Ghoul Younes, Laville julien, Cherrak Nassim

Serious game

« Jeu serieu »

Table des matières

1 État de l'art

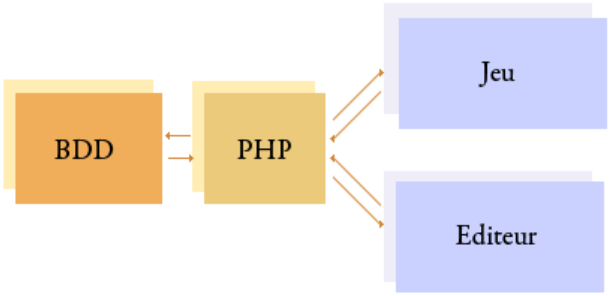
1 Définition

Les serious games sont des logiciels permettant d'apprendre, de s'entraîner ou de tester ses compétences et connaissances. Les serious games font partie de la vague des outils éducatifs, point de convergence entre les outils de e-learning et les jeux vidéos. Ils apportent à l'enseignement les mécanismes du jeu vidéo et couvrent une grande partie de l'ensemble des notions pouvant être apprises. Ils peuvent prendre l'apparence de tous les types de jeux vidéos (rts, économie, jeu de rôle, simulation). Ils permettent ainsi un apprentissage, une mise en situation ou véhiculent une idée.
Exemple concret de serious game : le cas du constructeur automobile Renault
Le constructeur automobile Renault propose des formations pour ses employés. Jusqu'à présent elles se composaient de séries d'une centaine de diapositives. Une étude a été faite pour mesurer l'efficacité de ce système et celle-ci a montré que la majorité des élèves perdaient leur attention avant la 10ème diapositive. C'est alors que la compagnie s'est orientée vers les serious games. Aujourd'hui une grande partie de leurs formations est faite de façon ludique.
Peut-on apprendre avec les serious games ?
Les technologies du milieu du jeu vidéo permettent des mises en situation: retranscrire un décor, simuler des interlocuteurs, etc. Lors d'un parcours de formation, il est intéressant de mettre les élèves dans les conditions particulières propres à la vie en entreprise, bien différente de la vie étudiante. Une mise en situation va, en effet, pouvoir apporter des leçons qu'un cours magistral ne peut offrir.
Ces jeux ont pour particularité de privilégier la qualité de l'apprentissage. C'est surtout l'immersion et la mise en situation qui sont recherchées, et par conséquent le serious game se révèle être un outil d'apprentissage et d'expérimentation particulièrement efficace.

2 Le projet

Les compétences nécessaires pour réaliser un serious game diffèrent de celles requises pour un jeu vidéo classique. En effet un serious game se situe au carrefour entre le savoir et le jeu vidéo. Il va donc falloir réunir les détenteurs du savoir avec les spécialistes du jeu vidéo.
Nous avons donc projeté de créer un serious game dans lequel des professionnels des métiers représentés (hôtellerie, vente etc.), des linguistes et des spécialistes des cultures étrangères puissent créer des exercices et/ou les corriger. L'exercice du serious game met le joueur dans une situation de conversation avec un seul interlocuteur dans laquelle il doit choisir le meilleur discours parmi ceux proposés. Cette contrainte est le point de départ d'une conception en trois parties :

- un outil de création des exercices
- une application permettant de faire l'exercice : le jeu lui même
- et enfin le média permettant l'exportation de l'exercice.



1 Organisation

Notre objectif principal était de créer un exercice et de pouvoir le charger dans le jeu, de réaliser l'exercice et d'obtenir un score final.

Le projet comporte donc 3 grandes parties : le jeu, l'éditeur et le suivi utilisateur.
Nous avons chacun assuré la conception et le développement d'une de ces trois parties. Les trois parties ont été réalisées individuellement, en parallèle avec des points de synchronisation.

La première chose à faire était de sélectionner les outils et technologies que nous utiliserions. Une fois une liste préliminaire établie, nous avons dû nous assurer que ces outils étaient capables de réaliser les taches définies. Les technologies web et scirra construct pouvaient poser un problème, car elles ne sont pas extensibles comme l'est java. Nous avons donc d'abord travaillé en commun pour trouver les solutions aux questions suivantes:

Concernant scirra construct:
- Comment parser du xml et comment afficher les informations extraites ?
- Comment se connecter à une application distante et échanger de l'information avec celle-ci?

Concernant le web:
- Comment faire un éditeur de graphe avec JavaScript et html ?
- Comment sérialiser un graphe en XML avec JavaScript ?

Une fois ces problèmes résolus nous sommes passés à l'étape suivante : standardiser le XML d'un exercice tel qu'il est écrit par l'éditeur et lu par le jeu. Pour cela nous avons construit ensemble un diagramme UML représentant la composition des objets stockés, puis, une fois d'accord sur les champs de chaque objet, nous avons créé un fichier XML minimum afin de définir ce qui serait attribut et nœud texte.

Nous nous sommes réunis pour effectuer les premiers tests, dans lesquels le jeu importe un fichier XML, puis propose l'exercice et enfin permet de le résoudre.

Cette partie clôture le premier prototype.

Il nous fallait ensuite réaliser le service permettant à un utilisateur d'importer depuis le web. Nous avons conçu ensemble le protocole permettant d'obtenir un exercice, puis nous avons à nouveau

développé en parallèle du côté du jeu et du serveur. Nous avons réalisé les tests d'accès à un document XML distant et clôturé le second prototype.

<table>
<tr><td>Laville Julien</td><td rowspan="3">Choix technologiques répondants à nos demandes</td><td rowspan="3">Tests des outils choisis</td><td rowspan="2">Accord sur le partage d'information entre le jeu et l'éditeur (XML)</td><td>Développement de l'éditeur</td><td rowspan="2">Test et lecture XML</td><td rowspan="2">Protocole d'importation de l'XML depuis une machine distante</td><td>Support PHP</td></tr>
<tr><td>Cherrak Nassim</td><td>Parsing XML</td><td>Connexion distante</td></tr>
<tr><td>Ghoul Younes</td><td colspan="5">Conception et suivi utilisateur</td></tr>
</table>

Organisation du travail

3 Le média

1 Choix de la technique

L'éditeur crée un exercice qu'il exporte vers le jeu. Il faut donc sérialiser l'exercice. Comment peut-on sérialiser une collection d'objets commune à deux langages ? Nous nous sommes naturellement tournés vers un troisième langage : le XML. En effet la plupart des langages embarquent la possibilité de lire ou d'écrire du XML. Le XML s'est donc imposé pour plusieurs raisons: d'abord parce qu'il offre un bon support de sérialisation d'objets ou de collections d'objets, ensuite parce qu'il permet de retranscrire un parcours non linéaire aisément.

