Rapport TER MICROMONDES 2009

CARIOU Loic
CROS Guillaume
LIMOUZIN-LAMOTHE Adrien
ROMAIN Sébastien

10 mai 2009

Table des matières

Première partie

Introduction

Le projet « micromondes » a pour but de permettre à des internautes de modéliser un espace réel (une pièce de leur appartement, un magasin, un bar, une piste de discothèque...), qui sera par la suite accessible aux autres internautes sous la forme d'une réprésentation en trois dimensions. Ces micromondes seront ainsi des lieux de rencontre et d'échange « customisés » au gré de leur créateur.

Ce projet qui se place dans le subtil contexte de la « réalité virtuelle » répond aux contraintes suivantes qui ont été définies par notre tuteur

- la topographie du micromonde, puis ensuite sa visualisation en 3D doivent être effectuées dans un navigateur via la technologie Flex (développement d'applications Flash grâce aux langages MXMLActionScript 3), cette technologie étant explicitée par la suite, et non par l'intermédiaire d'un client lourd (comme dans le cas de la célèbre mais un peu prétentieuse application « Second Life ») ;

- les différents micromondes seront centralisés sur un serveur unique qui gèrera les communications et les déplacements des avatars fréquentant ceux-ci.

Ce projet a débuté l'année dernière et a continué cette année en mobilisant plusieurs groupes de projets issus de la troisième année de Licence et du Master informatique. Si d'autres groupes ont eu pour consignes de permettre la modélisation d'un espace en y incluant des ouvertures (fenêtres, portes-fenêtres...), en texturant leurs faces internes par des photographies prises sur le lieu réel, de modéliser des avatars suivant les désirs des internautes ou encore de gérer leurs conversations, notre tâche est de gérer la création puis le placement de mobiliers simples (chaises, tables, éléments de cuisines..), au sein d'un micromonde.
Plus précisément, nous avons dû dans un premier temps identifier la framework 3D Flex le plus approprié parmi ceux qui avaient déjà été explorés par les groupes de projets précédents (Papervision et Collada), ainsi que de nouveaux (Sandy et Away3D), avant de permettre la modélisation d'un meuble « from scratch », ou à partir d'un catalogue de meubles existants ; de pouvoir le placer au sein d'un micromonde (réalisé par un groupe de deuxième année de la spécialité DIWEB du Master IFPRU, que ses membres soient loués) ; de le déplacer etou de le redimensionner.

Deuxième partie

Présentation du cadre du projet

Chapitre 1

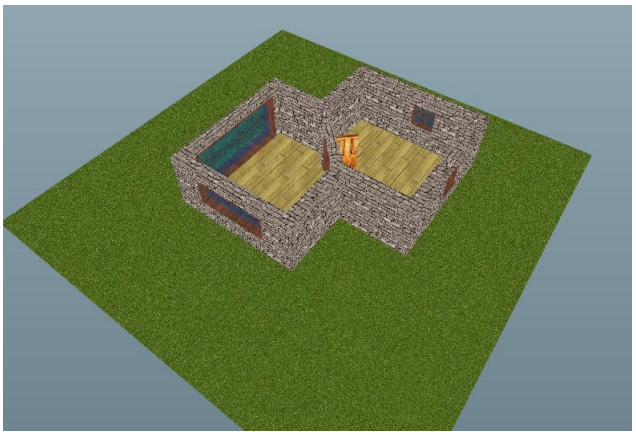
Présentation du projet

1.1 Le micromonde de départ

Comme dit précédément le projet microMondes est un métaprojet impliquant plusieurs groupes de TER. Ce projet, basé sur la technologie flash, permet de générer un monde en 3D dans un navigateur. Plusieurs versions de celui ci ont déjà été réalisées.
Quand notre projet a débuté, notre encadrant nous a fourni la nouvelle version codée en Alternativa3D. Voici les fonctionnalités qui étaient présentes quand nous avons vu cette version pour la première fois

- Un monde en 3 dimensions était déjà construit.
- Ce monde est décrit et créé à partir de fichiers XML.
- Un objet était créé, cet objet était un cube en 3D.
- Une caméra permettant de se promener autour du monde

Voici une capture d'écran de ce monde de départ



1.2 Gestion des données

Les données de ce monde sont décrites dans des fichiers xml. Une première étape de ce projet a été la compréhension de ces fichiers et de leur structure.
Il y a 2 types de fichiers Xml décrivant la création d'un monde
- Le monde monde.xml permettant de donner la structure du monde.
- Les objets Ex cube.xml qui permet de décrire la composition d'un cube en 3D.

Ces 2 types de fichiers ont été analysés. Ci-dessous une explication des principales balises et attributs.

1.2.1 Fichier décrivant le monde

<murs hauteur="10">
</murs>

Cette balise permet de définir les murs dans le monde. Tous les murs qui seront à l'intérieur possèderont donc dans cet exemple une taille de 10. Cette taille est une unité dans le microMonde et ne correspond pas aux pixels. À l'intérieur de cette balise, nous pouvons trouver 2 éléments différents des balises Mur et des balises Ouverture.

