Interfaces graphiques 3D orientés Web

Travail de diplôme réalisé en vue de l'obtention du diplôme HES

par :

Renaud SAUVAIN

Conseiller au travail de diplôme :

Peter DAEHNE, Professeur HES

Genève, 5 septembre 2008

Haute École de Gestion de Genève (HEG-GE)

Filière Informatique de Gestion

Déclaration

Ce travail de diplôme est réalisé dans le cadre de l'examen final de la Haute école de gestion de Genève, en vue de l'obtention du titre de bachelor of science. L'étudiant accepte, le cas échéant, la clause de confidentialité. L'utilisation des conclusions et recommandations formulées dans le travail de diplôme, sans préjuger de leur valeur, n'engage ni la responsabilité de l'auteur, ni celle du conseiller au travail de diplôme, du juré et de la HEG.

« J'atteste avoir réalisé seul(e) le présent travail, sans avoir utilisé des sources autres que celles citées dans la bibliographie. »

Fait à Genève, le 05.09.08

Renaud Sauvain

Remerciements

Ce travail de diplôme ayant été mené à bien avec le soutien de quelques personnes, je veux ici leur adresser mes meilleurs remerciements.
Je voudrais notamment citer les personnes et entreprises ayant donné de leurs temps pour répondre à mes questions concernant l'utilisation des interfaces 3D orientés Web, dont : Benjamin Pugliese (Iomedia), Vincent Greset (Innovagency), Luc St-Arnaud (Optaros) et M Wong (W3 Media).
Un grand merci à Romain Sauvain et Christine Bovet pour leur soutien et conseils avisés, ainsi qu'à M Peter Daehne qui m'a suivi tout au long de ce projet.

Sommaire

Représentant le meilleur de la technologie informatique, démonstration de la puissance de calcul des ordinateurs dernier cri, depuis toujours associé à une atmosphère futuriste et passionnante, l'affichage en trois dimensions a depuis des années envahi le contenu vidéo ludique de nos ordinateurs. Malgré tout, ces démonstrations se contentent d'apparitions craintives sur la toile. Ce travail à pour but principal de mettre en avant ces technologies et leurs utilisations et ainsi de donner un aperçu global du monde de la 3D sur internet.

Ce document est avant tout destiné à toutes personnes désireuses de s'informer sur l'état des technologies actuelles, mais également à être un point de départ pour une entreprise désireuse de se lancer dans un tel développement. Pour la majeure partie, ce dossier se veut accessible à tous, et aucune connaissance spécifique n'est préalablement requise. Il faudra néanmoins une certaine familiarité avec la syntaxe des divers langages de programmation utilisés pour pouvoir passer de cette étude à une réelle mise en place de solution.

Les informations relevés dans ce dossier ont été principalement puisées de l'intarissable source qu'est internet, la bibliographie classique étant pour ainsi dire inexistante sur le sujet. Pour cela, cette recherche est un reflet de la principale tâche qui fut le voyage virtuel, parmi les nombreux sites, forums, articles et tutoriels détaillant du sujet sur la toile.

Débutant par l'étudie des utilisations actuelles des affichages 3D sur internet, un tour d'horizon des technologies existantes est effectué, apportant une vue d'ensemble au plongeon dans les technologies nécessaires aux développements constituant la partie centrale de notre recherche. Celle ci démontre comment de telles solutions peuvent être effectivement mises en place; elle met en valeur deux exemples d'utilisation et de technologies distinctes. Lors de ces réalisations pratiques nous passerons sous la loupe les technologies choisies, la mise en place de leur solution, le déploiement de celle-ci sur un site internet et l'étude de points précis et de certains problèmes rencontrés.
En guise de clôture, une analyse des possibilités futures et des acteurs principaux du domaine sera menée, suivie d'une brève étude du marché local en Suisse.

La 3D est elle viable comme technologie commerciale ? Comment sera le Web de demain? Ce travail n'a pas la prétention de pouvoir répondre à de telles questions, mais bien de donner les grandes lignes directrices du domaine, permettant à chacun d'émettre son opinion en disposant d'un maximum d'informations et d'éveiller l'intérêt pour ces technologies.

Table des matières

Liste des Figures

1 Introduction

1.1 Concept général

Le débit des connections internet augmentant sans cesse, les utilisateurs sont en droit d'attendre de plus en plus de sites web interactifs et présentant une interface évoluée. Étant donné l'évolution impressionnante du contenu on-line que nous avons connue ces dernières années, que pouvons nous attendre du contenu de l'Internet de demain?

La science-fiction nous a depuis longtemps montré l'image de personnages évoluant à l'intérieur de mondes virtuels, immersifs et offrant à l'utilisateur des paysages futuristes. Comme cela est déjà arrivé à de nombreuse reprises, la réalité va-t-elle rejoindre, voire dépasser la fiction?

Depuis quelques années, le surfeur intrépide a eu l'occasion, lors de balades sur la toile, de rencontrer des interfaces de discussion dans lesquels chaque internaute est incarné par un avatar se mouvant dans un environnement en trois dimensions. Malgré tout, ces cas restent relativement anecdotiques et ces mondes de réalité virtuelle peu accessibles au néophyte.

Sans étendre le débat à une vue à trop long terme, que sommes-nous, utilisateurs, en droit d'attendre de la réalité virtuelle on-line et plus précisément des technologies permettant l'affichage en trois dimensions sur internet? Quels outils et technologies avons-nous à disposition ? Pour quel usage peuvent-ils être mis en place? De tels développements sont-ils accessibles ou resteront-ils réservés à des passionnés?

Nous allons tenter de répondre à ces diverses questions lors de cette étude qui a pour objectif d'établir une vue générale des technologies contribuant au développement d'interfaces complexes imbriquées web, d'analyser les modalités de leur mise en place et de proposer des exemples de réalisations concrètes les employant.

Ce rapport débute par une présentation des technologies actuelles ainsi que de leur utilisation actuelle et future. Ce premier chapitre posera les bases et donnera la vue d'ensemble du problème traité dans ce document. Suivra un exercice appliqué présentant la mise en place de technologies parmi les plus répandues. Pour conclure,

fort d'une recherche théorique et de réalisations pratiques nous tirerons les points forts et soulignerons les éléments à améliorer dans ce que nous appellerons le Web 3D.

1.2 Cas d'utilisation

Comme c'est souvent le cas dans le cadre de nouvelles technologies, les interfaces graphiques offrent une multitude de possibilités aux développeurs de sites Internet. Ce chapitre va essayer de répondre à la question « Quelles sont concrètement les utilisations faites actuellement de ces technologies? » et ainsi donner un aperçu global des développements réalisés et présentés sur la toile.

1.2.1 Des vitrines commerciales ou touristiques

L'utilisation la plus répandue d'un point de vue commercial à l'heure actuelle, et qui représente une majeure part du marché pour les développeurs 3D orienté Web, consiste dans la présentation de vitrines virtuelles.

En effet, de nombreux sites commerciaux présentent dorénavant leurs articles sous formes d'objets en trois dimensions, permettant à l'utilisateur de visualiser sous toutes leurs formes, sous tous leurs angles, les produits disponibles. Certains sites proposent des fonctions de personnalisation avancées permettant de visualiser un produit avec différentes options, notamment en termes de couleurs ou de matériaux. Les domaines tels que la construction automobile, l'horlogerie, la téléphonie portable ou le matériel de sport proposent de nombreux exemples de réalisation de ce type. En effet les marques Omega, Breitling, Nokia, Kodak et bien d'autres utilisent ce concept pour la présentation de leurs produits sur leur site internet.

Une telle visualisation encourage les clients à regarder l'objet convoité, choisir les couleurs, les options disponibles et le comparer visuellement avec d'autres produits. Comme le souligne le site reality_prime.com, une vitrine virtuelle permet la création de « connections émotionnelles » entre le client et le produit. Il est notamment relevé sur ce site que le degré de satisfaction des clients concernant leur expérience Web monte de 63% à 90% lors d'une migration réussite d'une présentation de produits de deux à

trois dimensions.

Figure 1

Exemple de vitrine virtuelle

Source : breitling.com(2008)

Ce coté ludique est un atout marketing non négligeable qui apporte ainsi une plus-value intéressante par rapport aux présentations standard.

D'autres corps de métier comme le génie civil et l'architecture ne sont pas en reste, offrant sous forme virtuelle leur vitrine architecturale.

Ces utilisations de vitrines virtuelles ne se limitent pas exclusivement à des produits commerciaux. En effet, certaines villes proposent, en guise de promotion touristique, une visite virtuelle de leurs rues. Camogli, St Nazaire, Las Vegas, New York, Ottawa, Toronto, Vancouver, Boston, Tokyo et bien d'autres villes encore ont ainsi leur homologue virtuel.

Dans une optique similaire, certains monuments tels que la tour Eiffel ou l'église de St Pierre aux Nonnains et des musées tels que lemuseevirtuel.com, le musée canadien des civilisations ou l'Université de Californie commencent à apparaitre pour offrir un échantillon virtuel de leurs œuvres. Notons que ces technologies ont permis à certains musées de montrer des collections jusqu'alors inaccessibles au public et qui, en raison de leur trop grande fragilité, ne pouvaient être exposées directement.

1.2.2 Des réalisations à but ludique

Une autre facette des plus représentatives de l'utilisation de ces technologies est le domaine du ludique. En effet, il est fréquent de trouver des jeux en trois dimensions disponibles d'un simple clic sur internet[1]. La gamme de sites référençant ce type d'application vidéo ludique est trop vaste pour être mentionnée ici de façon exhaustive. Citons néanmoins le MMORPG Wurm Online, qui offre un univers d'une taille et d'une réalisation approchant les jeux commerciaux classiques qui disposent d'un client lourd, c'est-à-dire requérant une installation locale. Cette réalisation illustre ainsi pleinement les possibilités offertes par ces technologies.

Une autre utilisation commune des technologies 3D Web dans un but ludique réside dans les différentes chambres de discussion virtuelles communément appelées chat.
Dans celles-ci, chaque utilisateur, sous l'apparence d'un avatar, a la possibilité de communiquer en direct avec d'autres personnes connectées dans un environnement en trois dimensions.
Ces « chatrooms » sont référencées en nombre par des portails spécialisés tels que www.mondesvirtuels.com, www.virgal.net et bien d'autres.

1.3 Technologies existantes

Dans le chapitre précédent, nous avons vu ce que peu apporter une interface graphique intégrée au navigateur Web. Se pose maintenant la question du « comment ». Une telle réalisation est effectivement plus complexe qu'une réalisation « classique » construite autour des technologies HTML, PHP, JavaScript voir Flash.
Pour ce faire, de nombreuses technologies sont actuellement disponibles. Celles-ci sont plus que variées tant au niveau de leurs spécificités que de leurs aboutissements. Il est donc primordial d'avoir un bon aperçu des possibilités offertes par chacune d'entre elles pour pouvoir, dans chaque cas, choisir la solution la mieux adaptée aux besoins. Nous nous proposons ci-après de proposer quelques exemples d'outils disponibles, liste non-exhaustive des solutions technologiques possibles. Il convient en outre de en garder en mémoire que celles-ci peuvent évoluer rapidement en raison de

1 Nous n'inclurons pas dans nos exemples les fameux Google Earth ou Second life qui nécessitent l'installation de programmes clients sur les ordinateurs des utilisateurs.

la volatilité élevée des technologies Web.
Il est à noter que l'affichage 3D n'est pas offert nativement dans les navigateurs actuels. Pour obtenir une telle visualisation, l'ajout d'un plug-in est nécessaire et cela quelle que soit la technologie utilisée.

VRML

Précurseur dans le domaine, présenté en 1994, le VRML, standard de l'International Standards Organisation (ISO) pour la représentation d'objets 3D, fut le support de nombreux chats 3D. Basé sur la technologie XML, et pouvant accueillir du script java, ces fichiers (format .wrl) nécessitent l'installation d'un lecteur VRML, pouvant être installé comme plug-in dans la plupart des navigateurs actuels. Les fichiers VRML peuvent être créés à partir de certains modeleurs comme Blender par exemple. Malheureusement la technologie VRML ne supporte ni l'accélération matérielle[2], ni la gestion d'effets graphiques complexes, ni l'ajout de nouveaux modules. Par conséquent, la qualité du rendu n'atteint pas le niveau que l'utilisateur a l'habitude de voir dans le contenu vidéo ludique actuel. Un autre frein à la diffusion du VRML consiste dans le fait que les lecteurs VRML ne sont pas dans tous les cas compatibles avec le contenu d'une page développée pour un autre visionneur. Par conséquent l'utilisateur sera amené à installer plusieurs plug-ins différents, rompant ainsi avec le concept de standard compatible qui était l'objectif principal du VRML.

3DML

Basé sur le standard XML, ce langage permet de décrire d'une manière extrêmement simple un environnement en trois dimensions basé sur une combinaison de blocs par « level ». Pour décrire un tel environnement, on procède en deux étapes : la définition des blocs à utiliser, puis le placement de ceux-ci dans un tableau. Le développement de présentations simples est donc aisé, rapide et utilise une taille de fichier réduite. La contrepartie est évidemment la difficulté, voir l'impossibilité de créer des scènes complexes, ainsi que l'absence totale de langage de script.

[2] L'accélération matérielle consiste dans l'utilisation du GPU, processeur dédié aux calculs d'affichage. A l'opposé le rendu logiciel est effectué par le CPU, processeur principale d'un ordinateur.

X3D

Successeur du VRML, le X3D, comble les manques graphiques de son prédécesseur en permettant l'ajout de modules et donc de nouvelles fonctionnalités graphiques. Le X3D pourrait donc constituer une alternative de choix lorsque cette technologie sera arrivée à maturité. Un navigateur X3D open-source est également en phase de développement, le XJ3D. Le X3D étant soutenu par la communauté du Web 3D Consortium, de nombreux exemples de codes et de projets sont disponibles sur internet.

