de l'organisation des données.

Les diagrammes de classes seront colorés de différentes couleurs pour déterminer la répartition de tâches d'implémentation. Chaque membre de l'équipe choisira les classes d'une couleur à programmer. Cette approche permet aux étudiants de travailler simultanément et de pouvoir progresser dans le travail d'équipe puisque une intégration de code sera nécessaire.

En fonction du niveau des étudiants, la gestion de ces données pour le jeu est assurée soit par une base de données relationnelle (niveau Licence 3 et plus), par des sources de données utilisant de nouvelles technologies comme Big Data, XML, etc. (niveau Master 1 et plus), ou tout simplement des fichiers séquentiels ou sérialisés (niveau Licence 2 et moins).

3) Après la phase de construction du modèle de données, les membres de l'équipe vont travailler sur le moteur du jeu. Il s'agit des algorithmes qui manipulent des données définies dans l'étape précédente. Les étudiants décrivent ces algorithmes avec les diagrammes de séquence UML. Dans des jeux plus complexes, on doit aussi définir l'architecture du système, par exemple, Modèle-Vue-Contrôleur (MVC). Chaque membre de l'équipe s'occupera de certains algorithmes ou d'une partie du système. Ceci permet un travail simultané de chaque membre. De plus, comme ce qui est illustré dans la figure 3 de la section 3, la partie vue (IHM) est minimisée dans ces projets universitaires mais reste ouverte et extensible suivant la motivation des étudiants.

Pour garantir le fonctionnement du code développé, des tests unitaires et des tests d'intégration sont réalisés pendant le développement du jeu. A tout moment du développement, chaque membre de l'équipe doit s'assurer que les fonctionnalités du jeu déjà développées sont opérationnelles, sans régressions (l'ajout de nouvelles fonctionnalités ne remet pas en cause des fonctionnalités déjà existantes.)

4) A la fin du développement, des documents seront réalisés par les étudiants : manuels utilisateur, développeur, d'installation, de maintenance, etc. Les étudiants ne se contenteront pas uniquement du fait que leur jeu "fonctionne", mais aussi d'avoir réalisé un vrai "produit" du niveau proche de l'industrie informatique. Pour cela, à part des tests déjà effectués pendant le développement, ils réaliseront aussi des tests des cas d'erreurs d'utilisateur, d'exception, ainsi que la stabilité de performance quand la charge du système augmente.

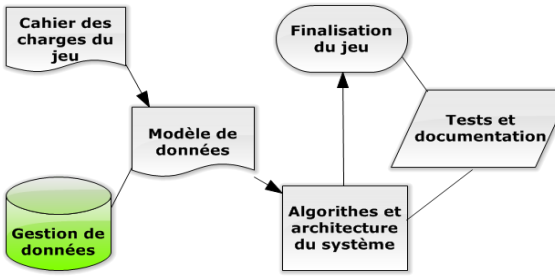


Fig 2: Elaboration du projet de jeu vidéo

Pendant le projet, différents outils de l'ingénierie logicielle sont pratiqués par les étudiants, par exemple, SVN pour la synchronisation de travail en équipe, les IDE (*Integrated Development Environment*) pour mieux manipuler le code source. Les frameworks comme JUnit pour gérer les tests, Log4j pour gérer le système de journaux, etc. Chaque année, les étudiants apprennent à utiliser ou à approfondir de nouveaux outils pour s'approcher des exigences des vrais projets industriels.

5 BILAN PEDAGOGIQUE

Les projets de conception et développement d'un jeu vidéo très joué par les étudiants sont toujours extrêmement bien accueillis par ceux-ci. Cela se constate 1) dès l'annonce du sujet où les étudiants sont enthousiastes, 2) durant la période projet où des multiples questions sont posées et des suggestions proposées spontanément, 3) lors de la soutenance où l'on découvre souvent que le jeu a été enrichi par de multiples fonctionnalités non demandées et ayant pourtant nécessité énormément d'apprentissage personnel. Il est même

arrivé qu'un étudiant aille pour un projet de Licence, se documenter dans des articles de recherche afin de puiser des idées pour améliorer l'IA du moteur de son jeu.

De plus, comme expliqué précédemment, cette approche d'enseignement par projet de jeux vidéo s'avère efficace car la seule activité de réalisation d'un tel projet permet d'approfondir de façon spontanée leurs techniques informatiques acquises, gérer et réaliser un vrai projet industriel, et acquérir des compétences/outils en ingénierie logicielle.