2 Création de la norme

Le média étant commun au jeu et à l'éditeur, la première chose à faire était de s'accorder sur sa représentation.

4 Le jeu

1 Contraintes et objectifs

Le jeu vise les étudiants en commerce international souhaitant suivre une formation. Cette formation sera conçue par un professeur ou un professionnel du domaine partageant ses créations.
Le joueur se verra donc mis en situation de dialogue pour par exemple une transaction. Le joueur sera l'un des interlocuteurs et devra choisir la proposition la mieux adaptée à la circonstance parmi celles qui lui sont proposées. Nous souhaitons ajouter un support de cours qui serait associé à certaines réponses jugées trop fausses. Ces cours seraient soit préparés par l'enseignant et intégrés au jeu, soit inclus sous forme de lien hypertexte redirigeant vers la page web de l'enseignant.

1 Maniabilité

Ce jeu n'étant pas adressé à un public de joueurs, les contrôles sont simples: les touches directionnelles font déplacer le personnage et une fois à proximité d'un "objet" avec lequel on peut interagir une petite bulle s'ouvre au-dessus de celui-ci indiquant la possibilité d'interagir. Afin de rester dans des réactions

intuitives, le clic permet aussi de faire défiler les dialogues et choisir une réponse.

2 Compatibilité et performance

Le public des joueurs est principalement doté d'une plateforme windows. Cependant si l'os est commun aux joueurs, les performances de leur machine représente une très grande disparité. Blizzard l'a bien compris et leur jeu vidéo phare est en mesure de fonctionner sur des machines très peu puissantes.
Une série de tests effectués sur scirra nous a assuré que nous pouvions fournir un jeu jouable sur une machine destinée à la bureautique.

3 Extensibilité

La construction du jeu étant en bloc (appelés "Layout"), les différents menus (lancement du jeu, connexion, jeu...) peuvent être eux-mêmes encadrés par d'autres fonctionnalités, telles que par exemple l'ajout d'un menu permettant de concevoir à l'avance l'enchainement des exercices a faire.

2 Conception et spécification de l'interface utilisateur:

Pour qu'un enseignement par la simulation ou mise en situation soit profitable, il est nécessaire que le joueur ait une estimation de la qualité de ses réponses.
Les cours eux, sont complémentaires du système de points et peuvent confirmer qu'une réponse est bonne et pourquoi, ou à l'inverse corriger l'étudiant en détaillant son erreur.
Le dialogue:
Le dialogue est obtenu par "parsage" du fichier XML produit par l'éditeur (celui qui a produit l'exercice). Le joueur devra donc se concentrer sur cette partie pour pouvoir en déduire la réponse la plus plausible.
Une fenêtre de dialogue apparaît au bas de l'écran lors du lancement de la simulation: il s'agit de la parole de l'interlocuteur. Cela représentera donc ses réactions, ainsi que les questions qu'il posera.

1 Les réponses

Une fois la situation présentée, l'utilisateur se doit de réagir. Aussi, un maximum de quatre réponses lui sont proposées. Il pourra choisir en cliquant sur la réponse qu'il estimera la plus appropriée.
Les réponses affichées sont encadrées au survol de la souris pour inciter au clic. Cependant, les choix proposés sont les idées que le joueur peut véhiculer (interrompre, calmer, etc.), Le jeu affichera le texte correspondant, mais de type texte de dialogue. Ce choix va dans le sens de l'apprentissage dans la mesure ou le joueur peut avoir une idée mais ne sais pas l'exprimer; il peut ainsi, en choisissant une idée, voir une bonne façon de s'exprimer.

2 Les transitions

Après avoir réagit à une situation, le joueur peut enchainer sur une autre situation ou un cours, selon la conception de l'exercice par son créateur. Il sera plus judicieux de placer un cours à la suite d'une mauvaise réaction du joueur.

3 Les points

Le système de points est affecté par les réponses du joueur. Les réactions de l'interlocuteur, ainsi que les points acquis au cours d'un exercice, permettront de juger du niveau et d'améliorer ses capacités d'adaptation de situation en répétant les exercices tout en essayant d'améliorer ses capacités.
Le système de point est une des ficelles les plus addictive du monde du jeu vidéo. Il est un élément de comparaison vis à vis des autres joueurs et il incite à s'améliorer.

4 Longueur des textes

Les textes contenant les dialogues entre les personnages dans le jeu peuvent être longs. Pour parer a cela, le texte est découpé en plusieurs blocs que le joueur pourra faire défiler.