JAVA (Sun Microsystems)

Ce langage de programmation bien connu offre, grâce à ses librairies d'applets et graphiques, la possibilité d'afficher des rendus 3D complexes de haute qualité. Le langage java étant libre, il a généré de nombreuses communautés de développeurs. Par conséquent, de nombreuses librairies sont disponibles.

Il y a actuellement deux choix de librairies permettant l'affichage 3D à l'aide de JAVA : JAVA3D' et JOGL

- **JAVA3D** est une API graphique de haut niveau pour Java, supportant OpenGL et DirectX. Un moteur graphique de base est déjà implanté ; il permet une réalisation plus rapide mais moins détaillée qu'avec JOGL. Malheureusement cette technologie ne sera plus supportée dans le futur.
- **JOGL** est une libraire reprenant les fonctions de l'API OpenGL. Par conséquent, tous les effets graphiques de haut niveau sont gérés. Cela permet également de bénéficier d'une documentation complète et d'une utilisation de code provenant d'autres langages. Jogl étant donc une librairie OpenGL pour Java, il s'agit d'un langage de bas niveau qui n'offre pas de moteur graphique intégré. D'autre part, les moteurs disponibles sont malheureusement peu nombreux. jMonkeyEngine est l'un de ceux présent actuellement. Orienté dans le développement de jeux 3D, il parait être le plus abouti. Dans la majorité des autres cas, il sera donc nécessaire de développer soi-même le moteur graphique, ce qui est malheureusement extrêmement chronophage.

Flash 3D

Technologie bien connue des développeurs Web, Adobe Flash (anciennement Macromedia) permet un affichage vectoriel, donc rapide et léger dans les navigateurs. Suite au succès énorme de cette technologie (90% des ordinateurs sont équipé du plug-in flash), de nombreux add-ons furent développés, dont certains convertissant des

images 3D en 2D. Des logiciels de modélisation 3D firent donc leur apparition sur le marché, ceux-ci gérant un moteur graphique capable autant de modeler des objets que d'en importer depuis d'autres formats comme par exemple le .3ds. Les plus connus de ces logiciels sont : Swift3D, Away3D, Electric3D, Papervision3D, Sandy, Alternativa3D. Les gros points fort de cette alternative sont, la légèreté du format flash, bien adaptée à la diffusion Web, et la très large diffusion du plug-in Flash. Destinées à des présentations simples, les scènes complexes auront quant à elles souvent du mal à trouver leur place dans le cadre de ce format.

Shockwave3D

Il s'agit d'un plug-in développé par Adobe permettant l'affichage de scènes crées sous Adobe Director dans un browser Web. Celui-ci était conçu initialement pour produire des animations, il changea radicalement d'orientation lors de l'ajout d'un langage de script, ce qui étendit ses capacités au développement d'interfaces, d'applications multimédia ou encore de jeux vidéo.

Acrobat3D

Bien que ce format ne soit pas une technologie embarquée dans le navigateur à proprement dit, il mérite qu'on s'y attarde au vu des possibilités offertes. Après le succès phénoménal du format PDF d'Adobe Acrobat, il était normal qu'Adobe développe ce nouveau type de format, permettant l'intégration d'animations multimédia. Ceci a été concrétisé avec Acrobat3D, outil qui supporte le U3D, le nouveau standard de fichier 3D développé par le consortium « 3D industry forum ». Adobe propose donc un modeleur 3D offrant entre autre la possibilité de « capturer » une scène d'un programme de modélisation 3D et de la coller directement dans Acrobat3D. Ce mode de diffusion comporte tout de même des limitations, notamment au niveau de l'interactivité. Il offre cependant une possibilité de créer des documents contenant des éléments graphiques en trois dimensions dont la diffusion est grandement facilitée par la popularité d'Adobe Acrobat Reader.

Un bon nombre d'autres solutions existent, la majorité étant des « modeleurs 3D », permettant l'infographie 3D complétée par l'ajout de scripts simples et par l'import de fichiers tel que le .3ds. Chacun utilise son propre plug-in, disponible pour Internet Explorer et, généralement, Firefox. Voici une liste de différentes solutions propriétaires disponibles : Octagon, Subdo, Viewpoint, Anark, Cosmos, Seemage, Cyan3d, Alambik,

Wirefusion, quest3d, TurnTool, Virtools,…

en conclusion, ces nombreuses solutions peuvent être regroupées en quatre catégories :

1. Les solutions bâties autour de la technologie Java, open source, très largement répandues, et disposant de nombreuses librairies, tutoriels et forums de discussions.

2. Les solutions propriétaires, telles que Adobe, Flash, Acrobat3D et Shockwave3D qui disposent d'une couverture massive sur le Web, ainsi que d'un prix attractif et d'une durée de vie certaine au vu de la position d'Adobe sur le marché.

3. Les solutions standardisées que sont le 3DML, le VRML et l'X3D. Le second ayant depuis longtemps atteint ses limites, tout les regards sont tournés vers le X3D, déjà utilisé par nombreux programmes propriétaires tels que Openworlds.

4. Les autres programmes propriétaires, généralement dédiés à des utilisations spécifiques, principalement commerciales. Ils permettent d'avoir accès à des modeleurs de haut niveau, offrant de bons rendus, sans besoin de compétences en programmation. En contrepartie, leurs offres ont des tarifs pouvant être élevés autant en coûts de licence qu'en royaltises.

Il est à noter que beaucoup de technologies 3D sont en voie d'abandon, si ce n'est même déjà le cas (3dml, Scol, VRML, Adobe Atmosphere et bien d'autres). C'est l'une des raisons pour lesquelles il est particulièrement important de bien choisir la technologie à utiliser. En effet un langage qui n'est plus supporté deviendra vite inutilisable et donc inutile.

1.4 Alternatives

Le développement 3D engendrant fréquemment un temps travail important et, par conséquent, des coûts élevés, des technologies palliatives ont été élaborées pour permettre une plus grande démocratisation de l'accès aux effets 3D sur internet.
En effet, la modélisation d'un objet et son rendu peuvent être contournés par des effets imitant une vue en trois dimensions. Il en existe une large variété, développée en

raison de la nécessité de s’adapter à la diversité des besoins spécifiques.

L'une d'entres elles, consiste en un diaporama d'images qui représente chacune d'elles la scène sous un angle précis. Cette succession d’images forme une animation contrôlable, donnant ainsi à l'utilisateur, par exemple, l'impression de pouvoir faire pivoter l'objet sur lui-même sur un axe, alors que, techniquement, il ne regarde qu'une image statique. L'énorme avantage de ce choix est la qualité d'image photo réaliste qui peut être obtenue tout en offrant une vue pseudo-interactive de l'objet.
Ce système entraîne évidemment de nombreuses limitations. Citons l'impossibilité d'obtenir une navigation de la caméra sous plusieurs axes, ainsi que la baisse de qualité en fonction du nombre d'images du diaporama. D’autre part, le changement d'une option telle que la couleur, nécessitera la création d’un nouveau diaporama complet. Ce système prendra donc, selon la qualité d'image voulue, une taille conséquente surchargeant facilement les petites connections. Pour exemples, les pages de présentation des entreprises Salomon ou BMW utilisent un tel procédé.

Parmi les autres alternatives répondant à un besoin plus spécifique, citons l’exemple du « panorama 3D » consistant dans la présentation de paysages statiques, mais permettant une navigation interactive à l’utilisateur. Une image couvrant 360 degrés de la vue à représenter peut ainsi être utilisée pour donner l'impression d'une vue virtuelle à l'utilisateur qui aurait la possibilité de « tourner » la tête. De nombreuses villes utilisent cette solution et offrent ainsi une sélection de leurs panoramas sans avoir recours à une modélisation lourde, comme c'est le cas avec une technologie VRML ou équivalente. De tels exemples sont disponibles sur les sites touristiques des villes d'Yverdon-les-Bains ou de Morges.

2 Projet de développement

Comme nous l'avons vu au premier chapitre de ce travail, le monde de la 3D pour internet est en constante mutation. Il est par conséquent ardu pour un néophyte de déterminer les étapes clefs ainsi que les difficultés associées à la mise en place d'un tel projet. Ce deuxième chapitre a pour but d'offrir des exemples simples mais concrets de telles réalisations et ainsi de mettre en évidence les étapes significatives d'un tel projet. Pour ce faire, nous allons effectuer le développement de deux applications illustrant les possibilités offertes par l'utilisation de deux des technologies majeures mentionnées précédemment.
Nous allons ici aborder le côté technique du Web 3D et ainsi montrer concrètement ce à quoi une entreprise désireuse de développer une interface de ce type va être confrontée.

2.1 Choix du type d'application

Lors de notre premier chapitre, nous avons présenté un grand nombre d'utilisations possibles pour une interface capable de rendu 3D sur des sites internet. Nos exemples devant être les plus représentatifs des utilisations actuelles, nous allons développer deux applications dont les buts et conceptions sont radicalement différents.

Nous l'avons vu précédemment, le type de demande commerciale le plus répandu dans le domaine de la 3D pour internet consiste dans la représentation d'une vitrine de produits virtuels. Nous allons donc montrer ce à quoi une entreprise pourrait être confrontée dans le cas d'un développement similaire.

Notre seconde illustration, à l'opposé du but purement commercial, se doit d'être une représentative des possibilités graphiques évoluées. Nous proposons par conséquent à la fois un aperçu de haute qualité visuelle et une interaction avancée avec l'utilisateur. Pour ce faire nous choisirons l'un des autres domaines dominants de la 3D sur internet, à savoir le développement vidéo ludique. En effet, un tel projet représente bien plus qu'un affichage dans une page Web. Il s'agit en effet du développement d'une application Web complète avec toutes les possibilités d'interactivités qui y sont

nécessairement associées. Nous allons nous tourner vers un choix classique et dont le temps de développement reste raisonnable dans le cadre de ce travail de diplôme : le jeu d'échecs. Ce développement vise à d'obtenir un bel échantillon des possibilités données aux développeurs Web.

2.2 Choix de la technologie

Les objectifs de nos développements étant fixés, nous allons pouvoir choisir les technologies les plus adaptées à ceux-ci.

Nous allons premièrement écarter les technologies les plus anciennes, celles-ci n'offrant plus ce que les cyber-surfeurs actuels sont à même d'attendre. Leur utilisation déclinant peu à peu, le support des navigateurs et des plugins tend à disparaître, ce qui engendre aussi bien des problèmes de développement que d'installation pour le client utilisateur. Enfin, ces technologies ont été, dans bien des cas, l'objet d'ouvrages vers lesquels nous redirigeons les lecteurs éventuellement intéressés.

Dans le cadre de ce travail, nous allons également écarter les solutions propriétaires pour plusieurs raisons. La première est avant tout le caractère onéreux de celles-ci. En effet, le coût de licence associé peut être fortement élevé. De plus, l'accès à leurs matériels de développement (SDK), comprenant tutoriels, documentation, exemples, et utilitaires associés, n'est généralement disponibles qu'avec l'obtention d'une licence. Pour finir, ces éditeurs offrent généralement un « modeleur » affichant directement des objets modélisés au préalable avec l'aide d'un logiciel de CAO. Ceci implique un accès limité, ou inexistant, aux technologies d'affichage elles-mêmes, ce qui n'est pas le but de ce travail.

Notre exemple de vitrine commerciale implique un certain nombre de contraintes qui vont guider notre choix de technologie.

Nous cherchons à optimiser les coûts de développement, et donc, de produire un résultat utilisable avec un minimum de ressources. Celles-ci sont, en l'occurrence, le temps de prise en main de la technologie, le développement à proprement parler ainsi que la diffusion du logiciel exécutable sur Internet. Nous allons pour ce faire nous tourner vers des solutions ayant le support d'une large communauté et ainsi bénéficier

d'un accès aux informations nécessaires à une prise en main efficace. Un format standardisé est indiqué pour apporter un maximum de compatibilité avec les navigateurs des clients potentiels de notre vitrine 3D. Pour différentes ces raisons, notre choix de technologie se portera sur le successeur du VRML le X3D, dont la syntaxe basée sur le format XML, permet une prise en main rapide et emploie une taille de fichier particulièrement adapté à ce type d'application web.

X3D étant le standard plein de promesses du web consortium, il est intéressant de voir ce que cette technologie est à même de nous apporter.

Notre choix pour le développement d'une application ludique sera quant à lui dicté par le besoin de flexibilité, de qualité graphique et d'interactivité, associé à ce type de logiciel. Nous allons donc écarter les solutions n'offrant pas un niveau de qualité graphique ou d'interactivité suffisante (tel que le VRML, 3DML,..), tout en restant dans les formats les plus utilisés pour bénéficier d'une diffusion maximale. Notre choix se restreint donc aux solutions proposées par Java, X3D, et les dérivés de flash. Nous écartons le X3D qui est l'objet de notre premier exemple. Les technologies basées sur le format Adobe Flash, bien que pouvant offrir une grande interactivité ou un rendu correct, ont du mal à associer les deux.
Du fait que les technologies basées sur Flash sont propriétaires et qu'elles offrent plus des outils qu'un véritable langage de programmation, nous allons finalement écarter cette solution.

Le langage de SUN sera donc notre choix pour réaliser notre exemple vidéo-ludique.