<mur couleur="0xffff00" texture="mur.jpg">
<point x="60" y="-80"/>
<point x="54" y="-80"/>
<point x="54" y="-82"/>
<point x="60" y="-82"/>
</mur>

Cette balise défini un mur. Elle doit être située dans une balise murs. Une texture est mise en attribut afin que le mur en 3 dimensions ait l'air plus réel. Les 4 points permettent de définir un rectangle au sol. Grâce à l'attribut hauteur de la balise mère, les 4 autres points permettant de le construire sont directement calculés et le mur est construit.

<ouverture hauteur="4" z="4" w="180" type="fenetre2" texture="mur.jpg">
<point x="54" y="-80"/>
<point x="47" y="-80"/>
<point x="47" y="-82"/>
<point x="54" y="-82"/>

</ouverture>

Cette balise défini une ouverture dans le mur. Un attribut hauteur est ici présent car cela sert à définir la hauteur de l'élément que l'on souhaite intégrer. Par exemple ici, nous souhaitons ajouter un objet fenetre2 de taille 4 et qui prend comme largeur toute la place prise par l'ouverture. L'attribut z permet de modifier la hauteur à laquelle l'ouverture commence. Enfin, l'attribut w décrit la rotation de l'objet afin qu'il soit dans le bon sens, ce chiffre représente l'angle en degrés.

<objets>
</objets>

En dehors de la balise mur, on peut apercevoir une balise objet. Celle ci contient tous les objets du monde. Cette balise ne possède pas d'attribut.

<objet hauteur="1" z="0" w="0" type="cube">
  <point x="35" y="-72"/>
  <point x="35" y="-73"/>
  <point x="34" y="-73"/>
  <point x="34" y="-72"/>
</objet>

Cette balise, contenue dans une balise objets, défini un objet dans le monde. On retrouve dans cet élément un attribut hauteur qui donne la hauteur de l'objet, un attribut w qui donne l'angle en degrés. Le troisième attribut est le type et donne le nom du fichier xml qui le défini. Enfin sont contenus dans la balise objet les points où l'objet sera positionné.

<sols>
</sols>

Cette balise permet de contenir les sols dans le monde.

<sol couleur="0xffff00" texture="herbe.jpg">
  <point x="0" y="0"/>
  <point x="-100" y="0"/>
  <point x="-100" y="-100"/>
  <point x="0" y="-100"/>
</sol>

Cette balise est contenue dans la balise sols et permet de définir un sol dans le monde. Un sol est un rectangle ou une texture est appliquée. Ici la texture est

passée en attribut avec le nom du fichier. Les points contenus dans la balise sol permettent de définir le rectangle ou se trouve le sol.

1.2.2 Fichier décrivant un objet

```
<meta>
  <name>cube</name>
  <description/>
  <dimensions h="1" l="1" p="1"/>
</meta>
```

Cet ensemble de balises permet de définir le nom de l'objet, ici « cube » ainsi que ses dimensions. Les dimensions doivent toujours être à 1 afin que les textures appliquées ne soient pas étirées.

```
<composition>
</composition>
```

Cette balise contient toutes les faces de l'objet 3D à afficher. Elle ne contient pas d'attributs.

```
<face couleur="ffffffff" texture="mur.jpg" alpha="">
  <point x="0" y="0" z="0"/>
  <point x="0" y="0" z="1"/>
  <point x="1" y="0" z="1"/>
  <point x="1" y="0" z="0"/>
</face>
```

Cette balise permet d'ajouter une face à un objet. Elle contient une texture qui est le lien vers le fichier image de la texture à appliquer sur la face. L'attribut alpha lui, permet de faire de la transparence sur la face en pourcentage. Les points qui sont définis dans la face permettent de tracer la forme à afficher.

1.3 Technologies utilisés

Serveur Web

MicroMonde est une application exécutée sur un serveur Web elle nécessite donc d'être hébergé. Le serveur local que nous avons employé est Apache HTTP Server au travers du pack WAMP/LAMP.

Apache HTTP Server, souvent appelé Apache, est un logiciel de serveur HTTP produit par l'Apache Software Foundation. C'est le serveur HTTP le plus populaire du Web. C'est un logiciel libre avec un type spécifique de licence nommée licence Apache.

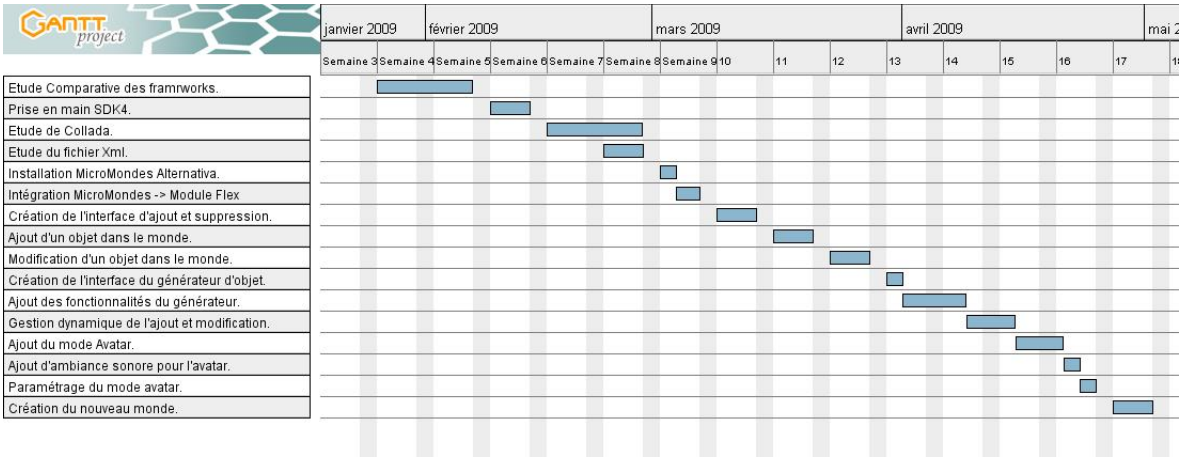
Langage de programmation
Coté serveur, la technologie utilisée dans les pages web est le PHP dans sa version 5.