3 Choix Techniques et développement

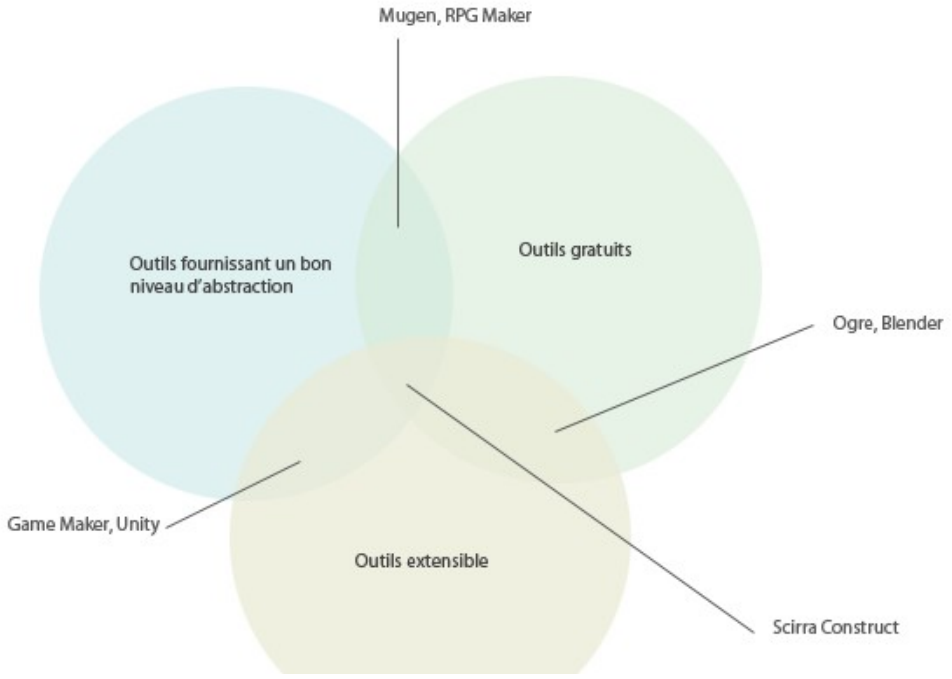


Diagramme de Venn de la distribution des créateur de jeux videos actuel

Nous avions besoin d'un outil rapide à prendre a main et à utiliser, qui soit gratuit et extensible.
La rapidité de développer un prototype était un facteur décisif et donc demandait un bon niveau d'abstraction. Cependant un logiciel de trop haut niveau risquait de ne pas être extensible. Il nous fallait aussi parser du XML ce qui n'est pas une utilisation commune des outils de création de jeux.
Scirra proposait le compromis idéal avec un très haut niveau d'abstraction: la création d'un jeu peut alors être faite en quelques clics et être extensible par son support du langage python.
Scirra s'avère utile de par sa facilité à créer un environnement avec une multitude d'objets qui interagissent entre eux. Ce qui nous enlève la tache de production de l'environnement et nous permet ainsi de nous concentrer sur l'acquisition et l'utilisation des informations produites par l'éditeur.
Nous avions besoin de traiter de l'information à partir d'un fichier. Le format XML s'avère le plus approprié dans notre cas grâce a la facilité de transport d'information de type textuel.
Le langage python étant intégré dans Scirra à travers l'utilisation de scripts, il nous a suffit d'utiliser la librairie du langage englobant le parsage de fichier XML. (Une mise à niveau pour Python fut nécessaire)

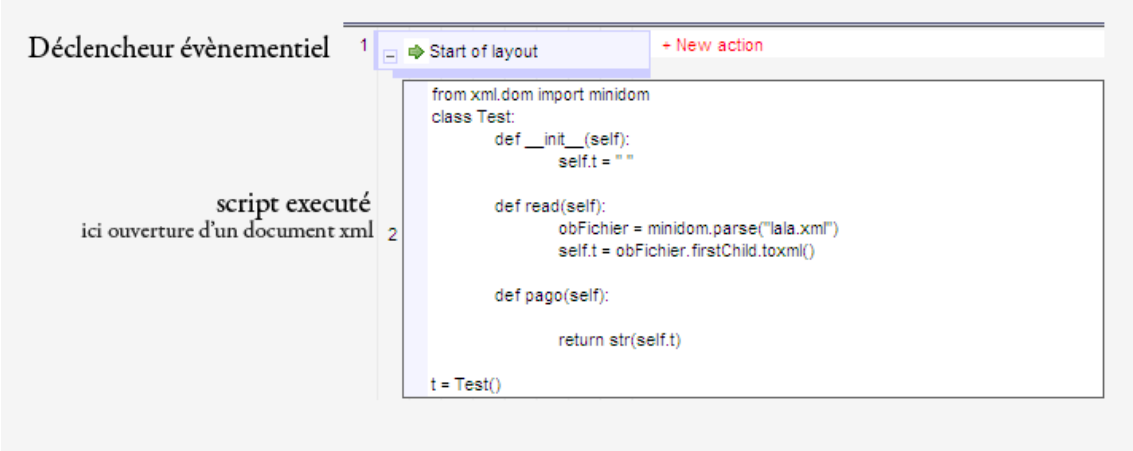
1 Importation

Lors de la connexion, les informations du joueur, telles que son niveau de compétence, sont obtenues de la base de données. L'obtention de nouveaux exercices se fait sous forme de téléchargement. Une fois cette opération faite, les exercices sont présents sur l'ordinateur du joueur sans interaction avec la base de données avant la fin de la simulation.

2 Structure et utilisation du fichier XML

Les fichiers se composent de deux types de balises: Cours (<cours>) et Questions (<questions>).
Chacune est identifiée par son identifiant en attribut.
Seules les questions possèdent des balises de type réponses (<reponse>).
Ci-dessous la composition de la balise réponse:
titre : représente la réponse telle qu'elle apparaitra parmi les choix de réponses possibles présentés au joueur.
texte : c'est le corps de la réponse prononcée par l'avatar du joueur.
futureid : c'est l'id de la future question ou du cours qui suivent si cette réponse est sélectionnée.
score : ce sont les points acquis en sélectionnant cette réponse.

Scirra est composé de conditions qui déclenchent des actions. Les actions peuvent être associées à des événements Scirra ou a des scripts python. Lors de la sélection d'une réponse par un joueur, un script se déclenche et recherche dans le fichier XML la future question ou cours qui y est associée. Ainsi, selon l'action du joueur, le script python de parsage XML nécessaire est déclenché.



Déclencheur et script exécuté

5 L'éditeur

Afin de créer un contenu éducatif pour les élèves, nous avons fait le choix de permettre la création d'exercices par des spécialistes. Dans cette optique, nous avons conçu et réalisé un logiciel qui les assisterait dans cette tache.
Un exercice est une suite de questions à choix multiples où chaque réponse modifiée a une influence sur le déroulement du questionnaire, ainsi que sur l'évaluation du joueur.

La création d'un exercice se découpe en trois phases :
- la création du graphe de questions réponses, c'est-à-dire l'ordre dans lequel apparaissent les questions, ainsi que le déroulement des questions en fonction des réponses choisies.
- la seconde phase est la rédaction des textes des réponses et des questions telles qu'elles seront soumises aux candidats.
- la dernière phase est facultative: elle permet une simulation du dialogue tout en proposant à l'utilisateur l'édition des textes ou la répondération des notes pour chaque réponse.

1 Contraintes et objectifs

Dans un premier temps nous allons analyser le public cible, puis définir la liste des fonctionnalités de l'interface. Pour cela nous répondrons à quelques questions:
- qu'est ce que l'utilisateur attend du système ?
- comment le système pourrait imiter la façon naturelle de réaliser ces taches ?