2.3 Premier développement, la vitrine virtuelle

2.3.1 Le mandat

Dans le cadre de ce premier développement, notre objectif est de réaliser la visualisation d'un objet commercial en trois dimensions. L'utilisateur aura la possibilité d'observer le produit sous tous les angles possibles, c'est-à-dire de pouvoir effectuer une rotation de 360° autour de celui-ci ainsi que de sélectionner diverses options qui seront affichées de manière dynamique.
Une contrainte importante que nous nous sommes fixée pour notre vitrine est de laisser la possibilité de suivre l'évolution de l'objet commercial. Notre client doit ainsi

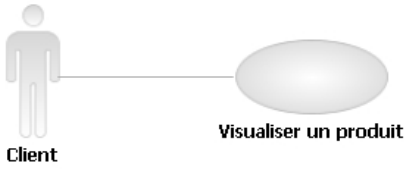
être en mesure de changer les modèles d'objets à afficher sans qu'il soit nécessaire de re-développer le système. Concrètement, cela signifie que la forme de notre objet ne devra pas être contenue dans notre logiciel mais chargée depuis une base de données lors son lancement.

Pour notre exemple, nous avons choisi arbitrairement le type de produit virtuel à proposer. Ce choix est limité par le fait que la modélisation d'objets en trois dimensions requiert un travail graphique important et, par conséquent, notre objet sera des plus simples. Une pièce de mobilier sous la forme d'un modèle de table composé d'un plateau ainsi que d'un système de pieds, conviendra parfaitement à notre démonstration. Ces deux parties seront modifiables dynamiquement par l'utilisateur, autant par leur forme que par leur couleur.

Notre cahier des charges :

1) Développer un programme léger offrant un temps de développement réduit.
2) Obtenir une qualité graphique suffisante pour l'affichage d'un produit commercial.
3) Offrir un mode de visualisation permettant d'obtenir toutes les vues désirées du produit.
4) Offrir la possibilité de modifier la forme et la matière de celui-ci.
5) Mémoriser les modèles trois dimensions dans une base de données.

Pour faciliter la compréhension du processus, nous formalisons les différentes actions que notre programme devra effectuer dans le cadre d'un use case.



Nom : Visualiser un produit

Déclencheur : Un utilisateur se connecte sur la page de présentation du produit

Acteur principal : Client

Flot de base

1. Le système initialise la fenêtre graphique
2. Le système charge l'objet à présenter depuis la base de donnée
3. Le système affiche l'objet
4. Le client quitte la page
5. Fin du UC

Flots alternatifs

3a. Le client choisit parmi plusieurs options

3a1. Le système applique les nouvelles options

3a2. Le flot continue au point 3

2.3.2 Interfaçage graphique

La technologie X3D consiste à mémoriser un objet dans un fichier au format standardisé, en vue de l'afficher en trois dimensions sur internet. Néanmoins, il s'agit là uniquement d'un format et non d'une méthode d'affichage à proprement parler. Ainsi, un élément logiciel extérieur sera en charge d'afficher cet objet en interprétant ce format dans une page Web. Ces éléments sont des « viewers X3D », programmes additionnels des navigateurs internet standards, qui sont capables d'afficher le contenu des fichiers .x3d. Il en existe de nombreux, libres de droits ou propriétaires, ayant chacun des spécificités qui leur sont propres. Le principal inconvénient de cette solution est le fait que l'utilisateur doit disposer d'un tel programme sur son poste client. Il lui faudra donc, dans bien des cas, le décharger avant de pouvoir profiter d'une visualisation en trois dimensions. Certains sites internet spécialisés répertorient les différents lecteurs disponibles dont voilà la synthèse.

Figure 2

Synthèse des lecteurs X3D

Software	Type	OS			Browser		X3D
		Win	Linux	Mac	IE	Firefox	
Cosmo Player	P	X			X	X	
Octaga Player*	P,S	X	X	X	X	X	X
Cortona3D*	P	X		X	X	X	
Flux Player	P,S,T	X			X	X	X
BS Contact*	P,S	X	X		X	X	X
SwirlX3D	P,S	X			X	X	X
FreeWRL	P,S,T		X	X		X	X
OpenVRML	P,S,T		X	X		X	X
InstantPlayer	S	X	X	X			X
Xj3D	S,T	X	X	X			X
Orbisnap	S	X	X	X			
Demotride	S	X					X
BS Contact J*	A	X	X	X	X	X	X

Types: P-plugin, S-standalone program, T-toolkit, A-applet
(* - Purchasing a license removes the product logo)

Source : cic.nist.gov/vrml/vbdetect.html (Aout 2008)

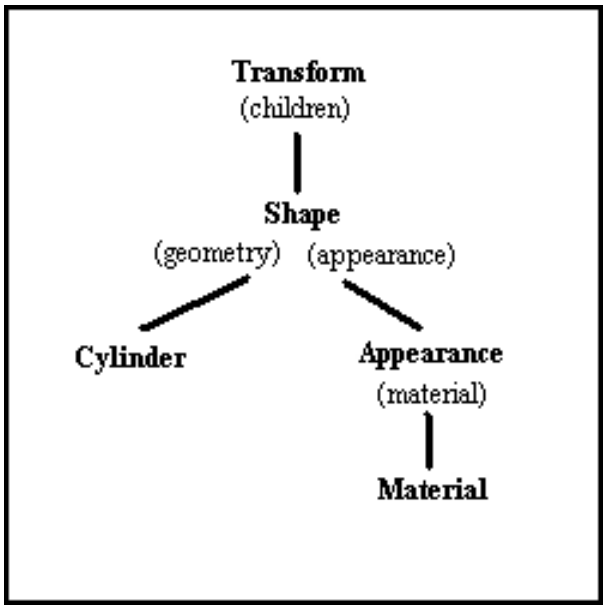
Relevons que malgré la standardisation du format que nous traitons, il existe des variations entre les lecteurs disponibles, notamment au niveau des possibilités d'interaction entre la page HTML et le lecteur X3D. Nous avons donc choisi arbitrairement d'utiliser le plug-in Flux de Media Machines pour notre démonstration. Celui-ci est devenu libre de droits et possède une documentation abondante et facile d'accès.

2.3.3 Le format X3D

Nous allons maintenant étudier le système d'affichage du format X3D. Il s'agit, comme nous l'avons mentionné, du successeur du VRML utilisant un format de fichier qui est basé sur le XML et représente une scène en trois dimensions. La structure ainsi qu'une grande partie des attributs sont similaires à son prédécesseur. Ainsi, de nombreux lecteurs sont à même de lire les deux formats de fichier. Un seul fichier peut inclure des objets, des textures, des lumières ainsi que les animations de ceux-ci.
Une scène est représentée par un arbre constitué de nœuds. Une forme (Shape) formera donc un nœud constitué de deux balises filles dont l'une définira la forme de l'objet et l'autre son apparence. L'apparence est elle-même constituée des divers nœuds représentant les matériaux et textures utilisés. Chacun de ces nœuds possède les propriétés et attributs nécessaires à son affichage.

Figure 3

Organisation des nœuds X3D



Source : www.pinecoast.com(2008)

Dans notre cas, notre objet sera placé dans un nœud de transformations, dans lequel la position et l'orientation pourront être paramétrées. Dans cette structure, une transformation se répercutera sur tous les éléments du sous-arbre formé par ses fils.

La syntaxe d'une scène X3D présentant un cube aura la forme suivante :

```
1   <?xml version="1.0" encoding="UTF-8"?>
2   <!DOCTYPE X3D PUBLIC "ISO//Web3D//DTD X3D 3.0//EN"
3   http://www.web3d.org/specifications/x3d-3.0.dtd">
4   <X3D profile='Immersive' >
5   <head>
6   </head>
7    <Scene>
8      <Transform translation="x y z">
9        <Shape>
10         <Appearance>
11           <Material diffuseColor="r g b"/>
12              <ImageTexture url='"texture.jpg"'/>
13         </Appearance>
14         <Box size='x y z'/>
15       </Shape>
16     </Transform>
17   </Scene>
18  </X3D>
```

1-3) Balise indiquant l'utilisation du format XML, suivie de la spécification X3D.

4) Déclaration du profil. Il indique au lecteur X3D le profil d'utilisation à employer, tel que le type de navigation par exemple.

7) Balise de déclaration de la scène.

8) Balise de transformation permettant le déplacement, l'agrandissement et la rotation d'objets.

9) Balise indiquant la construction d'un objet

10) Zone de déclaration pour le matériel à utiliser sur l'objet

11) Balise indiquant les couleurs et les paramètres de matières

12) Indication de la texture employée pour représenter la surface de l'objet

14) Balise de forme, ici une boite de taille x y z

2.3.4 La technologie AJAX3D

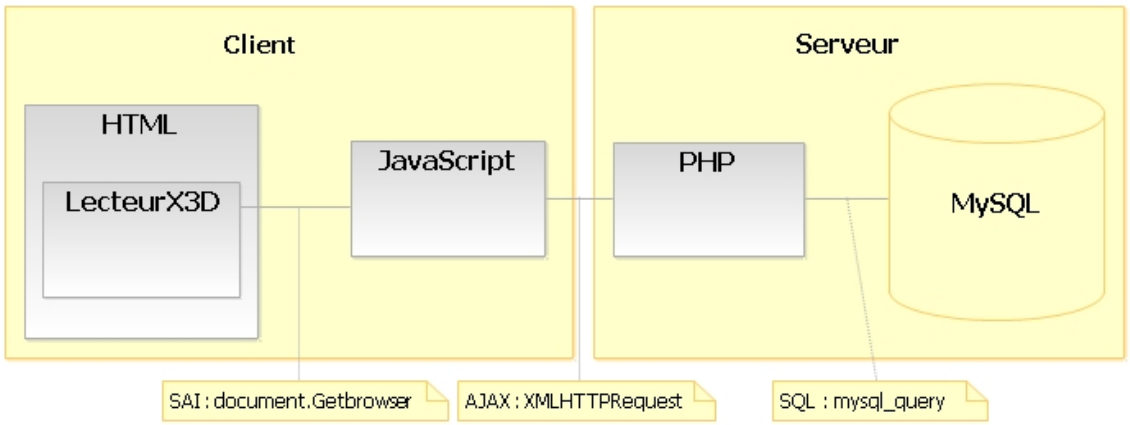
L'un des éléments primordiaux de notre programme et l'un des points clefs de notre cahier des charges consiste en l'interaction dynamique que l'utilisateur peut avoir avec l'affichage. Cela permet aux utilisateurs d'être à même, entre autre, de changer les caractéristiques de l'objet observé en évitant de recharger la page internet comme c'est le cas avec les formulaires PHP notamment.
Pour ce faire, l'emploi d'un langage de script est indispensable. Certaines commandes de script peuvent être directement incluses dans les fichiers X3D. Malgré tout, les limites de leur utilisation sont nombreuses et ne permettent pas l'interaction avec des éléments externes à la scène 3D. Nous allons donc nous pencher sur l'utilisation conjointe du format X3D avec les technologies AJAX (Asynchronous Javascript And XML). Cet assemblage technologique est reconnu sous le nom d'AJAX3D.

Développé pour la communauté des logiciels libre par Media Machines, créateur du lecteur X3D Flux 3D Player, AJAX3D apparaît comme étant l'une des solutions d'interactivité les plus évoluées pour les développements basés sur la technologie X3D. Le principe d'AJAX3D consiste dans l'utilisation du langage JavaScript pour accéder aux éléments d'une scène X3D de manière dynamique. En plus des caractéristiques classiques d'AJAX, de nombreuses possibilités d'interactivité s'offrent à nous. Mentionnons par exemple le chargement dynamique de scènes et d'objets en trois dimensions à partir de bases de données ou encore le contrôle du contenu 3D à partir d'une page Web standard.
Cet accès aux contenus du monde 3D par JavaScript est effectué via la SAI (Scene Access Interface), l'équivalent du DOM (Document Object Model) pour X3D. Il s'agit d'une standardisation des accès au contenu des navigateurs X3D par des langages externes dont le JavaScript.

La mise en place d'une telle infrastructure peut être représentée par la figure ci-dessous.

Figure 4

Présentation d'AJAX3D



Le programme sera divisé en deux parties l'une exécuté par l'ordinateur client et l'autre par le serveur. Le côté client comprenant :

- L'affichage composé de la page internet au standard HTML ainsi que le lecteur X3D.
- La couche programme JavaScript qui fait le lien entre les événements déclenchés par l'utilisateur, la recherche de données via le script PHP et la transmission des instructions d'affichage au Lecteur X3D.

La partie serveur contiendra elle :

- La base de données Mysql contenant les informations nécessaires.
-. Le code PHP en charge d'envoyer les requêtes SQL à la base de données.

2.3.5 Réalisation

Comme présenté ci-dessus, notre programme sera composé de différentes technologies et donc de différents fichiers décrits de manière succincte comme suit :

- **Produit3D.html**
  La page internet contenant l'appel à notre vitrine ainsi que les choix d'options disponibles pour les utilisateurs.

- **FluxLoader.js,**
  Le code d'initialisation du lecteur X3D ; plus de détails concernant ce fichier

seront donnés dans le point détaillant le déploiement.

- **Ajax.js**

  Ce fichier est l'élément central de l'architecture : il contient les fonctions JavaScript en charge du fonctionnement de la vitrine virtuelle, notamment :

  Une fonction déclenchant l'affichage de la scène. Nous utilisons ici la SAI mentionnée précédemment pour obtenir une relation avec l'affichage 3D depuis les fonctions JavaScript. Il s'agit concrètement de la création d'un objet lié au Plug-in X3D. Celui-ci pourra ensuite être référencé via la commande *document.objx3d.getBrowser()* pour les lecteurs X3D compatible Flux. « objx3d » est le nom donné, lors du chargement, à l'objet représentant le lecteur dans la page HTML.

  L'accès au lecteur va nous permettre de charger une scène de format .x3d via la méthode *loadUrlS("Scene.x3d")* de notre objet obtenu précédemment. Il est également possible de créer une nouvelle scène à partir d'une chaine de caractères par l'utilisation de *createX3DFromString(str).*

  Un pointeur sur la scène peut être obtenu par la commande *getExecutionContext()* qui retourne un lien sur l'objet de la scène active. Celle-ci comprend, entre autres, les méthodes *removeRootNode(node)* et *addRootNode(node)* permettant l'ajout et la suppression de nœuds dans notre scène.