PHP (acronyme pour Hypertext Preprocessor) est un langage de scripts libre principalement utilisé pour produire des pages web dynamiques via un serveur HTTP, mais également fonctionner comme n'importe quel langage interprété de façon locale en exécutant des programmes en ligne de commandes.

L'interface de MicroMondes est développée en Flex.

Flex est une solution de développement crée par Macromédia puis reprise par Adobe, permettant de créer et de déployer des applications Internet Riches 'RIA') multi plates-formes. Son modèle de programmation fait appel à 2 langages
- Mxml Des fichiers xml permettant de décrire la structure de l'interface graphique.
- ActionScript 3.0 Des fichiers ActionScript, langage à objet permettant de faire du code métier

1.4 Organisation du groupe



Voici le diagramme de gant correspondant à notre organisation pour le TER. Dans un premier temps, nous avons étudié les différentes technologies nécessaires pour le projet, à savoir le sdk4 d'Adobe Flash et les différents frameworks de flash pour faire de la 3D . Ensuite, nous avons étudié les différents fichiers décrivant des objets 3D (le format collada) et les fichiers xml du groupe de Master. Enfin nous avons intégré le microMonde existant dans un module flex et développé les différentes fonctionnalités d'ajout dans le monde et de modifications d'objets. Par la suite, nous avons développé un générateur d'objet afin de pouvoir enrichir notre monde de nouveaux objets. Pour mieux rendre compte de notre travail un nouveau monde plus complet a été créé (grâce aux outils développés) ainsi qu'un mode permettant de incarner un personnage dans le micromonde (Mode Avatar).

Chapitre 2

Etude comparative des framework

2.1 Introduction

Comme étude préliminaire à notre projet nous avons eu à comparer les principaux frameworks 3D disponibles pour flash. Afin de déterminer lequel serait le mieux pour le projet nous avons dégagé plusieurs critères de comparaison.

2.2 Support Flash 10

Mis à part Papervision en cours de reconstruction (PapervisionX retardant la version 2.0 prévue), les 3 autres frameworks supportent et tirent parti de la version 10 du flashplayer. (Sandy 3D depuis la 3.1 RC1, Away 3D depuis le 3.3 et enfin Alternativa depuis la 5.3)

2.3 Qualité de la communauté, fréquence des mise à jour

Les 4 frameworks disposent d'un blog cité en annexe, sur lequel figurent les mises à jour et les communications directes des développeurs. Ces blogs sont tous mis à jour environ une fois par semaine, de plus, chacun des frameworks est muni d'un forum ou d'un groupe google afin de résoudre les problèmes plus spécifiques et d'informer de manière plus ciblée et fréquente que sur le site officiel.

2.4 Gestion Ombre et Lumière

Tous les frameworks ici présents géraient les ombres et lumière de bases. Le plug-in Alternativa (celui choisi) gère en plus les lumières et les ombrages dynamiques, la diffusion d'ombres ainsi que le cycle journuit.

2.5 Qualité Graphique

Les différentes démos sur le web sont à peu près toutes du même niveau, avec une légère préférence pour Alternativa pour la qualité. A ce jour il n'est pas possible de savoir si l'un des framework est mieux puisque cette évaluation est faite sur les démos. Ainsi un meilleur démomaker ou une plus grande équipe pourra mieux mettre en valeur son plug-in.

2.6 Format d'objet 3D

Les 4 frameworks supportent le format Collada et 3DS ainsi que la description de scène via fichier .wrl.

2.7 Type de licence

Hormis Alternativa, les autres frameworks sont sous licence Open Source. Alternativa possède une licence hybride, gratuite pour les projets non commerciaux, et coûtant 1000euros dans le cas d'un projet à but lucratif.

Chapitre 3

Etude du format collada

3.1 Les origines de collada

Le format Collada (Collaborative Design Activity) est un format d'échange entre applications 3D (exemple 3Ds Max, Maya, Sketchup, ...). L'extension de ce format est .dae (Digital Asset Exhange) Le format Collada a été créé par Sony pour des besoins de modélisation 3D sur sa Playstation 3 et PsP. Depuis Collada est maintenu par Khronos Group qui continue à avoir la licence avec le créateur du format. Le but est de créer un format qui soit compatible avec un maximum d'outils de développement 3D et utilisé par les infographistes.
Aujourd'hui, Collada a été adopté par un grand nombre de sociétés de développement dans les jeux-vidéo et leurs moteurs.

3.2 Intéret du format

Collada (avec sa structure ouverte et évolutive) a pour but de devenir le standard des fichiers 3D, remplaçant ainsi petit à petit l'ensemble des formats propriétaires. Il est aujourd'hui en bonne voie, vu que de nombreux studios l'ont adopté et que de nombreux logiciels le supportent. Certains en ont même fait leur format natif.