6 CONCLUSION

Nous nous sommes basés sur la programmation d'une suite de jeux vidéo afin de motiver les étudiants. Nous avons catégorisé les jeux existants suivant leur genre et les avons classés en fonction des compétences visées tout au long de la formation informatique à l'université de Cergy-Pontoise. Nous avons montré l'efficacité des résultats positifs de l'enseignement par projets de jeux vidéo.

La formation que nous souhaitons offrir à nos étudiants est une formation plus transversale et ouverte : enseigner les sciences informatiques de façon plus globale sans se limiter à la conception de jeux vidéos et sans dénaturer le programme de la formation visant à couvrir d'autres aspects qui ne sont généralement pas vus dans les jeux (par exemple la cryptographie, le datamining, les agents distribués). De longues discussions avec les étudiants ont permis de constamment se tenir au courant quand aux jeux vidéo qui les intéressent et ce qu'ils y recherchent.

Remerciements

Nous remercions nos collègues du département des sciences informatiques de l'université de Cergy-Pontoise qui ont travaillé avec nous dans l'élaboration, et l'encadrement de ces projets. En particulier Philippe Laroque, Pierre Andry, Marc Lemaire et Ghilès Mostafaoui. Nous remercions également l'ensemble des étudiants qui ont « subi » ces projets au cours des 14 ans durant lesquels nous avons essayé de leur donner ou d'entretenir la motivation de l'informatique par une approche ludique en conciliant passion de l'enseignement, des sciences informatiques et des jeux vidéo.

Bibliographie

[1] Syndicat National du Jeu Vidéo (SNJV) "Le jeu vidéo en France, éléments clés 2013", Livre blanc, 2013.

[2] Centre National du Cinema et de l'image Animé (CNC) "Les pratiques de consommation des Jeux Video des francais", étude réalisé par TNS SOFRES, septembre 2013

[3] J. Piaget, "De la pédagogie", Éditions Odile Jacob, 1988.

[4] D.W. Winnicott. "Playing and Reality" (London: Tavistock, 1971

[5] L. Vygotsky. The Role of Play in Development (pp. 92-104). In Mind in Society. Cambridge, MA: Harvard University Press, 1978

[6] Seymour A. Papert "Mindstorms: Children, Computers, And Powerful Ideas", ISBN-10: 0465046746

[7] Lemersie Tamara « Évolution d'un devis pédagogique favorisant le développement de la pensée procédurale en LOGO ». Résumé des actes du Colloque Logo et apprentissages, Fribourg, Suisse, 1990

[8] StarLogo : http://education.mit.edu/openstarlogo

[9] Scratch : http://scratch.mit.edu/

[10] A. Cockburn, A. Bryant, "Cleogo: collaborative and multi-metaphor programming for kids", Computer Human Interaction, 1998. Proceedings. 3rd Asia Pacific

[11] Alice : http://www.alice.org/index.php

[12] D. J. Cook, L. B. Holder, M. Huber, and R. Yerraballi, Enhancing Computer Science Education with a Wireless Intelligent Simulation Environment, Journal of Computing in Higher Education 16(1), 2004.

[13] Abt C), "Serious Games". New York: The Viking Press, 1970

[14] Marc Prensky "Don't bother me, Mom, I'm learning! How computer and video games are preparing your kids for 21st century success and how you can help", St. Paul: Paragon House, 2006

[15] http://www.ceebot.com/colobot/index-f.php

[16] http://armorgames.com/play/6061/light-bot-20

[17] Code Hero : http://primerlabs.com/codehero

[18] https://sites.google.com/a/eng.ucsd.edu/codespells/

[19] Codemancer : http://importantlittlegames.com/

[20] M. Muratet, P. Torguet, J-P Jessel, F. Viallet, "Towards a Serious Game to help Student Learn Computer Programming", International Journal of Computer Games Technology, 2009

[21] S.Leutnegger, J.Edgington "A Games first approach to teaching introductory programming", dans les actes du 38ème Technical Symposium on Computer Science Education (SIGCSE), pp 115-118, USA, 2007

[22] W-K. Chen, Y.C. Cheng "Teaching Object-oriented programming laboratory with computer game programming", IEEE Transactions on Education, vol 50, no 3, pp 197-203, 2007

[23] P. Gestwicki, F.S. Sun, "Teaching design Pattern through computer game development", ACM Journal on Educational Resources in Computing, vol. 8, no. 1, art. 2, pp 1-22, 2008