  La fonction d'appel au serveur qui permet d'obtenir des informations stockées dans la base de données via AJAX. Pour ce faire, il faut créer un objet de connexion XMLHttpRequest en l'instanciant[3] par la commande *new window.XMLHttpRequest()*. Sa fonction *open(type,URL, boolean)* nous permet ensuite d'ouvrir un flux vers un fichier. Le premier paramètre de cette méthode représente le type d'envoi d'informations («POST » ou « GET »), le second l'adresse de destination et le dernier permet l'activation de connexions asynchrones dont l'utilisation ne sera pas abordée ici.

  Pour envoyer la requête, nous employons la méthode

[3] Désigne la création d'une instance d'un objet, c'est-à-dire l'assignation d'une place mémoire à celui-ci.

send(paramètres_à_envoyer), qui a pour effet d'interpréter le fichier et de mettre à disposition le résultat dans l'attribut *responseText* de l'objet *XMLHttpRequest*.

Nous trouvons également, dans ce même fichier, les fonctions d'événements déclenchées par l'utilisation des éléments de formulaire de la page Html. Par exemple la modification de la matière du produit par le client. Cette opération commence par le chargement, depuis la base de données, des informations nécessaires, et est suivit par l'affichage des changements engendrés, via les fonctions décrites ci-dessus.

- **RequeteServeur.php**

  Ce fichier constitue le serveur faisant le pont entre notre programme JavaScript et une base de données MySQL. Concrètement, il s'agit d'un script PHP effectuant une requête et retournant l'information souhaitée.

```
1  <?php
2  mysql_connect("server", "user", "mdp");
3  mysql_select_db("nom_base_de_donnée");
4  $result = mysql_query("requete_SQL");
5  while($r = mysql_fetch_array($result))
6  echo $r["nom_colonne"];
7  ?>
```

  1-2) Suite à la balise indiquant le début d'un script Php, nous effectuons la connexion à la base de donnée.

  3) Nous sélectionnons la base sur laquelle nous allons travailler.

  4) La requête est exécutée.

  5) Chaque ligne de résultat est mise dans une variable temporaire.

  6) Le résultat est finalement retourné sous la forme d'un affichage.

- **Scene.x3d**

  Représente l'environnement où seront affichés nos produits. Nous retrouvons ici les balises XML d'entête, ainsi que les différents éléments de configuration du type de navigation et de décor.

  Une navigation permettant l'orientation de la scène par l'utilisateur se décrit donc par la balise *<NavigationInfo type=' "EXAMINE" "LOOKAT" "ANY" '/>* et le positionnement de la caméra par *<Viewpoint position='x y z' />*

2.3.6 Déploiement

Il existe deux méthodes d'appel pour une scène X3D dans une page HTML. L'une d'elles utilise la balise <*embed*> que nous détaillons ci-dessous.

> *<embed WIDTH="640" HEIGHT="480" NAME="FLUX" SRC="maScene.x3d" TYPE="model/x3d" DASHBOARD="0" BGCOLOR="0xFFFFFF">*

La deuxième voie pour inclure le fichier est l'implantation orientée objet. La syntaxe en est la suivante :

```
1  <object id="objx3d" class="x3dscene" type="model/x3d+xml"
2  classid="clsid:918B202D-8E8F-4649-A70B-E9B178FEDC58"
3  codebase=http://mediamachines.com/download/SetupFluxPlayer.exe
4  <param name="DASHBOARD" value="0">
5  <param name="BGCOLOR" value="0xaaaaaa">
6  <param name="scr" value="maScene.x3d">
7  <div>Texte affiché en l'absence de lecteur X3D</div>
8  </object>
```

1) Indique le nom de l'objet et son type.
2) Cette ligne est facultative et réservée au navigateur Internet Explorer, il s'agit du numéro assigné au plug-in Flux pour forcer son utilisation.
3) Indique l'emplacement du téléchargement du plugin Flux.
4) Ce paramètre indique que l'on ne veut pas afficher la barre de menu du navigateur.
5) Indique la couleur de fond.
6) Indique le fichier .x3d à charger, cette ligne est facultative si l'on ne veut rien charger à ce moment.

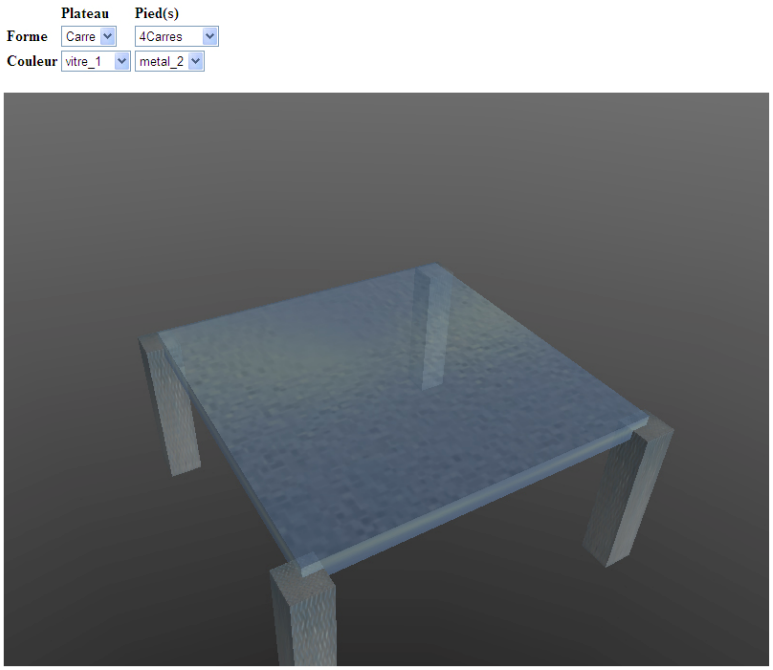
2.3.7 Analyse du développement X3D

La mise en place de notre vitrine virtuelle a mis en évidence les éléments clefs de l'utilisation du format X3D. Comme nous l'avons vu, le principal argument en faveur de ce format était la standardisation de ce type de fichier pour l'échange et la transmission d'objets et de scènes en trois dimensions.
Par cette standardisation, la majorité des logiciels de modélisation graphique du marché, tels que les incontournables Autodesk 3DStudio Max, Maya ou l'open source Blender 3D, disposent d'une fonction ou d'un plug-in d'importation de leurs contenus au format X3D.

Cela se traduit par une portabilité accrue des applications basées sur son utilisation, en raison des possibilités de reprise d'éléments sans nécessité de modification.

Figure 5

Projet de vitrine virtuelle



2.3.7.1 Rapidité de prise en main

Ce langage, basé sur l'imbrication de balises, offre un codage très structuré s'adaptant facilement à un traitement automatique. La déclaration d'objets au format XML permet par conséquent une analyse efficace de tels fichiers, traitement connu sous le nom anglais de « parsing ». La facilité de génération de ceux-ci est, de plus, accrue. L'œil humain s'accommode bien à ce format, reconnaissant sans difficulté les éléments d'une scène pour autant que la complexité de celle-ci soit réduite. En effet, la réalisation d'une scène complexe, c'est à dire représentant plus d'une centaine d'objets, est impensable sans l'aide d'une génération informatisée par le biais d'un modeleur graphique.

La clarté apportée au code par la nomenclature des balises XML permet néanmoins une facilité de prise en main hors du commun, plus aisée que celle nécessaire pour la majorité des langages de programmation ou d'affichage du marché.

Ceci est aidé par le fait que les lecteurs X3D font office de moteur graphique intégré. Une lourde charge de développement spécifique à l'affichage et notamment au

déplacement de la caméra ainsi que la sélection d'objet à la souris est donc évitée.
Cette prise en main rapide rend également cette technologie accessible à la majorité des programmeurs en quelques jours d'auto-formation, permettant le développement de tels projets en l'absence de connaissances préalables du domaine. Il va de soi que le temps de développement en sera réduit d'autant.

2.3.7.2 Couches supplémentaires & inconvénients

Il faut néanmoins admettre que cette technologie est plus appropriée pour des projets de petite taille. Par « petite taille », nous entendons d'abord le nombre d'objets représentés par la ou les scènes à afficher, mais surtout leurs interactions avec leur contexte, comme c'est le cas d'une page HTML. Cette précision pèse son poids et constitue la première note sombre concernant cette technologie.

Lors du chapitre détaillant notre développement, nous avons exposé le système d'interaction basé sur le concept d'Ajax3D qui offre la possibilité d'avoir une interactivité avec les pages internet, voire avec une base de données. Ce système est très efficace à petite échelle, néanmoins le passage entre les différentes couches utilisées ne s'effectue pas sans un certain coût. En effet, l'accès du JavaScript à la scène affichée via la SAI, induit une connexion suivie d'une inévitable recherche de l'objet sur lequel nous voulons effectuer une action quelconque.
Ainsi, chaque opération sera nettement plus coûteuse en termes de performance qu'avec l'utilisation de technologies telles que java ou C++ qui ont un accès direct à l'interface d'affichage.
Dans le même ordre d'idée, lors de l'affichage d'une scène provenant d'une source externe tel que nous l'avons réalisé dans notre mini-projet, une étape indispensable consiste en la génération d'une scène X3D comportant les nouveaux éléments. Une fois cette étape faite, les nouveaux éléments pourront être ajoutés à la scène active. Cette opération constitue évidemment un traitement supplémentaire devant être effectué préalablement à l'affichage.
Finalement, l'utilisation d'Ajax pour le chargement de données, impose dans le cas d'un accès à une base de données, l'interprétation d'un fichier au format PHP en charge d'effectuer les requêtes SQL nécessaires.

Nous avons là trois exemples d'étapes intermédiaires dont l'utilisation est forcément coûteuse. Évidement, ce coût supplémentaire est imperceptible pour toute scène

simple, comme dans le cas de la vitrine virtuelle de notre étude. Mais nous pouvons affirmer que le système mis en place pour l'utilisation de ce format n'est à ce jour pas adapté pour des créations lourdes de type vidéo ludique par exemple. Ceci en raison des éléments ralentisseurs cités précédemment mais également du fait que les technologies Web ne sont pas spécialement adaptées à ce type de développement. Ceci pourrai être amené à changer au vue du développement objet du langage PHP notamment.

2.3.7.3 Disparité des lecteurs X3D

Pour conclure notre analyse, il nous faut parler des inconvénients relatifs à la disparité des lecteurs X3D. En effet ces plug-ins, bien que tous capables d'afficher les objets standardisés de base, ne prennent en revanche pas tous en compte la gestion d'effets avancés. Ceux-ci sont généralement développés et supportés par un lecteur X3D spécifique. L'interprétation d'une balise inconnue entraîne généralement le non-affichage de la scène dans son intégralité, voire un plantage global du lecteur X3D. La robustesse du logiciel s'en trouve par conséquent gravement diminuée.
De plus, la connexion aux plug-ins X3D par l'appel aux SAI, bien que faisant eux aussi l'objet d'une standardisation, utilise dans bien des cas des méthodes d'accès différentes. Dans certains cas, des méthodes de signatures identiques produisent des résultats différents suivant le lecteur utilisé, rendant ainsi les lecteurs concurrents incompatibles.
Du fait de ces deux problèmes de compatibilité inter browser X3D, l'utilisateur se verra forcé d'utiliser le plug-in pour lequel le logiciel à été développé, ce qui entraînera la présence d'une multitude de lecteurs différents sur le poste du client. Cela est regrettable puisqu'un format conçu sous forme standardisée ayant pour but la compatibilité d'un lecteur à un autre ne l'est pas dès lors qu'on s'attaque réellement à des réalisations concrètes.

2.3.7.4 Solutions libres contre propriétaires

Dans bien des domaines de l'informatique d'aujourd'hui, les deux modes de pensée que sont l'open source ou les solutions propriétaires se trouvent face à face lors d'un choix de technologie. Le petit monde du X3D n'échappe pas à la règle et illustre parfaitement cette situation.

Sans entrer dans le grand débat qui anime cette question, rappelons dans les grandes lignes les principes du logiciel libre. La philosophie open source soutient l'idée d'une collaboration ouverte entre développeurs pour la conception de logiciels libres de droits, pouvant être utilisés gratuitement par tout un chacun. Dans cette optique d'ouverture, toutes les composantes d'un programme, telles que le code source, doivent être mises à disposition de la communauté. Les logiciels open-source utilisent pour ce faire différents types de licences. Nous allons brièvement détailler la plus utilisée d'entre elles à ce jour, la licence GPL GNU. L'utilisation de la majorité des lecteurs X3D Open source, tel que Flux (utilisé dans notre exemple) ou Xj3D est, en effet, réglementée par GPL GNU.

La licence GPL pour Général Public Licence, définie par la société à but non-lucratif Free Software Foundation, consiste en un contrat d'utilisation extrêmement peu restrictif. En effet, un logiciel sous cette dénomination peut être modifié et réutilisé à des fin commerciales ou non. Parmi les quelques restrictions de la license GPL, relevons la nécessité d'enregistrer tous les sous-produits d'un logiciel GPL sous cette même licence, ainsi que l'obligation de mettre à disposition tous les codes sources.

L'un des avantages des solutions utilisant une licence open source réside dans la communauté généralement dynamique autour du produit. Cela constitue une source d'informations importante comprenant exemples et tutoriaux disponibles gratuitement sur internet. Au contraire, certain éditeurs de logiciels réservent leur documentation aux personnes disposant de la licence de leur produit.

Les prix du marché sont quant à eux très variables en fonction des options disponibles. La plage de prix pour l'obtention d'une licence pour un lecteur X3D varie entre 50€ et 300€ selon les éditeurs. Le coût d'un SDK (Software Developpement Kit) peut s'élever à 2'000€ et une installation comprenant serveur et base de données dépasse généralement les 20'000€.