Notre utilitaire devra permettre la création d'un graphe où les sommets représenteront soit un cours soit une question et où les arêtes représenteront le lien entre une réponse et la réaction de l'interlocuteur. Il devra aussi permettre d'associer chaque réponse à un score. L'application proposera aussi la sauvegarde et la restauration d'un fichier de travail et enfin un outil permettant à l'utilisateur d'éprouver son exercice et de le modifier en même temps qu'il l'essaye.

1 Utilisabilité

Comment accompliriez-vous cette tache naturellement ? Cette question nous l'avons posée autour de nous.
Pour accomplir cette tache il faut un crayon et une feuille, sur laquelle la première question et les réponses possibles sont écrites. Il faut ensuite tracer une flèche allant de chaque réponse vers une autre question. C'est cette technique que nous devions imiter avec notre logiciel. L'éditeur est destiné à un public composé, entre autres, de linguistes, d'experts en cultures étrangères et de professionnels de la vente, c'est-à-dire un public non expert dans le domaine de l'informatique. Nous devions, de ce fait, fournir un outil facilement utilisable par le grand public. Il n'existe pas de recette pour faire une interface destinée au plus grand nombre, cependant le respect de quelques règles élémentaires et le développement par prototype nous ont permis de parvenir à un résultat satisfaisant.
Les règles principales à observer sont: simplicité, intuitivité et cohérence.

2 Compatibilité

Notre outil est destiné à être utilisé sur les ordinateurs du réseau universitaire, ainsi que les ordinateurs personnels, c'est-à-dire des plateformes unix, linux et Windows. Nous devons trouver un compromis pour que notre application soit accessible sur toutes ces plateformes.

3 Maintenabilité

Le développement par prototype contraint à effectuer un grand nombre de modifications du code au fil des différentes versions. Un grand soin doit être apporté afin de rendre le code maintenable, à travers le choix du langage et la qualité de la documentation. Enfin les limites de temps nous ont poussé à réaliser un projet embarquant un nombre limité de fonctionnalités. Le projet va, pourtant, être amené à grandir par la suite et nous devons donc anticiper les évolutions futures dès à présent.

4 Extensibilité

L'application devra être en mesure de proposer un nombre grandissant de services. Dans un premier temps, elle offrira un premier éditeur de dialogues, mais, par la suite, les évolutions des fonctionnalités du jeu demanderont une augmentation
du nombre d'outils proposés afin d'éditer les exercices du jeu.

2 Conception et spécifications de l'interface utilisateur

Existe-t-il des outils auxquels les utilisateurs sont déjà habitués et qui réalisent des taches similaires à l'application que nous concevons ?

1 Éditeur de graphes

Notre outil est en tout point un outil de création de graphes. Pour le concevoir nous nous sommes inspirés de ce qui existe dans le commerce en matière d'outils de création de graphes et, plus précisément, des outils de création UML.
La plupart des outils graphiques de création de diagrammes UML proposent une interface de glisser déposer, d'une part pour agencer les "boites" et d'autre part pour les relier. C'est cette fonctionnalité que nous avons reproduite dans notre application. Des boites représentant des questions ou des réponses, seront aisément positionnables ou pouvant être reliées pour créer le graphe. Cette particularité n'est cependant pas une technique d'interface classique, c'est-à-dire que la plupart des utilisateurs n'y sont pas habitués, bien que son usage soit naturel. Pour cela nous devons guider l'utilisateur dans son utilisation, grâce à une série d'indications. Par exemple, par une aide contextuelle lors du survol de ces boites, de leur sélection et lors d'interactions de liaison.

2 Édition des textes et notation

La seconde partie est l'édition des textes. Pour cela chaque question, réponse et cours correspondra à une série de champs de formulaires éditables.
Les champs de formulaires peuvent atteindre une taille trop importante, d'autant plus qu'une question propose quatre réponses et que chaque question et réponse se compose de deux champs de texte. Nous devons permettre à l'utilisateur de cacher ces zones de texte dès qu'il a terminé d'éditer les champs textes.
Pour le score, chaque réponse est associée à un nombre de points et nous devons permettre à l'éditeur de donner une note rapidement et de façon naturelle. Il existe, là aussi, une technique d'interface à laquelle l'utilisateur est habitué avec, entre autres, le système d'évaluation des sites de cinéma, winamp, pour la notation d'une musique et les blogs pour l'évaluation de la qualité d'un ticket.

3 Sauvegarde et restauration

Créer une simulation est une tache longue qu'il est indispensable de pouvoir sauvegarder et restaurer. Au niveau de l'interface, les utilisateurs sont habitués à trouver ces options dans le menu fichier des logiciels. Nous n'avons pas prévu de menu déroulant dans l'application, mais nous placerons cette option en haut à gauche.

4 Simulation

Le simulateur d'exercices doit pouvoir être exécuté à n'importe quel moment. Il doit permettre aussi l'édition des textes et des notes. Dans l'optique de ne pas rendre les formulaires d'édition invasifs, nous avons choisi d'utiliser des textes éditables sur place. De la même façon que le glisser déposer, cette technique d'interface n'est pas habituelle et nous avons donc dû assister l'utilisateur dans son emploi. D'abord au survol de la souris les textes éditables sur place devront modifier leur apparence pour indiquer une interaction possible, d'autre part, une aide contextuelle précisant que les textes sont éditables devra être fournie.

5 Aide contextuelle

Cette partie est une contrainte imprévue due aux choix de technique d'interface. En effet si notre éditeur ne se composait que de champs de formulaires, tout utilisateur pourrait créer son exercice sans être assisté. Cependant les glisser déposer et les éditions sur place nous obligent à créer un outil pour guider, aider et conseiller. C'est dans cette optique que doit être créée une zone de notification d'aide contextuelle. Cette aide pourra indiquer les actions possibles, indiquer par où commencer et conseiller. L'aide peut aussi se présenter sous la forme de "help nuggets" c'est-à-dire des bulles d'aide numérotées qui apparaissent au fur et a mesure des actions à faire. Ainsi, par exemple la première indique qu'il faut ajouter une question, la seconde qu'il faut éditer le texte de cette question et ainsi de suite. Les "help nuggets" ont la particularité d'apparaître là où l'utilisateur doit cliquer.