De nombreux outils reconnus l'ont adoptés et ont donc contribué à sa popularité (3DS Max, Maya, Cinéma 4D notamment). D'autres logiciels utilisant la modélisation 3D en ont fait leur format de base, comme exemple, citons l'un des plus connus : Google Earth.

3.3 Collada trop complet pour notre projet ?

Pour notre projet ce format avait donc de prime abord l'air adapté, mais quand nous sommes passé à la compréhension du code d'un fichier Collada les choses se sont compliquées.
Pour commencer nous avons pris le .dae d'un simple cube. Il y avait un grand nombre de balises dont certaines était inutiles pour notre projet. Ne voulant pas perdre trop de temps, nous avons donc tenté une autre approche pour l'intégrer dans notre projet. Ainsi nous avons regardé si les différents logiciels de modélisation 3D disponibles nous permettaient de convertir des objets XML en .dae par simple ligne de commande. Après avoir cherché avec les logiciels Maya et sketchup nous avons décidé de trouver un autre format (plus simple) pour nos objets 3D dans le monde. Grâce au groupe de M2 aussi sur le projet nous avons pu réutiliser leur primitives et avoir ainsi nos propres spécification, certes moins abouties, mais beaucoup plus simples et lisibles.

Troisième partie

Les fonctionnalités ajoutées

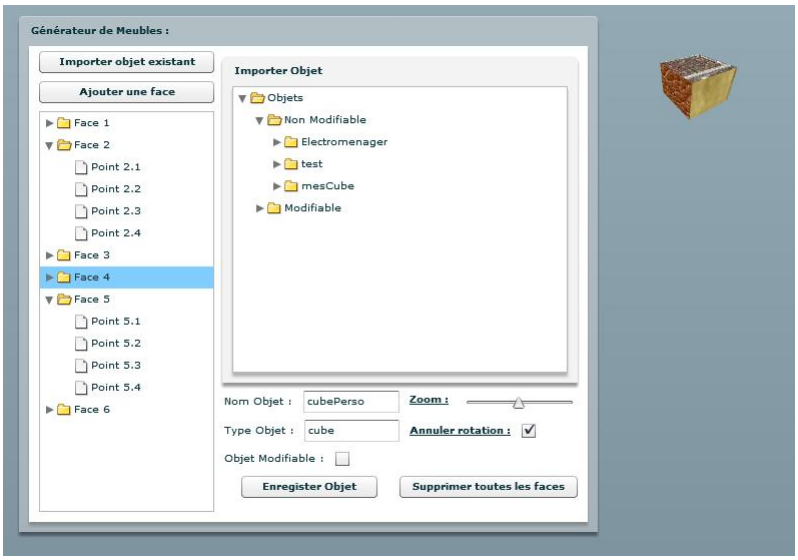
Chapitre 4

Générateur de meuble

4.1 Objectif

Le but du générateur d'objet est de pouvoir créer de manière intuitive n'importe quel objet 3D. La version précédente crée par les Master2 nous semblait trop limitée. En effet, il n'y avait que deux types d'objets proposés, il n'était pas possible de créer notre propre type d'objet. Nous avons donc décidé de concevoir notre propre générateur afin de pouvoir modéliser n'importe quel type d'objet.

4.2 Interface



Ci-dessus, l'interface du générateur d'objet. L'arbre de gauche énumère les différentes faces et leurs points respectifs. En cliquant sur l'une d'elles, le panel gestion apparait à droite, permettant éventuellement de changer la texture. En cliquant sur un point on a alors accès à son menu de modification, celui-ci permettant

de modifier ses coordonnées en x,y,z. Il est aussi possible d'ajouter un point qui sera alors inséré aux mêmes coordonnées que le point courant. Sur la droite de la fenêtre on peut apercevoir un fenêtre de prévisualisation pour avoir un aperçu de l'objet en cours de modification.

4.3 XML de lobjet en construction

Aperçu du fichier XML :

```
<!DOCTYPE objet SYSTEM "objet.dtd">
    <objet>
      <meta>
        <name modif="true">cube</name>
        <description/>
        <dimensions h="1" l="1" p="1"/>
      </meta>
      <composition>
        <face couleur="ffffffff" texture="mur.jpg" alpha="" nom="Face 1">
          <point x="0" y="0" z="0" nom="Point 1.1"/>
          <point x="0" y="0" z="1" nom="Point 1.2"/>
          <point x="1" y="0" z="1" nom="Point 1.3"/>
          <point x="1" y="0" z="0" nom="Point 1.4"/>
        </face>
      </composition>
    </objet>
```

Par rapport à un objet standard du Micromonde, nous avons rajouté un attribut nom à nos faces et nos points correspondant à leurs numéros respectifs dans l'objet (Face numéro de la face, Point Numéro de la face, numéro du point).