2.3.7.5 Difficultés rencontrées lors du développement X3D

Durant ce développement, la prise en main de la technologie fut en grande partie rendue aisée par la communauté d'Ajax3D qui met à disposition un tutoriel clair et suffisamment complet pour un développement de base. Nous nous sommes néanmoins heurtés à certains problèmes de rigidité du format X3D ; en effet, il n'est par exemple pas possible de déclarer plusieurs formes pour une seule déclaration d'apparence. Il est donc obligatoire de déclarer un champ de texture pour chaque

forme, même si celui-ci est identique pour de nombreuses entités.
Les problèmes de compatibilité relevés précédemment ont également fortement ralenti la mise en place de ce projet, les informations disponibles sur le sujet ayant tendance à être occultées pour des raisons marketing évidentes. Ainsi, les différents problèmes de compatibilité ne se sont révélés que par la pratique et la recherche d'informations pour les résoudre fut des plus laborieuses.

2.4 Second Développement, le jeu d'échecs

2.4.1 Le mandat

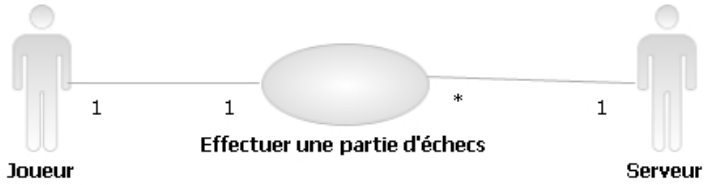
L'objectif est de proposer, par le biais du développement d'un jeu d'échecs en trois dimensions, un échantillon des possibilités que nous offre Java et ses librairies. Nous allons tout d'abord fixer les limites de notre développement.

Bien qu'il soit usuel dans ce type de d'application ludique de pouvoir jouer seul contre un ordinateur, une telle réalisation nécessite la mise en place d'une Intelligence artificielle de base dont développement serai sans lien aucun avec le champ de cette étude. Nous avons donc décidé d'orienter notre jeu vers les parties opposant deux adversaires humains.

Notre cahier des charges comprend donc :
1) Un programme utilisable et accessible via Internet par la majorité des ordinateurs.
2) Un jeu d'échecs respectant et implémentant les règles de celui-ci.
3) La possibilité à l'utilisateur d'effectuer une partie contre un autre joueur.
4) Offrir des effets graphiques de qualité.

2.4.2 Cas d'utilisation

Nous pouvons établir sous la forme d'un « use case » les différentes actions que notre programme devra effectuer.



Nom : Effectuer une partie d'échecs

Déclencheur : Un utilisateur entre dans le programme

Acteur principale : Joueur

Acteur secondaire : Serveur

Flot de base :

1. Le système se connecte au serveur
2. Le serveur crée une nouvelle partie avec le joueur présent dans la file d'attente
3. Le système affiche l'échiquier
4. Le joueur actif sélectionne une pièce lui appartenant
5. Le système affiche la pièce active en surbrillance ainsi que les déplacements possibles
6. Le joueur actif sélectionne la case de destination voulue
7. Le système affiche la case en surbrillance
8. Le joueur valide son coup
9. Le système déplace la pièce sélectionnée dans la case cible et envoie le coup au serveur
10. Le serveur transmet le coup effectué
11. Le système active l'autre joueur
12. Le flot continue au point 4

Flots alternatifs

2a La file d'attente est vide

2a1 Le serveur met le joueur connecté en attente

2a2 Un autre joueur se connecte

2a3 Le serveur crée une nouvelle partie avec le nouveau joueur

7a La case cible est occupée

7a1 Le système déplace la pièce visée dans la zone des pièces mortes

7a1a La pièce morte est une pièce « roi »

7a1a1 Le système envoie un message de fin de partie au serveur

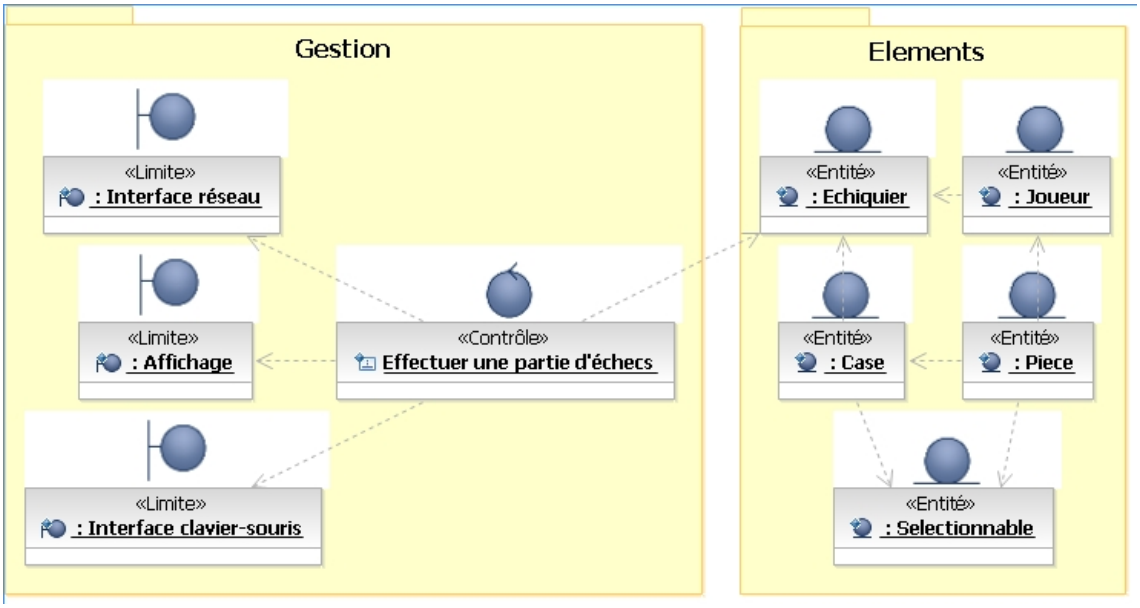
7a1a2 Fin du UC

2.4.3 Structure logique

Nous pouvons identifier ici les éléments de notre jeu d'échecs à partir de notre use-case.

Figure 6

Analyse objet du jeu d'échecs



Les cases ainsi que les pièces pouvant être sélectionnées de manière similaire, nous pouvons les factoriser en une super-classe commune du nom de « Selectionnable ».

D'un point de vue structurel, nous avons identifié trois éléments d'interface. L'élément en charge de l'affichage, l'interface réseau pour la communication avec le serveur et l'interface d'entrée-sortie chargée de récupérer les informations transmises par l'utilisateur via le clavier et la souris.

2.4.4 Interfaçage graphique

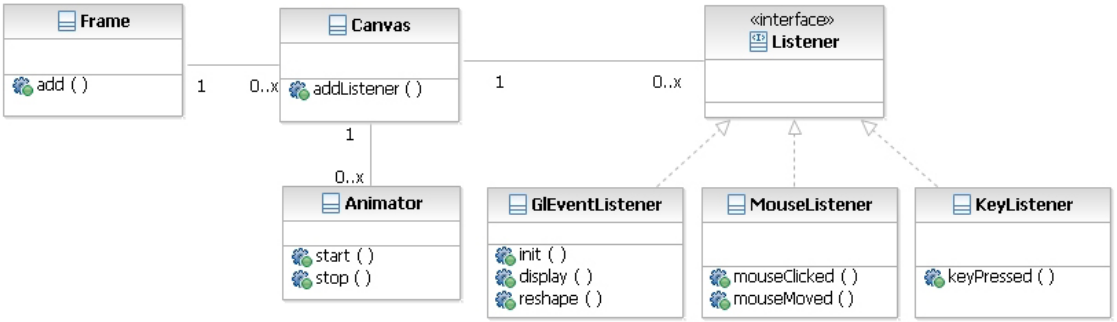
La partie clef de notre jeu d'échecs est bien sûr son système d'affichage.
En utilisant le langage Java, nous avons accès à deux librairies graphiques permettant un affichage en trois dimensions : JOGL et JAVA3D

Jogl étant la transposition de l'API OpenGL pour Java, cette librairie nous donne accès à un grand nombre d'options graphiques, ceci à condition de maîtriser son environnement.
JAVA3D est une API de plus haut niveau, c'est à dire que nombre de modules sont disponibles. JAVA3D ne permet par contre pas le développement des effets élaborés auxquels nous aimerions avoir accès. Cette technologie semble également ne plus être supportée par ses créateurs, ce qui limite considérablement la durée de vie d'une application basée sur celle-ci.

Jogl sera donc notre librairie graphique. Par conséquent, nous commençons par brièvement étudier le fonctionnement d'OpenGL dont JOGL est tiré.

Figure 8

Classes JOGL



Au lancement d'une application, une vue est créée par l'intermédiaire d'une Frame. Celle-ci sert de cadre pour l'affichage. Un « GLCanvas » chargé de l'affichage dans ce « cadre » lui est assigné. Nous allons y joindre les différentes interfaces d'entrée-sortie que sont : l'affichage « GLEventListener », les événements provenant du clavier « KeyListener », ainsi que les événements provenant de la souris « MouseListener. »

Le dernier élément de base indispensable est l'entité contrôlant le fonctionnement du canevas : un animateur « Animator ». Celui-ci permettra de lancer ou d'arrêter les modules d'écoute du canevas par l'intermédiaire de ses méthodes « start() » et « stop() ». Concrètement, notre animateur va exécuter en boucle les modules liés au canevas. A chaque boucle, une image statique est affichée par la méthode display(), correspondant au taux rafraichissement de l'image.

Nous allons maintenant détailler notre « GLEventListener » puisque que ce module est

chargé des éléments à afficher dans notre Frame. Il sera représenté par une classe implémentant l'interface « GLEventListener » et devra contenir les différentes méthodes[4] indispensables qui sont :

- **init(GLAutoDrawable pileOpenGL)** qui sera exécutée à l'appel de la méthode « start » de l'animateur. Elle contient toutes les spécifications graphiques initiales, tels que les paramètres d'affichage global, les lumières, les matériaux par défaut, etc.
- **display(GLAutoDrawable pileOpenGL)** qui sera exécutée en boucle et affichera notre scène. Toutes les instructions d'affichage OpenGL seront appelées à partir de cette méthode. La cadence d'exécution de celle-ci donnera le FPS (Frame Per Second), le nombre d'images par seconde de notre moteur graphique.
- **reshape(GLAutoDrawable pileOpenGL)** sera exécuté en cas de re-dimensionnement de la fenêtre graphique ; il faudra alors réinitialiser les paramètres de vue et de perspective en fonction de la nouvelle taille de fenêtre. Le paramètre de type « GLAutoDrawable » constitue la pile de matrices qui sera envoyée par la librairie Jogl à la carte graphique pour affichage. Tous les dessins et transformations se feront donc via ce paramètre.

Concernant les paramètres et instructions OpenGL, je vous invite à lire les ouvrages et sites internet dédiés à ce langage qui sont nombreux et généralement bien construits.

2.4.5 Déploiement

Notre application est terminée et opérationnelle en local en tant que programme Java. La phase de déploiement de programmes Java sur internet peut être relativement complexe. Toutefois, les possibilités qu'offre la librairie JOGL, constituée des deux packages : jogl.jar et gluegen-rt.jar, sont multiples.

L'API JOGL utilise un rendu matériel effectué par la carte graphique à l'opposé des API logiciels (par exemple le format d'Adobe Flash). Celles-ci faisant appel uniquement au processeur, ce qui peut provoquer une surcharge du CPU lors du traitement de scènes

[4] Désigne les instructions associées à un objet.

complexes, entraînant de ce fait d'importants ralentissements visuels. En revanche,, pour accéder aux ressources matérielles de la machine client, JOGL utilise des fichiers systèmes spécifiques à l'environnement d'exploitation présent sur la machine utilisée par le client.
Pour gérer les nombreux environnements disponibles (MacOs, Windows, Unix, Linux,..) un script capable de choisir les librairies adéquates devra être mis en place.

Heureusement la communauté de JOGL met à disposition des solutions pour effectuer ce travail. En effet, deux structures de déploiement Web sont mises en avant par SUN. Celles-ci sont la méthode JavaWebStart et les Applets Java. Cette dernière permettant l'exécution de classes Java par un navigateur Web et, dans notre cas spécifique, notre jeu d'échecs et son interface graphique.

JavaWebStart consiste en un format de fichier .jnlp (Java Network launching protocol) utilisant la syntaxe XML capable d'exécuter des classes Java et pouvant être appelé par le biais d'une balise HTML classique du type

lance mon programme Java.

Relevons que l'exécution de ce format de fichier requiert un serveur Web reconnaissant le type MIME « application/x-java-jnlp-file » ce qui peut être contraignant dans le cas où votre hébergeur Internet ne gèrerait pas celui-ci.

Voici la syntaxe du fichier monApplication.jnlp capable de lancer notre application JOGL.

<?xml version="1.0" encoding="utf-8"?>

Indique que ce fichier utilise le format XML.

<jnlp spec="1.0+" codebase="http://url_de_la_page.net"
href="monApplication.jnlp">

Indique l'emplacement URL et le nom du fichier JNLP. Ces informations permettront au navigateur de rechercher la dernière mise à jour de celui-ci dans le cas où ce fichier est déjà dans le cache.

<information>
<title>monApplicationWeb</title>

Indique le nom de l'application.

<vendor>Sauvain Renaud</vendor>

Indique le nom du distributeur.

<homepage href="http://renaud.sauvainr.ch" />

(Facultatif) Indique l'adresse du site Internet

<description>Description de l'application.</description>

Contient une brève description de l'application.