3 Choix techniques et développement

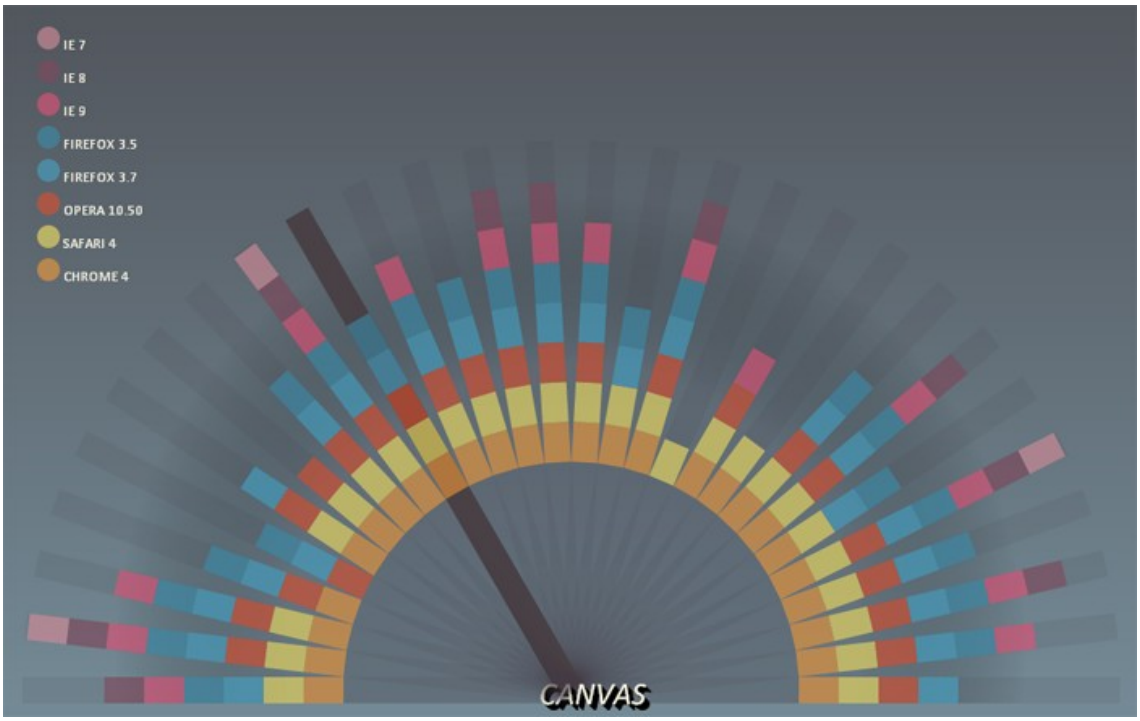
Il est nécessaire que notre application soit indépendante de la plateforme. Plusieurs possibilités s'offrent à nous: développer un logiciel en java, développer dans un langage dont les librairies sont communes aux principaux os et réaliser une compilation par plateforme ou développer une application web.

Le choix d'une application web a un avantage certain dans notre projet, contrairement aux autres options: il n'y aura pas d'installation à faire pour le client et les améliorations des outils ne nécessiteront pas de mise à jour à faire par le client.

Notre formation nous a donné les clés de deux technologies de développement web, la première étant php et la seconde l'utilisation de servlet java. Nous avons décidé de prolonger l'enseignement de php par le biais du passage à la programmation php objet.
L'éditeur sera une architecture trois tiers, tout comme le jeu. Un client se connecte à un serveur donnant l'accès aux méthodes relatives à l'édition, et enfin une base de données stockant les exercices. Du coté du client, nous avons opté pour une application en JavaScript afin de la rendre dynamique. Cependant, JavaScript n'est pas interprété de la même façon par tous les navigateurs. Nous avons inclus les frameworks prototype et scriptaculo.us. Ces deux frameworks ajoutent à la librairie standard une série de méthodes compatibles tout navigateur.

Le dernier pré-requis pour commencer le developpement et verrouiller le choix des technologies web est la possibilité de dessiner les liens entre les différents sommets. Html 4 ne permet pas cela mais

html5 embarque la balise canvas qui est l'équivalent d'un jpanel de dessin. Html5 n'est pas encore officiellement sorti, mais déjà supporter par les principaux navigateurs sauf internet explorer qui reste le seul navigateur qui n'est pas encore compatible avec la dotation canvas



compatibilité des navigateurs avec canvas

En résumé, notre application va être développée en php5 objet pour le coté serveur et en html5 JavaScript et les librairies prototype et scriptaculo.us pour le coté client.

1 Création du graphe

Créer un graphe revient à ajouter les sommets (les questions et les cours), puis relier les réponses à d'autres questions.
Comment créer des boites mobiles et comment allons-nous les relier ?
Le "framework" JavaScript scriptaculos.us apporte la méthode "draggable()" rendant une division dom déplaçable. La méthode "droppable()" transforme, quant à elle, une division dom en objet récepteur. Nous allons combiner les deux et ainsi créer une interaction entre les divisions mobiles.
Le tracé des arêtes est géré par le canvas de html 5. Cette dotation de la prochaine monture du langage html offre une grande quantité de méthodes de dessin par le biais de JavaScript. Nous superposerons ainsi un canvas à la zone des divisions mobiles. De cette façon nous pourrons superposer les liens avec les divisions représentant les questions et les cours.

2 Stockage des informations d'édition

C'est ici qu'intervient JavaScript objet. JavaScript propose le paradigme objet. Nous l'utilisons pour créer des classes de questions, cours et réponses qui contiendront les données à stocker. Il faut préciser que la structure particulière d'une question et d'un cours. (??? Phrase incomplète) En effet comme chaque texte sera disponible en plusieurs langues nous avons opté de les stocker dans des tables de hachage où la clé représente la langue.

3 Exportation

Cette fonction de l'éditeur consiste à retranscrire dans un document XML les collections de cours et de questions, ainsi que les liens. Les navigateurs proposent par l'intermédiaire de JavaScript un parseur XML. Il ne restait plus qu'à parcourir les collections de questions et créer l'arbre dom au fur et à mesure.
Pour l'exportation d'un exercice, il faut créer le document XML, puis construire les feuilles contenant les données propres à l'exercice (difficulté, décor, avatar de l'intermédiaire). Il faut ensuite parcourir les questions en créant pour chacune un noeud question, chaque noeud question contenant 4 noeuds réponses et enfin un noeud réponse pouvant contenir l'attribut "futur" qui est l'identifiant unique de la question future associée à cette question.