4.4 Création de l'objet

Lors de la création d'un objet, on ajoute d'abord une face, on défini les points de celle-ci et ainsi de suite (3 point minimum par face). Coté serveur, à chaque ajout de faces et de points un script php parse le fichier xml du nouvel objet, et ajoute les balises dans la partie composition. - Les faces sont ajoutées les unes à la suite des autres. - Les points peuvent être ajoutés entre deux autres points à la suite du dernier selectionné. - L'attribut nom d'une face est calculé à partir du nombre de faces précédentes. - L'attribut nom d'un point est calculé par rapport au précédent, et les suivant sont décalés. Les codes coté serveurs sont disponibles dans l'annexe.

4.5 Import/Export d'objet

4.5.1 Import

Le choix de l'objet à importer se fait dans un panel ouvert grâce au bouton importer objet, celui-ci contient un tree créé à partir du fichier xml référençant tous les objets existants. L'importation permet de réutiliser des objets déjà créés. Un script coté serveur parse l'objet à importer et copie le xml dans le fichier de l'objet à éditer en calculant le numéro des faces et des points. Un objet peut être ainsi réutilisé afin de créer de nouveaux objets. La modification à lieu sur une copie de l'objet afin de préserver l'original.

4.5.2 Export

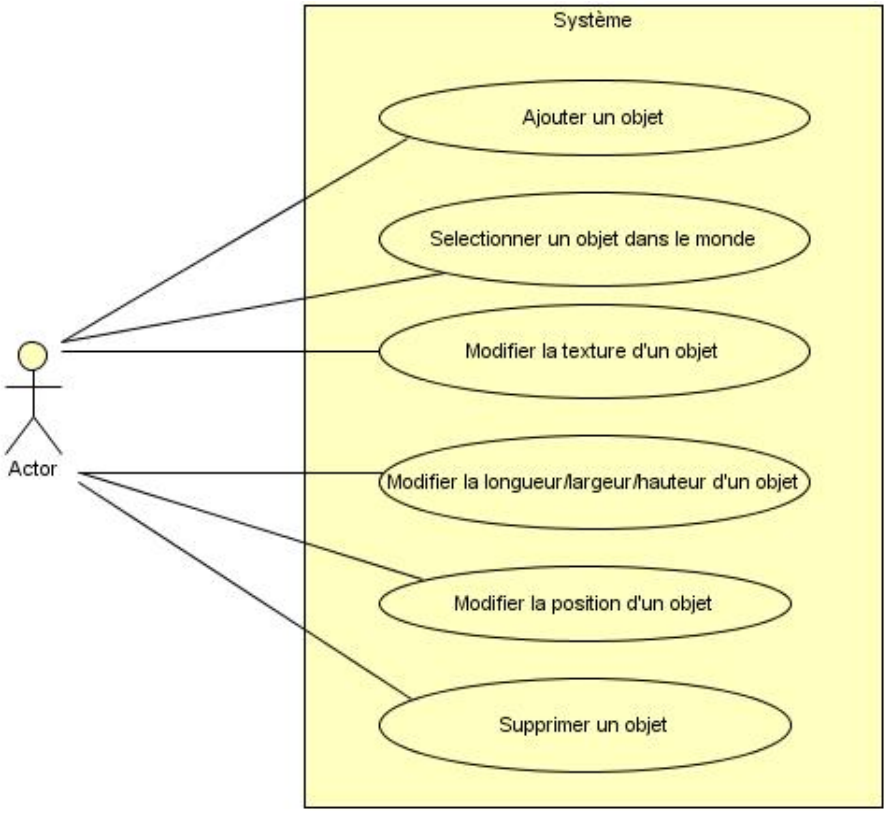
Une fois l'utilisateur satisfait de son objet, il peut l'enregistrer. Pour ce faire, il devra renseigner trois paramètres - Nom Objet nom que portera l'objet dans la collection - Type Objet Nom de la catégorie dans lequel l'objet sera inséré. Si celle-ci n'existe pas elle sera créée. - Objet Modifiable Défini un attribut modif. Si celui-ci est vrai l'objet créé sera avec une texture unique et il sera possible de le modifier directement dans le micromonde. Sinon les texture seront indépendantes et il sera uniquement possible de le déplacer et de le réorienter. L'enregistrement créé un nouveau fichier Xml portant le nom Nom Objet.xml. Lors de sa création, les attributs nom seront supprimés pour correspondre au modèle d'objet de base défini précédemment.

Chapitre 5

Gestion des objets

5.1 Objectif

Ce diagramme de cas d'utilisations permet de résumer les différents taches que nous avions à effectuer pour la gestion des objets.



Le but principal de ce TER était de pouvoir intéragir facilement avec les objets dans le monde et ce de manière dynamique afin que les utilisateurs puissent

aisément représenter l'environnement de leur choix. Cette gestion d'objet sous-entend l'ajout d'objet dans le monde, leur suppression, leur modification ainsi qu'une eventuelle composition.