<offline-allowed />
Indique si l'application peut être lancée hors-connection.

</information>
<resources>
<j2se version="1.4+" />
Indique la version de Java utilisée.

<jar href="monApplication.jar" />
Indique le fichier Java à charger.

<jar href="http://download.java.net/media/jogl/builds/archive/jsr-231-webstart-current/jogl.jar" />
<jar href="http://download.java.net/media/gluegen/webstart/gluegen-rt.jar" />
Indique les librairies Jogl à charger.

<extension name="JOGL"
href="http://download.java.net/media/jogl/builds/archive/jsr-231-webstart-current/jogl.jnlp"/>
Indique le lancement associé d'une autre application JavaWebStart. Celle-ci chargeant les fichiers systèmes.

</resources>
<security>
<all-permissions />
</security>
Définit les permissions accordées au programme Java. Celui-ci doit être néanmoins signé et validé pour que le programme puisse se lancer. Nous parlerons de la signature de programmes Java plus tard.

<application-desc main-class="ClassePrincipale" >
<argument>arg1</argument>
</application-desc>
Indique la classe principale à utiliser. Ainsi que les éventuels arguments à transmettre à celle-ci.

</jnlp>

Pour des raisons de sécurité, certaines limitations ont été jointes par SUN aux Applets Java ainsi qu'au programme JavaWebStart. Ceux-ci s'exécutent dans un environnement limité généralement appelé « sandbox » qui restreint les accès aux ressources de la machine client telles que le système de fichier de l'ordinateur, ou encore l'exécution de programmes installés sur la machine locale.
Les librairies utilisant dynamiquement les ressources du système ne sont également pas autorisées, or c'est bien évidemment le cas des fichiers de la libraire JOGL utilisé dans notre exemple.

Il nous faut donc contourner ce problème et, pour ce faire, deux méthodes existent.
La première consiste à ajouter une signature cryptographique au package de notre programme. Ainsi, lors du lancement de celui-ci, l'internaute devra autoriser

explicitement le lancement de celle-ci via une fenêtre pop-up affichant le certificat du programme.

Il faut préciser que la signature doit être validée par une société de certification. Cette opération est malheureusement payante et par conséquent, certaines applications Web génèrent un message de confirmation signalant qu'une signature digitale est invalide. Néanmoins, il est extrêmement déconseillé d'accepter un certificat non certifié émis par un éditeur inconnu pour des raisons de sécurité évidentes.

Figure 8

Certificat Java non-validé

Pour éviter ce problème, nous allons utiliser la librairie « JNLPAppletLauncher » éditée et signée par SUN. Cette librairie permet de charger des packages Java tels que Jogl en utilisant le certificat de SUN.

Figure 9

Certificat validé

Dans le cas ou l'hébergeur ne supporte pas le format Jnlp ou pour toutes autres raisons, une Applet peut également faire le lien entre notre page Web et notre

programme Java.

Pour ce faire, une classe ou un package étendant l'interface Applet (incluse dans les librairies Java de base) pourra être appelée via une balise HTML.

La syntaxe d'une telle balise est la suivante :

```
<applet code="org.jdesktop.applet.util.JNLPAppletLauncher"
  archive="http://download.java.net/media/applet-launcher/applet-launcher.jar,
  http://download.java.net/media/jogl/builds/archive/jsr-231-webstart-
current/jogl.jar,
  http://download.java.net/media/gluegen/webstart/gluegen-rt.jar,
  ./monProgramme.jar">
```

Celle-ci indique la classe à exécuter ainsi que les différents packages à charger pour notre Applet, c'est-à-dire dans notre cas, le fichier du JNLPAppletLauncher, les packages de l'API Jogl et notre programme lui-même. Notons que nous utilisons directement les librairies disponibles sur le site de SUN. Ceci qui évite de devoir stocker les librairies sur son serveur et donne accès immédiat aux dernières versions compatibles.

<param name="codebase_lookup" value="false">
Ce paramètre permet l'utilisation de fichier plat sur le serveur. Sa désactivation accélère le chargement de notre application.

<param name="subapplet.classname" value="MaClassePrincipale">
Le nom de la classe principale de notre programme.

<param name="subapplet.displayname" value="mon programme">
Le nom de notre Applet.

<param name="noddraw.check" value="true">
L'API OpenGL rencontrant des problèmes d'incompatibilité de driver avec DirectDraw il est recommandé de le désactiver par ce paramètre pour notre Applet. Une fenêtre demandera à l'utilisateur de confirmer la désactivation de DirectDraw.

<param name="noddraw.check.silent" value="true">
Ce paramètre permet de désactiver DirectDraw sans l'accord explicite de l'utilisateur.

<param name="progressbar" value="true">
Montre l'état du chargement des librairies.

<param name="jnlpNumExtensions" value="1">
Nécessaire au chargement de fichier de type .jnlp indique le nombre de fichiers à charger.

```
<param name="jnlpExtension1"
      value="http://download.java.net/media/jogl/builds/archive/jsr-231-webstart-
current/jogl.jnlp">
```

Indique l'emplacement du fichier effectuant le chargement de nos fichiers systèmes.

2.4.6 Développements

Nous allons ici détailler la mise en place de certains éléments importants de notre programme, éléments dont la réalisation technique nécessite un approfondissement.

2.4.6.1 Mode multi-joueurs

Comme indiqué dans notre cahier des charges, notre partie d'échecs se doit d'inclure la possibilité d'effectuer des parties entre deux joueurs connectés via internet.
Il s'agit donc de faire dialoguer les programmes des utilisateurs appelés « clients » via un programme unique en charge de la communication nommé « serveur ». Pour ce faire une architecture client-serveur doit être mise en place.

Il existe différentes méthodes pour une telle réalisation. Celle appliquée dans cet exemple consiste en un programme Java installé sur un serveur internet. Celui-ci sera chargé de la création des parties lorsque des joueurs se connectent. Par la suite, il fera office de pont entre les deux clients pour la diffusion des coups effectués.

La réalisation d'un tel programme nécessite l'utilisation des deux concepts primordiaux que sont les « connexions socket » et « Threads ».

2.4.6.1.1 Les Threads

Les « Threads », littéralement fils en français, sont des portions de programme indépendants dans le fait qu'ils travaillent de manière asynchrone avec le reste du logiciel. Ainsi plusieurs tâches peuvent être exécutées de manière simultanée et parallèle. Cette conception est indispensable dans notre cas, puisque le serveur sera en charge de plusieurs traitements. Il s'agit d'être à la fois attentif à la connexion de nouveaux clients, à l'écoute d'informations provenant de clients déjà connectés telles que les déplacements de pièces par les joueurs, mais également de traiter les informations reçues, ainsi que de gérer la création des nouvelles parties.
Ces processus autonomes seront déclarés comme classe étendant la super-classe

« java.lang.Thread » et en implantant l'interface « runnable ». Une fois ces classes instanciées, l'appel à la méthode héritée « start() » déclenchera l'exécution de la méthode « run() » qui lancera l'exécution du processus de manière Asynchrone.

2.4.6.1.2 Les connexions socket

2.4.6.1.2.1 Théorie

Les sockets internet constituent l'une des méthodes les plus répandues en termes de connexion réseau. Le terme socket définit une adresse combinant l'adresse IP représentant une machine, unique à un numéro de port. Un port est une adresse codée sur 16 bits, dont le but est de représenter une application sur un réseau TCP/IP. Ainsi, il sera possible à l'aide du nom de la machine et du numéro de socket d'un programme de dialoguer avec celui-ci depuis une machine quelconque connectée à un réseau commun tel qu'internet. Une telle connexion ouvre un dialogue bidirectionnel permettant l'envoi et la réception simultanés de flux de données. En termes techniques, un socket est une interface déployée par le système d'exploitation pour permettre à un programme d'utiliser le protocole de communication TCP/IP.

La librairie « java.net » offre les outils pour une telle implémentation. La classe « Socket » constitue un objet représentant un canal par lequel les informations peuvent transiter.
La création d'un tel canal s'effectue par l'ouverture du socket obtenue par la commande *new ServerSocket(numerot_du_port)*, qui utilise en paramètre le numéro de port assigné à notre application. Le choix de ce numéro, bien que large, doit être malgré tout étudié. En effet l'utilisation des 65536 ports possibles sur un code 16 bits, a été standardisée par l'IANA (Internet Assigned Numbers Authority). Il existe trois plages d'adresse. Les ports de 0 à 1023 sont réservés aux processus système et notamment aux différents protocoles de haut niveau comme le FTP, HTTP, Telnet, etc. Les ports compris entre 1024 et 49151 sont appelés ports enregistrés. Dans cette plage le fournisseur d'applicatifs enregistre les ports utilisés, notamment pour éviter des éventuels problèmes de collision. Les ports restants sont réservés quant à eux à une utilisation privée ou dynamique.

L'attente de la connexion du client se fera donc par l'utilisation d'un Thread spécifique qui sera informé par l'arrivée d'un dialogue par la commande *serverSocket.accept()*

retournant l'adresse de socket de la machine connectée.

La connexion étant effectuée, il est maintenant possible de dialoguer entre nos deux applications. Ceci réalisé par la mise en place de deux canaux, l'un d'entrée et l'autre de sortie.
Nous allons créer un canal d'écoute permettant de réceptionner les informations. D'un point de vue technique, nous utilisons la méthode *getInputStream()* fournissant un flux d'informations binaire provenant du port. Ce flux doit être transformé par *inputStreamReader(),* faisant le pont entre un flux binaire et un flux de caractère. Finalement, le flux sera stocké dans un buffer nous permettant la lecture des informations reçues.

Nous avons donc en une ligne notre connexion entrante.

```
canal_d_entree = new BufferedReader(new
inputStreamReader(socket_client.getInputStream()))
```

La récupération des données se faisant via la méthode *read()* ou *readLine()* du *BufferReader.* Il est possible de traiter des commandes provenant de la console java de manière similaire, le flux de saisie de la console s'obtenant par l'attribut System.in.

Il nous reste à créer un canal de sortie pour l'envoi de données. Le flux de sortie est obtenu par la méthode *getOutputStream()* qui, par l'intermédiaire d'un objet *PrintWriter* enverra un flux de caractères au destinataire.

```
canal_de_sortie = new PrintWriter(socket_client.getOutputStream()))
```

L'écriture dans le flux est effectué par *print(variable_a_envoyer)* et *println(variable_a_envoyer)* du canal de sortie. Finalement, le flux sera envoyé puis nettoyé par l'utilisation de la méthode *flush()*.

2.4.6.1.2.2 Réalisation

Nous avons maintenant en mains tous les concepts nécessaires à la création de notre serveur. Ce programme sera conçu en plusieurs pièces. Il va de soit que les deux éléments que nous traitons ici sont les joueurs et les parties en cours. Nous allons donc utiliser une classe « Partie » représentant les parties en cours, celles-ci contenant

les deux protagonistes.
Ces derniers, seront représentés par une classe « SessionClient » contenant le matériel nécessaire à la connexion entre le serveur et leur ordinateur. Cette classe inclura notamment les éléments étudiés au point précédent qui sont le socket du client, le canal d'entrée et celui de sortie. Cette classe devant bien entendu contenir un Thread lui permettant de gérer les dialogues avec les différents clients de manière continue.

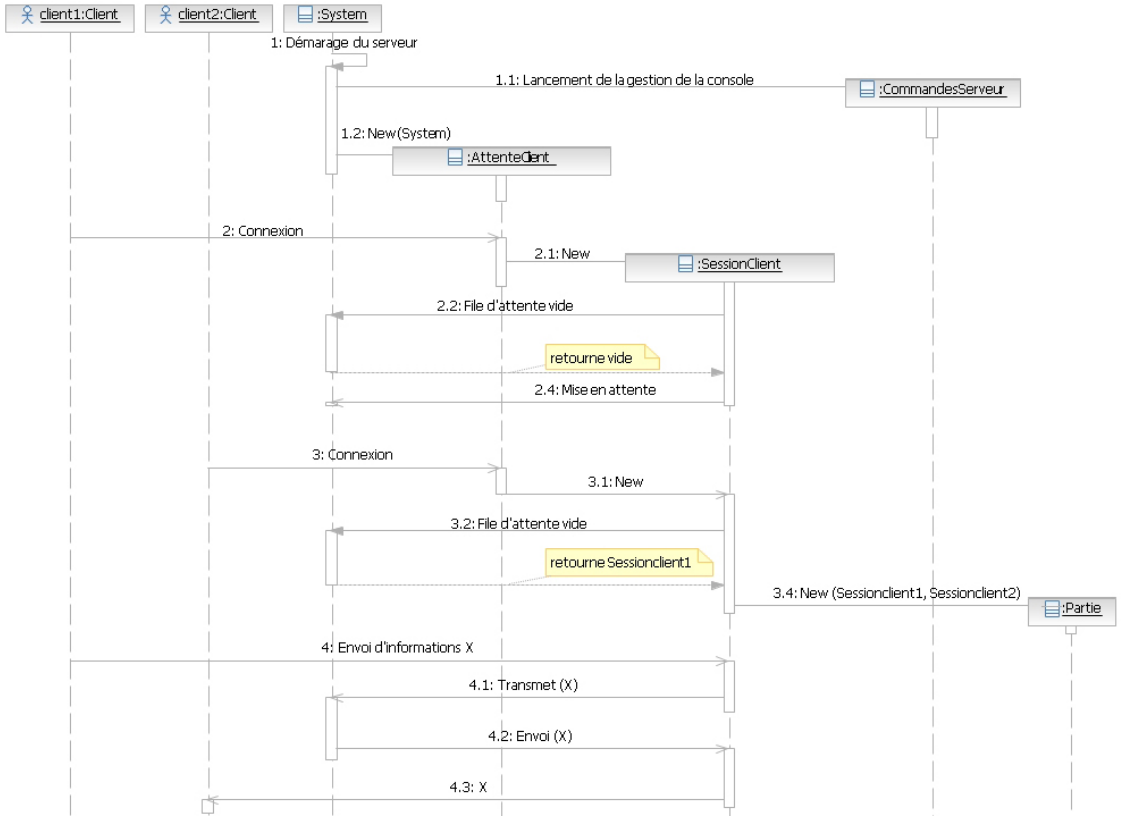
Un autre élément de notre structure sera le processus d'attente de joueur qui sera représenté par une classe « AttenteClient », travaillant également de manière asynchrone avec le reste de l'application.