4 Importation et sauvegarde

A la différence d'une exportation pour le jeu, la sauvegarde devait conserver l'integralité des informations de travail. En effet l'exportation vers le jeu n'est faite qu'en une seule langue, là ou l'edition d'un exercice peut etre faite en plusieurs langues. Une autre donnée importante à conserver est la position des différentes divisions flottante dans la page. Il s'agit de remettre à la restauration les questions et les réponses là où l'utilisateur les avaient placés à sa dernière utilisation.
Le document XML devra donc conserver les positions de toutes les questions et cours.
L'acquisition des position de se fait par une méthode de prototype. L'information est stocké dans le Document XML, la restauration se fait par la méthode "moveTo()".

6 Développement de l'application serveur

1 Plugins

Le serveur se devait de fournir un support a une application dont le nombre de fonctionnalités ira en grandissant l'éditeur pour d'autres types d'exercices, support d'un service statistique...)
Comment permettre un support aisé d'un nombre grandissant de fonctionnalité ?
Nous avons choisis de gerer chacun de ces module sous forme de plugin.
C'est l'option qui a été retenue : Chaque module est placé dans un dossier plugin. Il sera automatiquement ajouté au menu principal, et aura accès aux fichiers javascript communs, feuille de style, et aux objets de la base de donnée.
Pour qu'un plugin soit disponible il doit respecter un certain nombre de règles, malheureusement il n'existe pas d'interface comme en Java ou en C++ obligent le développeur à implémenter d'une façon précise. Ces règles sont détaillés dans le guide du développeur.

2 La base de donnée

Les objets manipulés par php sont stockés dans une base de donnée relationnelle. Il nous a fallu créer une passerelle ente le coté objet et le coté relationnel. Pour cela nous nous sommes inspirés d'active record qui est un module très puissant du "framework" ruby on rails.
Un dossier nommé "modèle", contient les fichiers contenant les méthodes de création, de mise à jour de suppression et de recherche de chaque type d'objet contenue dans la base de donnée. ces fichiers sont chargés automatiquement et donne ainsi l'accès à ces méthodes par n'importe quelle page du site.

7 Suivi utilisateur

1 Contrainte et objectifs

Nous imaginons éventuellement une interface Web pour permettre à une équipe d'enseignants de suivre la progression des étudiants sur un serveur collectant les données de jeux, pour comparer et analyser les différents résultats qu'ils auront obtenus.
Cette interface se divise en 2 parties:

- coté client : le login, l'acquisition des exercices adaptés et envoie des statistiques
- coté serveur une gestion des utilisateurs: Gestion des sauvegardes, envoi des missions adaptés et traitement des statistiques.

Caractéristiques :
Transparence de la localisation : le processus serveur peut résider sur la même machine que le client ou, via un réseau, sur une machine différente. Le logiciel client/serveur masque aux clients la localisation du serveur en redirigeant les demandes de services si nécessaire. Un programme peut être client, serveur ou les deux.
Échange de messages : clients et serveurs sont des systèmes à couplage faible qui interagissent au moyen de messages. Le message est le moyen d'émission de demandes de services et de réponses à celles-ci.

Et Comme les applications client/serveur peuvent se différencier par la façon dont les fonctions réparties se partagent entre le client et le serveur. On a choisi le modèle *orienté serveur en* plaçant plus de fonctionnalités sur le serveur (serveurs web).
En plus, la logique de présentation est déplacée coté client et l'état du service (les données) reste côté serveur étant donné que toutes les interactions de l'utilisateur ne nécessitent pas l'interrogation du serveur ce qui permet à l'utilisateur de continuer ces activités durant le traitement d'une requête.

2 Conception et spécifications de l'interface utilisateur :

Le client envoie une requête au serveur sous la forme d'une URL avec éventuellement un passage de paramètres (login, scores…)
Le serveur commence par vérifier si la page demandé dépend du module php ou est statique
Il s'agit d'une page PHP, le script est alors interprété par le moteur qui peut éventuellement envoyer une requête vers un serveur de base de donnée MySQL (enregistrer/récupérer les données)
Celui-ci lui renverra alors les données demandées afin de terminer l'interprétation du script
Au final, il générera une page statique du même type qu'une page html pouvant être interprétée par le navigateur du client et qui affiche les données des étudiants
Cette architecture nous permets de garder un certains nombres de contrainte de point de vue :

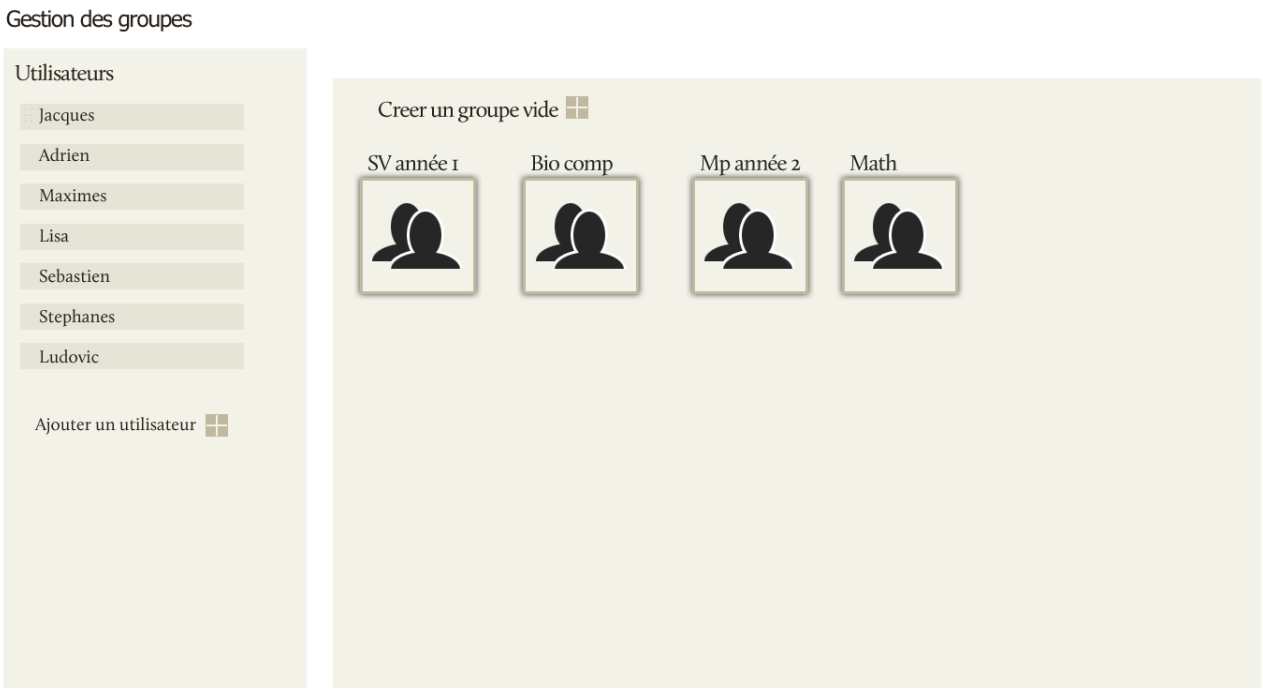
- Unicité de l'information : Certaines informations de notre application (Login, Scores) sont stockés dans une base de données sur le serveur. De cette manière, les informations restent identiques. Chaque utilisateur accède aux mêmes informations.
- **Meilleure sécurité :** Lors de la connexion un PC client ne voit que le serveur, et non les autres PC clients. De même, les serveurs sont en général très sécurisés contre les attaques de pirates.
- **Meilleure fiabilité :** En cas de panne, seul le serveur fait l'objet d'une réparation, et non le PC client.