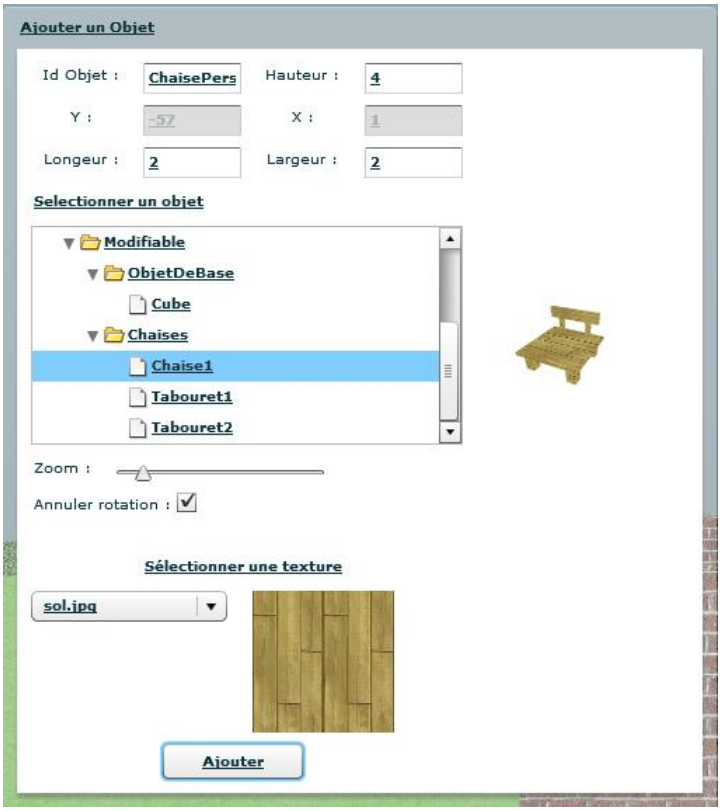
5.2 Nouveau fichier XML

```
<!DOCTYPE objet SYSTEM "objet.dtd">
    <objet>
      <meta>
        <name modif="true">cube</name>
        <description/>
        <dimensions h="1" l="1" p="1"/>
      </meta>
      <composition>
        <face couleur="ffffffff" texture="mur.jpg" alpha="">
          <point x="0" y="0" z="0"/>
          <point x="0" y="0" z="1" />
          <point x="1" y="0" z="1" />
          <point x="1" y="0" z="0" />
        </face>
      </composition>
    </objet>
```

Par rapport aux fichier Xml créé par le précedent groupe, nous avons rajouté dans la balise name l'attribut modif définissant si l'objet est éditable.

5.3 Ajout dans le monde

5.3.1 Interface



L'interface comprend plusieurs champs

- Id correspond à l'id donnée à l'objet lors de son ajout dans le monde.- Hauteur correspond à la hauteur de l'objet à insérer. - Longueur correspond à la longueur de l'objet à insérer. - Largeur correspond à la largeur de l'objet à insérer. - X et Y correspond aux coordonnées de l'objet dans le monde, ce champ sera rempli lors d'un clic à la souris sur le monde à l'endroit où insérer l'objet. - Sélection de l'objet se fait dans un tree renseigné par le fichier XML de collection d'objet. - Zoom et rotation ces deux composants permettent d'intéragir sur la prévisualisation de l'objet sélectionné. - Choix texture il se fait via une comboBox renseignée par un fichier XML contenant le nom des fichiers des textures. Le choix d'une texture fera apparaître un aperçu et modifiera la prévisualisation de l'objet.

- Ajouter objet ce bouton permet d'ajouter l'objet dans le monde aux coordonnées saisies.

Si l'objet sélectionné est dit « non-éditable » les champs de taille, longueur,

largeur et texture ne seront pas pris en compte.

5.3.2 Fonctionnement de la prévisualisation de l'objet

Lors de la sélection d'un objet dans le tree, on peut voir une prévisualisation de cet objet en rotation. Pour cela, nous avons écrit une nouvelle classe basée sur la classe microMonde, elle est en fait une version allégée d'un microMonde. Cette classe miniMicroMondeAffObj va lire un fichier XML ne comportant qu'un seul objet (celui sélectionné dans le tree).
Coté serveur, à chaque changement d'objet sélectionné, un script PHP va remplacer l'ancien objet par celui sélectionné dans le xml du miniMicroMonde. Lors d'un changement de texture, un autre script PHP est appelé pour remplacer la texture de toutes les faces de l'objet sélectionné dans son XML.

5.3.3 Fonctionnement de l'ajout d'un objet dans le microMonde

Premièrement, un script PHP va modifier le fichier XML définissant le monde en lui ajoutant une balise objet correspondante à l'objet à insérer. Pour effectuer celà il va parser le fichier XML, trouver la balise objets et y insérer une balise objet de ce type :

```
<objet id="table" hauteur="1" z="0" w="0" type="objPerso_table">
   <point x="51" y="-11"/>
   <point x="51" y="-12"/>
   <point x="50" y="-12"/>
   <point x="50" y="-11"/>
</objet>
```

Dans un deuxième temps, le script va recopier, dans un fichier XML, tout le fichier XML de l'objet sélectionné. En effet, tout les objets sélectionnables possèdent un fichier XML de description. C'est ce fichier qui sera copié pour éviter, lors de la modification d'un objet, de modifier tout les objets d'un meme type. Donc chaque objet ajouté dans notre monde possèdera son fichier de description XML, qui sera nommé objetPerso_id.XML. Pour finir, il ne reste plus qu'a ajouter l'objet dans notre monde. Une première approche a été de détruire le monde puis de le recréer à la fin de l'exécution du script PHP. Nous nous sommes vite rendu compte que cette méthode n'était pas performante car le chargement d'un monde complet demande un certain temps. Nous avons travaillé sur un ajout dynamique et donc créé une nouvelle fonction actionScript