Une dernière classe, « CommandesServeur » sera en charge de la récupération des éléments saisis via la console Java. Cela permet à un administrateur de gérer le serveur.

Tous les éléments sont maintenant réunis. Nous pouvons donc représenter de manière concrète et compréhensible le fonctionnement de notre processus par le schéma ci-dessous présentant le déroulement séquentiel de l'utilisation du serveur.

Figure 10

Interaction séquentielle client-serveur



La partie répondante du client, capable de dialoguer avec notre serveur va être développée d'une manière similaire. Pour se connecter à un serveur nous allons instancier un port avec l'adresse url du serveur et le numéro de port de notre application par la commande « new Socket(Url, no_port) ». Notons que cette instanciation engendre l'assignation dynamique d'un numéro de port à notre client. Ce socket créé, nous pouvons utiliser les canaux d'entrée-sortie de manière identique à celle utilisée pour notre serveur

A ce stade nous avons un serveur de jeux d'échecs, certes simple, mais fonctionnel.

2.4.6.2 Le Bump mapping

2.4.6.2.1 Théorie

Avec les évolutions technologiques concernant l'affichage 3D et conjointement à l'apparition de cartes graphiques de plus en plus performantes, les développeurs ont peu à peu mis en place différentes techniques d'effet visuel, dans le but de rendre de plus en plus réaliste les scènes en trois dimensions. Ainsi, les simples couleurs unies

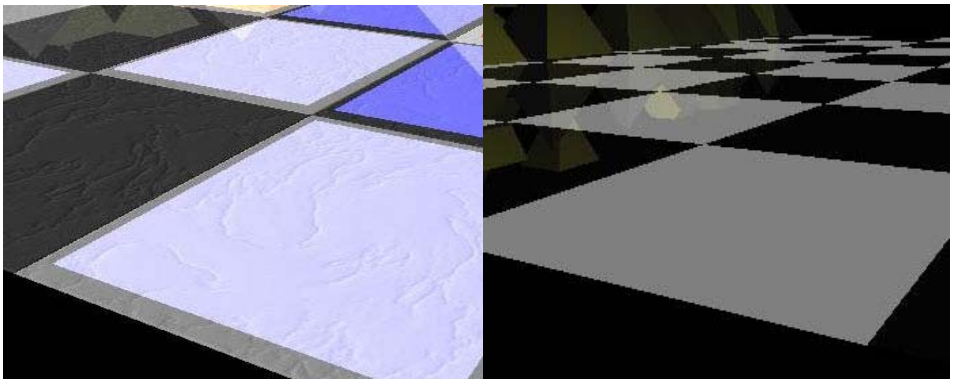
couvrant les objets laissèrent leur place à des textures châtoyantes, puis suivit l'ajout d'effet tel que la transparence et ainsi de suite. Cette évolution a grandement été aidée par les besoins de certains domaines de rendu réaliste, tel que pour l'industrie du cinéma, mais également par la concurrence sans merci que se livre les fournisseurs de moteurs graphiques pour rendre les contenus visuels, notamment du monde vidéo ludique, plus attrayant. L'exemple le plus fragrant étant celui de la lutte technologique menée entre Id Software (Doom, Quake,..) et Epic Games (Unreal).

Le défi constant de ce domaine se situe dans l'affichage le plus somptueux possible tout en conservant la fluidité graphique nécessaire à un confort visuel. Il va de soit que la rapidité d'affichage, le fameux FPS est inversement proportionnel à la complexité de la scène à afficher. Concrètement un compromis est indispensable entre d'un coté un nombre élevé de polygones dont découle une grande qualité de rendu, et d'un autre une rapidité d'affichage élevée. Ainsi furent développées des techniques graphiques apportant un attrait visuel important sans augmenter pour ce faire la complexité des objets.

Nous allons développer l'une d'entre elle, le bump mapping pour illustrer les possibilités avancées offertes par la technologie JOGL. Celle-ci consiste à utiliser non pas une texture définissant la couleur de l'objet, mais plusieures. En effet, une seconde texture représentant le relief de l'objet peut-être utilisée pour donner à une surface plane un effet de profondeur.

Figure 11

Bump Mapping : Avec et sans

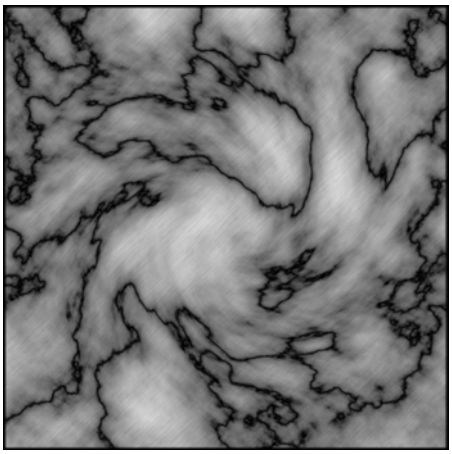


Le bump mapping consistait à l'origine dans la déformation de la forme d'un objet avec l'aide d'une texture de profondeur. Ceci a pour effet de réellement créer un relief. Mais cette déformation entraine la création d'un nombre élevé de polygones rendant ainsi

chaque objet extrêmement complexe à afficher. Partant d'un principe similaire, le Normal Bump Mapping, utilise également une texture de profondeur, mais cette fois chargée de déformer non pas l'objet lui-même, mais les normales associée à celui-ci. Les normales, vecteurs associés à chaque face d'objet en trois dimensions, sont utilisées par l'affichage pour calculer les zones d'ombre et de lumière. Par conséquent, en spécifiant différentes normales pour une même surface nous obtenons visuellement, par le jeu de lumière, un effet de profondeur très réaliste. Néanmoins la charge de calcul nécessaire à la modification des normales constitue parfois une surcharge pour les applications. C'est pour cette raison qu'une troisième solution, appelé Emboss Bump Mapping, naquis. Celle-ci consiste dans la superposition d'une texture de lumière et d'une texture d'ombre légèrement décalée par rapport à la texture de base en fonction de la direction de la source lumineuse.

Figure 12

Texture de profondeur



Le gros gain apporté par cette opération réside dans l'imprécision de notre relief puisque celui-ci sera le même pour toute la surface et non recalculé pour chaque pixel comme c'est le cas du normal bump mapping. Cette technique a malgré tout le double mérite de demander un minimum de calculs lors du rendu de la scène et d'avoir une mise en place plus aisée que celle des deux premières solutions.

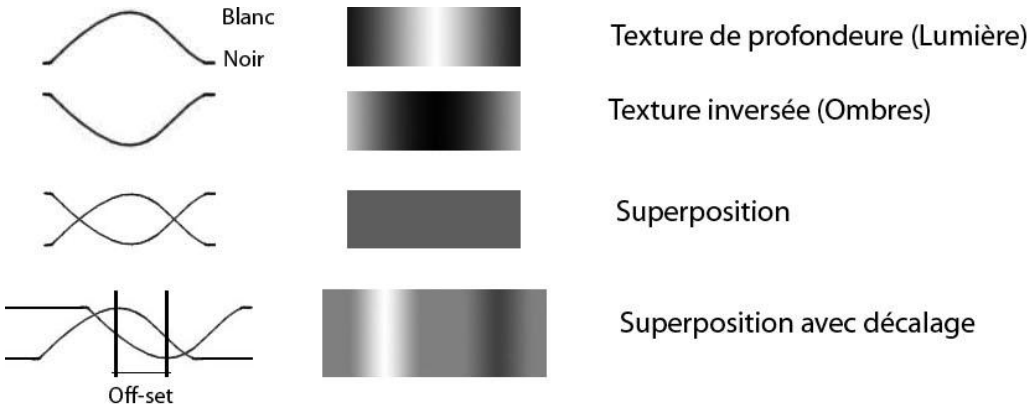
2.4.6.2.2 Réalisation

Notre logiciel implémente donc deux textures supplémentaires faisant office de calque de luminosité pour donner un effet de relief à notre damier. Concrètement, une texture monochrome représentant le relief de la surface est réalisée et, lors de son

chargement par le logiciel, nous générerons la texture d'ombre en inversant simplement les couleurs de la première.

Figure 13

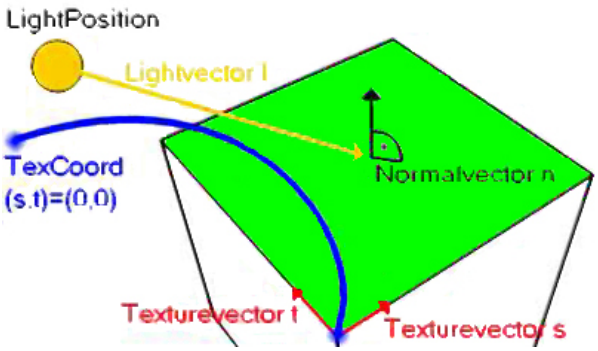
Emboss Bump Mapping



Il nous reste à décaler ces deux textures en fonction de l'orientation de la source lumineuse par rapport à la surface. Ce décalage appelé « offset » s'obtient par la multiplication du vecteur de lumière par la normale. Cette formule est bien évidemment simplifiée et engendre une légère imprécision dans le placement des textures, mais le résultat reste satisfaisant.

Figure 14

Schéma des vecteurs



Source : nehe.gamedev.net (Aout 2008)

Possédant ce décalage nous pouvons à présent afficher nos textures. OpenGL permet l'assignation de plusieurs textures par face. Ce principe est appelé mutli-texturing et il nous permet de superposer les trois textures dont nous avons précédemment parlé. La

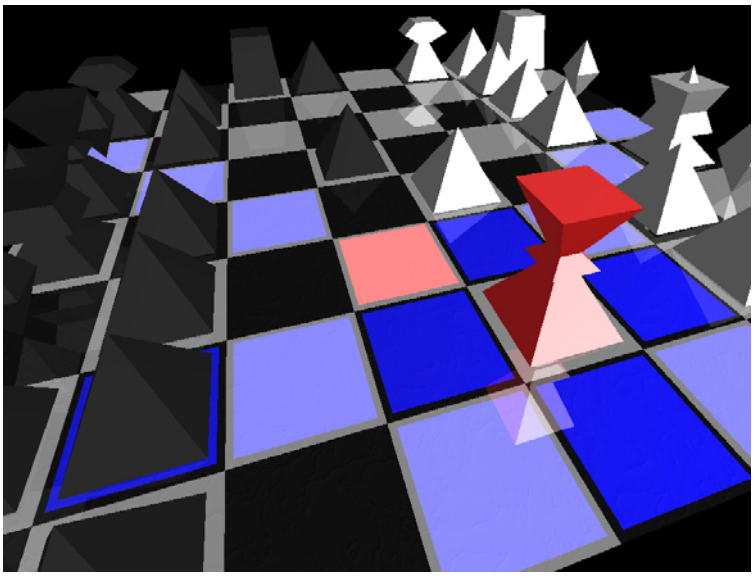
mise en place de cette solution demandant une connaissance certaine d'OpenGL, nous invite les personnes intéressées à étudier les commentaires de la classe *GestionduBumpMapping.java* du projet Echecs3D conjointement à une documentation sur la librairie Jogl.

2.4.7 Analyse du développement JOGL

Nous l'avons déjà souligné, le développement d'affichage imbriqué dans une page internet grâce à l'utilisation du langage de programmation Java ne constitue pour ainsi dire pas une solution web. La mise en place d'un tel projet représente le développement d'une application classique, dont les limites sont imposées par les limitations associées aux applets et par la taille du logiciel, ce dernier doit en effet être déchargé sur la machine cliente lors de chaque utilisation. Nous pouvons dire que la démocratisation des connexions à haut débit repousse chaque année cette limite. Cependant, une attente de chargement nuisant considérablement au confort d'utilisation, il est par conséquent vivement déconseillé de diffuser des applications trop exigeantes en matière de poids sur Internet.

Figure 15

Projet du jeu d'échecs



2.4.7.1 Une grande granularité de développement

Le développement proposé dans ce travail a démontré que l'utilisation de JOGL nécessite la réalisation de nombreux éléments et demande de bonnes compétences en programmation. En raison des concepts avancés nécessaires à leur réalisation, le développeur chargé d'un tel projet devra avoir de solides connaissances en OpenGL, en graphisme ainsi que dans les mathématiques nécessaires aux calculs d'affichage. Malgré les importantes sources d'informations disponible sur le langage OpenGL, que ce soit dans la littérature ou par les diverses communautés Internet, celui-ci reste un langage difficile d'accès qui, en raison de sa complexité, est ainsi dire inaccessible au néophyte.
La charge de développement, sans utilisation de codes préexistants, est par conséquent bien plus lourde que pour certaines technologies concurrentes. Le développeur doit, avant tout, mettre en place la structure capable d'effectuer l'affichage, c'est-à-dire le moteur graphique, il est détourné du but même de l'application que constituent les parties métier du logiciel. Nombre d'options telles que la gestion de la caméra, des lumières, des déplacements, de la sélection à la souris doivent également être développées intégralement.

Le temps d'apprentissage élevé ainsi que la nécessité de la mise en place d'une structure de base font qu'une solution utilisant cette technologie ne peut définitivement pas être un projet isolé. Seule une suite de plusieurs projets, réutilisant les compétences et les structures mises en place, permettront de rentabiliser les investissements nécessaires.