- **Facilité d'évolution :** Une architecture client/serveur est évolutive car il est très facile de rajouter ou d'enlever des clients, et même des serveurs (possibilité d'élargir le jeu en multijoueurs)

Pour accomplir tous ces taches on a définit les différents besoins du système tout en analysant la nature des utilisateurs ciblés par notre application, pour cela on a diviser le problématique de la conception de :
-point de vue utilisateur: définir les taches à réaliser
-point de vue système: s'inspirer des exemples pratiques et courants en informatique en adaptant notre application à des vues courantes en informatique
Notre Serveur devra permettre la gestion des groupes et les utilisateurs de l'application au meme temps.

Gestion des groupes

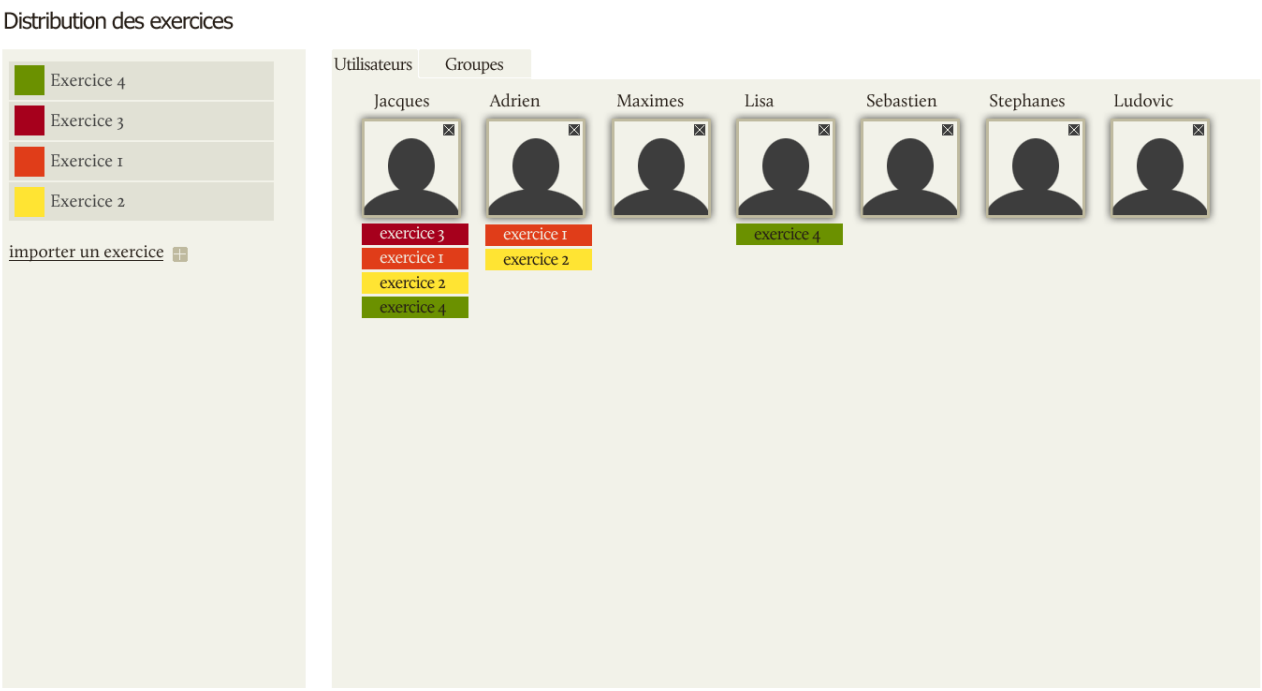
En effet , on a commencer par analyser et chercher la manière la plus simple et plus pratique pour gérer les groupes à partir d'une liste des utilisateurs.
A premier vue, on pourra utiliser le principe de glisser un utilisateur donné dans le groupe. Cette tache est inspirer de l'exemple de la "barre d'outils de Windows" en marquant les différents utilisateurs par des points pour montrer la disponibilité de cette fonction "glisser/déposer"
Ensuite pour l'amélioration de cette interface et faciliter encore plus la taches pour l'utilisateur on peut commencer par sélectionner un groupe et pour lui affecter les utilisateurs, il suffit juste de cliquer sur le "signe +" qui se trouve a coté de chaque utilisateurs et pareil pour le supprimer il faut juste cliquer sur "la croix".



Distribution des exercices

On a utiliser le même principe : à partir d'une liste de question, on peut affecter à chaque utilisateur une ou plusieurs questions tout en utilisant "glisser/déposer" et pour l'amélioration on peut cliquer sur un

utilisateur donné, pour ouvrir une fenêtre et lui affecter les différents questions. Pour la suppression il faudra juste cliquer sur la croix.