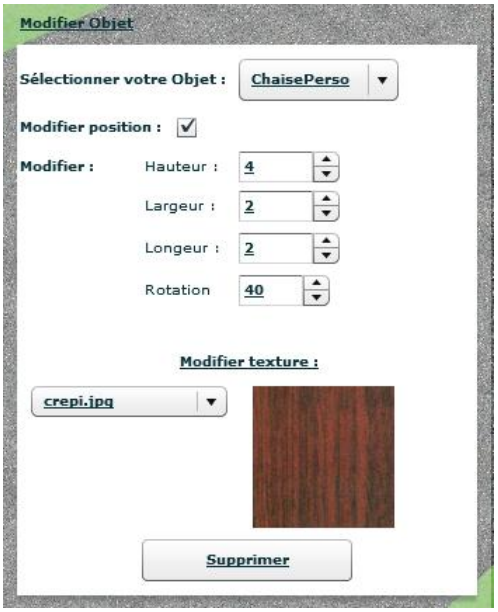
```
public function loadObjetsDynamicByName(modele :XML,id :String) :void {
    var scale :Vector3D ;
    for each(var obj :XML in modele..objet)
```

```
        if(obj.@id == id)
            var pts :Array = [] ;
            for each(var p :XML in obj..point)
                pts.push(new Point3D(p.@x, p.@y, obj.@z)) ;
                scale = new Vector3D() ;
                scale.x = Point.distance(new Point(pts[0].x, pts[0].y), new Point(pts[1].x,
pts[1].y)) ;
                scale.y = Point.distance(new Point(pts[0].x, pts[0].y), new Point(pts[3].x,
pts[3].y)) ;
                scale.z = obj.@hauteur ;
                scale.w = obj.@w * Math.PI/180 ;
                var objet :Objet = new Objet(obj.@type, pts[0], scale)
                addChild(objet) ;
                this.listObjets.push(objet) ;
            }
        }
    }
```

Cette fonction aura prend deux arguments le fichier XML de définition du monde et l'id de l'objet à insérer. Elle va donc rechercher l'objet à insérer avec l'id et l'ajouter au monde. Le gain de temps par rapport à la première méthode est significatif car nous ajoutons simplement l'objet à insérer sans recharger tout le monde. Une fois l'objet ajouté dans la description du monde nous appellons une fonction de rafraichissement qui mettra le monde à jour.

5.4 Modification d'objets

5.4.1 Interface



L'interface comprend plusieurs champs

- Sélection Objet cette comboBox contient tout les objets ajoutés au monde et permet de choisir un objet pour le modifier. Un objet aussi peut être sélectionné par simple clic dans le monde.

- Modification Position en sélectionnant cette checkBox nous pouvons déplacer l'objet d'un clic dans le monde. De plus, cette fonctionnalité nous permet de composer différents objets en posant par exemple un objet sur un autre.

- Modifier hauteur permet de modifier la hauteur de l'objet dans le monde.

- Modifier largeur permet de modifier la largeur de l'objet dans le monde.

- Modifier longueur permet de modifier la longueur de l'objet dans le monde.

- Modifier texture cette comboBox est renseigné de la même manière que celle de l'ajout d'un objet. Le changement de texture sera visible directement sur l'objet dans le monde.

- Suppresion permet de supprimer l'objet du monde.

Toutes ces champs n'affectent pas un objet non modifiable, hormis le changement de position et de rotation.

5.4.2 Fonctionnement de la modification d'un objet

Chaque élément de modification possède sont propre script PHP qui va modifier le fichier objPerso_<id>.XML de l'objet à modifier. Toutes les modifications ont lieu dans ce fichier sauf pour la modification de la rotation (celle-ci se faisant au niveau du fichier de description du monde, dans la balise objet sur l'attribut w).
Lors de chaque modification, nous effaçons l'objet du monde et nous rappelons la fonction loadObjetsDynamicByName pour réafficher l'objet après modification. L'effet graphique de cette dernière est intéressant car il fait comme une preview des futures dimension de l'objet.

5.4.3 Fonctionnement de suppression d'un objet

Lors de l'appui sur le bouton Supprimer un script PHP va supprimer l'objet dans le fichier de description du monde, grace à son id. Le fichier de description de l'objet objPerso_<id>.XML va lui aussi être détruit et au niveau du monde l'objet sera supprimé de la scène dynamiquement.

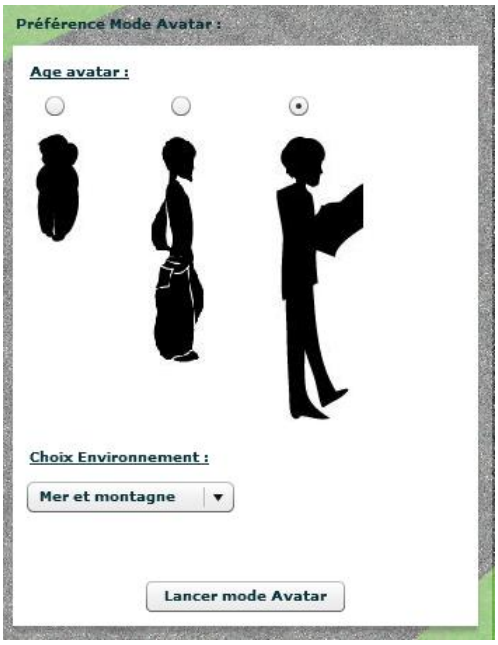
Chapitre 6

Mode avatar

6.1 Objectif

Après avoir répondu au cahier des charges qui comprend en grande partie la gestion des objets, nous avons pensé qu'il pourrait être intéressant de pouvoir visiter le monde en vue à la première personne et non avec un vision globale de l'éditeur. Nous avons donc réalisé un mode simulant un avatar.