Cette organisation conceptuelle de l ’ensemble des fonctionnalités du système en un ensemble de commandes et de leurs effets est décomposée en deux partie : la fenêtre principale, le menu principal et les menus contextuels; et pour chaque partie on a spécifié les éléments suivants:

- les composants
- la disposition
- les interactions
- les couleurs

La conception repose donc sur une analyse préliminaire: analyse de tâche, observation des individus en situation pour faciliter l'apprentissage et l'utilisation en gardant une cohérence externe avec une commande externe (glisser/déposer).
En effet cette représentation graphique des objets permet a l'utilisateur :

- d'avoir un effet physiques sur ces objets au lieu de l ’utilisation d ’une syntaxe complexe
- donne un aspect : rapides, incrémentales et réversibles
- Évite de surcharger l'écran: affichage de l ’info utile pour l ’activité en cours

3 Choix technique et développement

L'implémentation de cette partie en PHP nous a permis surtout d’interagir avec une base de données par l’intermédiaire de fonctions tout en développant un serveur Web qui permet lui aussi d’enregistrer le login , le score… à partir de notre jeu (coté client) et afficher la liste des utilisateurs et les statistiques du jeu coté serveur.
L'**avantage** de PHP, outre sa flexibilité, est le fait qu'il possède les même fonctionnalités que les autres langages permettant d'écrire des scripts CGI (Common Gateway Interface) comme collecter des données, générer dynamiquement des pages web ou bien envoyer et recevoir des cookies … .
La plus grande qualité et le plus important avantage du langage PHP est le support d'un grand nombre de bases de données et la simplicité d’interfaçage avec eux.

De plus, le fait que PHP soit à la base un processeur hypertexte en fait l'outil idéal pour plusieurs raison :

- Excellent moteur d'expressions régulières, puisque l'une des principales fonctions du langage est de transformer ou parser du texte
- Génération presque instantanée de rapports formatés en HTML et servis sur le Web, puisque PHP est conçu pour générer des pages Web en HTML
- PHP est utilisable sur la majorité des systèmes d'exploitation
- PHP supporte aussi la plupart des serveurs web actuels : Apache, Microsoft Internet Information Serveur, Netscape et beaucoup d'autres encore.
- La gratuité et la disponibilité du code source
- La simplicité d'écriture de scripts

Les **inconvénients** d'utiliser PHP avec scirra sont peu nombreux.
Le seul défaut théorique de PHP pourrait être ses mauvaises performances inhérente à sa nature de langage non compilé et qu'il ne soit pas adéquat en terme de rapidité et de maintenance pour des projets de grandes envergures , mais le fait de ne l'utiliser que des appels http à partir de notre jeu rend l'envoi des données au serveur et le chargement plus efficace.
En effet, HTTP dispose d'un mécanisme permettant de tirer parti efficacement du système de cache et des autres capacités du navigateur client, pour économiser la bande passante, la puissance de calcul du serveur, et améliorer les temps de réponse.
La programmation de cette interface a donc été rapide, pour un résultat extrêmement satisfaisant.

Conclusion

La création d'un jeu vidéo est quelque chose de nouveau pour nous. Nous avons du appréhender de nouveaux outils, une méthode de travail propre au développement d'une telle application et nous avons aussi créé deux applications étroitement liées.
Il nous était impossible d'anticiper sur le temps de développement, ni sur les problèmes que nous allions rencontrer. Nous avons tout de même réussi à tenir le principal objectif qui était de faire fonctionner un jeu pouvant importer un exercice créé via l'éditeur.

L'échéance nous a limité dans le nombre de fonctionnalités à implémenter, mais nous avons, en plus de ce qui a été réalisé, envisagé une grande quantité d'améliorations et d'ajouts. Dans cette optique, nous avons développé de façon à ce que les futures améliorations soient aisément incluses dans le projet.
Ce projet nous a beaucoup plu dans la mesure où nous avons travaillé avec les outils pédagogiques vus pendant de nos études qui nous plaisent le plus. En outre, le projet est la base d'un projet qui va grandir et sera peut-être le futur outil pédagogique des étudiants de l'université de Nice Sophia Antipolis.

Bibliographie

[1] http://en.wikipedia.org/wiki/Serious_game
[2] http://www.scirra.com/

[3]Rapport d'enquète: « Peut-on apprendre en jouant » Master 1 Management des Inovations en
[4]Communication : http://www.jeuxserieux.fr/?p=869
[5]Le guide de survie – Python – L'essentiel du code et des commandes
[6]Ajax : Développer pour le Web 2.0 de Luc Van Lancker

Annexe

Plateforme php

ManagoEditor	Dossier principal
app	Dossier contenant les fonctions principale au fonctionnement du site
model	Dossier contenant
active_controller.php	Fichier octroyant des methodes d'assistance pour developper le site
active_record.php	Fichier de gestion de la base de donnée
engine.php	Fichier permetant l'imporation des methodes d'accés a la base de donné et gestion des plugins
config	Dossier contenant les fichiers de configuration
database.php	Fichier de configuration de la base de donnée
plugin	Dossier contenant les plugins

public	Dossier contenant les fichier communs a toutes les pages
js	Dossier ou sont stoqué les fichiers javascript
css	Dossier ou sont stoqué les feuilles de styles
images	Dossier ou sont stoqué les images
index.php	Accueil

Methode d'active controller

link_to(url, texte)	assisntant créant une balise lien
image_tag(url, alt)	assistant créant une balise image

Methode de l'active record

A importer nécessairement : enginesimple.php

MODELE::create(field 1, ...)	créer un objet en base de donée avec les champs passés en parametre
MODELE::find()	retourne une collection de tous les objets de la base de donnée.
MODELE::find_one(id)	retourne un objet ayant l'id passé en paramètre.
MODELE::update(id, field1,...)	met à jours l'objet ayant l'id passé en parametre avec les champs passés en paramètres.

Plugin

Toutes les pages doivent importer

engine.php

header.php

un fichier placé a la racine doit se nommer index.php

il doit contenir un png dommé "nom du plugin".png pour etre affiché dans le menu

un plugin a accès aux méthodes d'active record ainsi que les methodes link_to() et image_tag()

Le plugin editeur

variable reservé

questions, courses, liens, lgs, lg, current.

canvas.js	fichier d'ecoute et d'ecriture du canvas.
Course.js, Question.js, Reponse.js	classes des objets java script
simulation.js	fichier gerant la simuation
xml.js	fichier d'importation, de sauvegarde, et de restauration d'un exercice par serialisation xml
aide.js	fichier d'affichage de l'assistant