6.1.1 Interface



- Sélectionner âge groupe de boutons radios permet de sélectionner l'âge de l'ava-

tar afin d'avoir une taille approximative dans le monde (Enfant,Adolescent ou Adulte).

- Choix environnement cette comboBox permet de sélectionner l'environnement autour de notre monde. C'est à dire, le fond visuel et sonore choisi.

- Lancer le monde Avatar Nous place dans le peau de notre avatar dans notre monde.

6.2 Fonctionnement de l'avatar

Notre micro-monde possède une caméra, pour pouvoir la déplacer en mode éditeur nous utilisons une primitive d'Alternativa 3D qui est un cameraController. Ce contrôleur nous permet de déplacer la caméra dans notre micro monde. Pour le mode avatar il nous a suffit d'utiliser à la place d'un cameraController un walkController. Le changement entre les deux contrôleurs se fait au niveau de la fontion onEnterFrame

```
private function onEnterFrame(e :Event) :void {
        if(fps){
            this.fpsController.processInput();
            if (!fpsController.onGround || (fpsController.currentSpeed == 0))
            soundManager.stopSteps();
        }
        else{
            soundManager.playSteps(SoundManager.STONE_WALK);
        }
    }
    else{
        cameraController.processInput();
     }
    scene.calculate();
}
```

Lorsque le mode Avatar est activé, nous passons dans la condition et nous activons le walk_controller sur la caméra, ce qui simule un déplacement de type humain. Quand nous repassons en mode éditeur, nous sautons la condition de la boucle dans la conditionnelle. Ainsi, c'est le cameraController basique qui est activé.

6.3 Environnement sonore

Afin de rendre d'augmenter l'immersion de l'utilisateur, des sons sont joués afin de créer une ambiance dans le monde. Les sons joués dépendent de l'environnement visuel. Ils sont gérés par la classe soundManager, (classe issue d'un exemple d'Alternativa 3D que nous avons modifié pour les besoins du projet). Elle permet de jouer un son constant de type ambiance, ainsi qu'un son de pas à chaque déplacement du walk_controller. Ceci se déroule au niveau de la fonction onEnterFrame, quand on est en mode avatar si le controller est sur un objet et que la vitesse est supérieure à 0, alors le son de pas est joué. Quand on repasse en mode éditeur, tout les sons sont coupés.

6.4 Environnement visuel

Afin de rendre notre monde encore plus immersif, un environnement visuel est ajouté tout autour du monde. Il y a deux environnements disponibles, soit un environnement composé d'une plage avec des massifs montagneux, soit une montagne enneigée. Il est géré par la classe environnement qui est issue de l'exemple précédent. Elle crée un très grand cube tout autour du monde, qui sera ensuite mappé avec un panorama cubique. Cet environnement est ajouté lors du passage en mode avatar. Lors du passage en mode éditeur, cet environnement est retiré.

Quatrième partie

Conclusion

En arrivant au bout de notre TER nous avons essayé de correspondre le plus possible au cahier des charge du projet. Celui-ci, un peu flou, nous permettait une certaine flexibilité au niveau fonctionnalités, ceci en rapport avec la nature étendue du projet micromonde. Ainsi nous avons donc géré l'édition de meuble dans son ensemble, que ce soit l'ajout d'un meuble dans le monde, la modification de ses caractéristiques, sa suppression ou même, la composition. Nous avons aussi créé notre propre éditeur de meuble pour plus de possibilités de création et de personnalisation du monde. Une fois ces objectifs principaux atteint nous avons implémenté une fonctionnalité Mode Avatar nous permettant de nous déplacer dans le monde avec une caméra en vue à la première personne. Cette caméra est d'ailleurs paramétrable en taille en fonction de l'âge de l'avatar que l'on souhaite. Pour une plus grande immersion nous avons aussi ajouté au monde en mode avatar une ambiance sonore et visuelle.

Lors de notre TER nous avons rencontrés deux difficultés majeures. Premièrement, une difficulté récurrente, à savoir reprendre un projet déjà existant, comprendre le fonctionnement, les bugs. Dans un second temps le format Collada. Après plusieurs essais nous trouvions le code du format trop complet, c'est pour cela que nous avons regardé du coté de certaines applications pour savoir si on ne pouvaient pas les piloter en ligne de commande. Ainsi après y avoir consacré deux semaines nous avons préféré ne pas perdre d'avantage de temps dessus, de plus nous avions le format XML du projet du groupe des Master2 qui était plus simple et plus adapté au projet.

Afin que le projet Micromonde continue à évoluer, nous avons réfléchi aux perspectives de développement - Utiliser la technique de mappage du paysage du mode FPS pour appliquer des textures panoramiques sur les murs dans le monde.
- Faire évoluer le mode avatar de manière à afficher les autres personnes connectées.
- Mettre en place le système réseau de chat commun à tous les jeux sociaux.