D'un autre point de vue, la grande granularité du développement qui est offert par cette solution permet une paramétrisation au plus près des besoins utilisateur. La livraison du logiciel sera alors épurée de tous les éléments superflus. Dans le même ordre d'idée, la grande liberté offerte permet, moyennant les connaissances requises, le développement d'effets graphiques extrêmement intéressants, limités uniquement par l'imagination et le temps à disposition du programmeur et dont le résultat sera certainement apprécié par l'utilisateur.

Soulignons que la technologie Java a pour elle l'avantage de l'utilisation d'un langage unique pour la globalité du projet ainsi que de la centralisation du programme en un seul fichier. Ces deux points facilitent de manière notable la maintenance de l'application.

Concernant le déploiement, la solution offerte par Sun est également des plus attractives. Un client possédant la JVM aura pour toute manipulation à effectuer une éventuelle validation de certificat. Du fait de la compatibilité multiplateforme de la JVM, la diffusion d'une telle réalisation est de plus extrêmement puissante dans le sens où tous les navigateurs et systèmes d'exploitation sont supportés. Ceci confère une inter-portabilité quasi totale.

Parmi les autres atouts liés à l'utilisation de Java, notons encore que les technologies Java sont enregistrées sous une licence GPL décrite précédemment, conférant ainsi une grande souplesse d'utilisation.

2.4.7.2 Difficultés rencontrées lors du développement Jogl

L'une des premières embûches auquel est confronté le développeur java se situe dans le format de diffusion du logiciel. Les classes sont fournies sous forme d'un package qui demande en outre une signature. Ainsi, une mise en place correcte une lecture approfondie de la documentation disponible sur le sujet. Les explications développées sur ce point dans le chapitre concernant le développement devraient apportées, nous l'espérons, des indications suffisantes au débutant intéressé par ce sujet.
La principale cause de perte de temps à laquelle nous avons été confrontés provient de la nécessité de l'implantation conjointe des différents éléments du logiciel. En effet, malgré l'existence de bons tutoriels sur les différentes options OpenGL, telles que la réflexion ou la sélection souris, il a été souvent malaisé de rassembler ces différentes techniques dans une seule entité. Le manque de vue d'ensemble des informations disponibles sur Internet fut un frein non négligeable à la réalisation de ce projet.

La question du déploiement doit être également bien étudiée. La diffusion de programme Java via internet étant possible par l'utilisation du format JNLP, nous nous sommes dans un premier temps tournés vers cette solution. Malheureusement, l'exécution de celle-ci implique l'installation d'options spécifiques sur le serveur Web les hébergeant. Notre hébergeur Internet ne supportant pas lesdites options, nous avons dû nous mettre à la recherche d'une solution alternative. La solution que nous avons finalement retenue fut l'emploi du JnlpAppletLauncher.

Un dernier point est enfin à relever à propos des tutoriels auxquels nous faisons

référence dans ce dossier. Bien que leur présence apporte généralement une aide précieuse, voir dans certain cas, la seule source d'informations disponible sur un problème précis, ils peuvent s'avérer trompeurs et sources de perte de temps si l'utilisateur ne prend pas garde. Il arrive en effet fréquemment qu'ils soient fragmentaires, incomplets ou encore qu'ils fassent appel à l'utilisation de ressources extérieures dont l'implantation n'est pas toujours possible ou souhaitable pour le logiciel en développement. C'est le cas par exemple, de librairies développées dans le cadre d'un autre tutoriel. Pour finir, la librairie JOGL provenant d'OpenGL est généralement utilisée avec le langage C++. De ce fait, la majorité des tutoriels sur la question emploient ce langage, ce qui rend leur compréhension rébarbative pour celui à qui la syntaxe C++ est inconnue.

2.5 Pour conclure nos développements

Par l'expérience pratique détaillée au chapitre deux, nous avons pu voir ce qu'implique la mise en place d'une solution 3D sur internet. Tout d'abord, chaque technologie à une orientation de prédilection. En utilisant la technologie JOGL, la réalisation de notre vitrine virtuelle aurait été largement plus coûteuse en termes de temps du fait de la nécessité de développement d'un moteur graphique et de la navigation. Inversement, le développement du jeu d'échecs aurait été problématique sur une technologie X3D en raison de l'architecture non-orientée objet de Javascript, de l'absence de certaines options graphiques ainsi que des allers-retours d'informations entre les différentes technologies AJAX.

3 Analyse

3.1 Le future du Web 3D

Il est aujourd'hui impossible de pouvoir prédire avec exactitude ce que sera le web de demain et la place de la 3D dans celui-ci. Néanmoins, il est évident qu'il sera grandement orienté par les entreprises les plus présentes sur le marché. Ainsi, l'examen de la position que celles-ci ont prise par rapport à ce domaine spécifique nous permet d'entrevoir le futur contenu de nos écrans.

IBM s'est par exemple fortement tourné vers les questions d'utilisation de mondes virtuels par les entreprises. Étant l'entreprise la plus représentée dans l'univers virtuel qu'est « Second Life », une section privée de ce metavers[5], gérée et hébergée par IBM à été développé parallèlement à l'utilisation de la plate forme officielle offerte par Linden Lab. Cet espace offre un support pour le développement des possibilités d'e-learning, de recherche marketing et de conférences virtuelles.

Le leader mondial du software qu'est Microsoft a quant a lui récemment développé un nouveau système de présentation graphique nommé WPF, acronyme de « Windows Presentation Foundation ». Ce système basé sur le l'environnement framework 3.0 .net et utilise le format XAML (eXtensible Application Markup Language), basé sur le standard XML pour la transmission de contenu graphique en deux ou trois dimensions. Ce format permet un déploiement de contenu 3D sur Internet via les XBAPs (XAML Browser Applications) intégrés à Internet Explorer ou en utilisant un plug-in pour Mozzila FireFox.

Concertant la position de Microsoft sur le sujet, citons le co-créateur de la société qui a déclaré lors d'une conférence de presse sur les technologies Web.

> *« In fact, one of the reasons 3D failed is that if the refresh rate is not just incredibly high, and the quality is not incredibly good, 2D is better, and you can*

[5] Meta-univers synonyme de monde virtuel.

just read the information and create the illusion of whatever you want in your brain, as opposed to something that literally can make you feel poorly because it's just jittery and out of date. » *(Bill Gates)*

Bill Gates relève que de nombreux efforts dans ce domaine restèrent vains en raison de la relative pauvreté de l'immersion des mondes 3D. Selon lui, tant que ceux-ci n'apporteront pas une beauté et une facilité d'utilisation reflétant la réalité, l'utilisateur préférera un affichage bidimensionnel classique. Encore selon lui, l'une des raisons de la difficulté d'expansion de la 3D sur internet provient notamment du manque de périphériques adaptés à une navigation dans un monde en trois dimensions. Réciproquement, le manque de demande sur le marché actuel est une des causes de la faible offre de tels périphériques.

La solution X3D du web 3D consortium pourrait être un bond en avant concernant l'unicité des formats dans ce domaine. Malheureusement le manque de leader du marché attaché à ce projet rend l'avenir de ce format incertain. Tel l'exemple d'Intel, ancien membre du W3C, qui abandonna le projet pour développer avec Macromédia un format propriétaire basé sur le logiciel Shockwave.

Suite au rachat de Macromedia par Adobe, le support de Shockwave fut abandonné. Pour ne pas répéter les erreurs du passé, Adobe et Intel décidèrent de s'associer à plusieurs géants de l'industrie dans le but d'être au plus près des attentes du marché. Ainsi fut créé le 3D Industry Forum, regroupant notamment Boeing, HP, Airbus et Bentley Systems. De la naquit l'U3D (Universal 3D), format d'échange pour le contenu 3D. Ce format est le seul utilisé par Acrobat pour l'insertion de contenu tridimensionnel dans le format PDF. Parallèlement, la philosophie d'Adobe concernant flash et ses futures évolutions est claire: Flash est et restera un format d'affichage vectoriel. Ainsi, la seule évolution prévue dans le sens d'un affichage 3D est l'implémentation de fonctions permettant la réalisation d'effet de perspective.

Quant au concurrent direct de Microsoft dans le domaine des navigateurs web, Mozilla Firefox, il développe un add-on permettant l'affichage d'objet 3d dans les canevas HTML. Ce projet, connu sous le nom de Gecko, utilise une accélération matérielle via la librairie graphique OpenGL et sera disponible pour la version 3.0 de Firefox.

De son côté, Google, a fait son impressionnante entrée dans le monde de la 3D avec

son fameux Google Earth et du logiciel associé permettant la modélisation intuitive d'immeubles : Google Sketchup utilisant le format COLADA. Google a récemment ajouté au panel des fonctionnalités offertes, sous le nom de Google Lively, la possibilité pour tout un chacun de créer sur sa page internet des salles de discussions virtuelles de façon simplifiée.

Une multitude de formats similaires sont en cours de développement, tel que le 3DMLV, un support au format XML, développé par 3D Technologies R&D, ou encore le projet d'Unity3D proposant une solution de moteur graphique extrêmement performante, dédiée aux développements de jeux on-line.

Figure 16

Moteur graphique préconstruit

Source : Unity3D.com (Aout 2008)

À la vue de cette grande disparité de formats et technologies disponibles, il apparaît de façon assez claire que nulle révolution 3D n'est à attendre pour les prochaines années, mais bien une évolution des solutions actuelles qui permettra une démocratisation du développement d'objets isolés comme le cas de notre vitrine virtuelle par exemple. Certains sites Internet essayent néanmoins d'évaluer l'éventail des possibilités futures. C'est le cas de www.metaverseroadmap.org qui, via sondage auprès de professionnels du domaine, tente tant bien que mal d'imaginer ce qu'offrira le Web 3D en 2016.

3.2 Le marché de la 3D en suisse

Loin des grands projets des poids lourds de l'industrie, qu'en est-il de la réalité du marché en Suisse? Pour étudier l'étendue de l'implantation des technologies 3D web en suisse romande nous avons pris contact avec des professionnels provenant d'entreprises actives dans la diffusion internet, basées sur l'arc lémanique. Ces interviews ont permis de relever les points importants développés ci-dessous.

Bien que les professionnels s'accordent pour confirmer l'accroissement de l'intérêt qu'une visite virtuelle peut apporter à un site internet, il a été clairement relevé que le marché suisse pour ce type de produits est restreint. En effet, seule une minorité des entreprises interrogées ont l'occasion de travailler sur de tels projets.

Les raisons en sont multiples. La principale d'entre elles consiste dans le fait que les besoins de la clientèle dans ce domaine restent faibles. En effet, la majorité des projets utilisant un affichage en trois dimensions sont destinés à la mise en avant de produits ou éventuellement à la réalisation de panoramas. Ce marché cible donc principalement les entreprises proposant des produits basés sur leur attrait visuel, comme c'est le cas de l'horlogerie pour citer le plus fort besoin en la matière en Suisse.

Une autre cause du manque de possibilités offertes réside dans la nécessité de connaissances spécifiques pour mettre en œuvre de tels projets. Le marché de l'emploi en Suisse a effectivement de la peine à subvenir à ce type de demandes. Pour une agence Web, trouver les compétences requises relève du défit. Le temps d'apprentissage élevé et le manque de formations disponibles oblige les compagnies Web à trouver des personnes autodidactes prêtes à investir de leur temps pour s'auto-former. Cette difficulté à trouver une main d'œuvre qualifiée à évidemment un impact sur les coûts de réalisation.

Selon l'Office Fédéral de la Statistique (OFS), plus de 90% des entreprises dans le domaine informatique emploient moins de 10 employés. Ainsi, la grande majorité des entreprises informatiques suisses n'atteignent pas la taille critique susceptible de pouvoir supporter l'infrastructure humaine et logicielle nécessaire à de tels développements.

Les efforts pour encourager les entreprises à utiliser ces technologies se heurtent

souvent aux limitations de budget. En effet, en plus du développement en soi, le besoin de modélisation des objets à présenter représente un surcoût non négligeable. Des entreprises spécialisées existent *(*à l'étranger), mais la création de chaque objet peut nécessiter, selon la qualité voulue, entre une demi-journée et quelques jours de travail.

Dans les sociétés de développement Web où de tels projets sont effectivement menés, ceux-ci restent des exceptions. L'utilisation de telles technologies n 'est alors jamais mise en œuvre pour la totalité d'un site, mais uniquement pour un certain nombre d'éléments précis touchant entre une et deux pages, correspondant approximativement à 15% de la totalité d'un site internet. Les technologies utilisées varient, quant à elles, de cas en cas selon le type du projet et les compétences à disposition, parmi celles-ci citons Java, Quicktime VR, Flash Swift et Papervision 3D.

Concertant le futur de la 3D, les avis des personnes interrogées divergent grandement selon leur propre expérience à cet égard. Celles qui participent déjà à de tels projets qui prédisent un bel avenir pour le Web3D. Les autres estiment que les efforts se centreront plutôt sur les développements destinés aux nouveaux périphériques comme que le fameux I-Phone d'Apple.
Il a été relevé que l'utilisation de scènes pré-calculées, telles que celles réalisées avec des logiciels comme Maya, est en régression, notamment dûe au prix élevé des rendus. Ces limites laissent le champ libre pour des développements modulaires, permettant une création de scènes personnalisées à partir de libraires d'objets réutilisables.

3.3 Conclusion

Ce travail nous amène à conclure que, qu'en dehors de certains cas efficaces, l'utilisation de la 3D sur internet a encore du chemin à faire pour remplacer le contenu actuel des sites Web. Il est évident que l'un des points les plus frappants, est la disparité des formats utilisés. Ce manque de standardisation freine notamment l'apparition de la gestion de contenus 3D par les navigateurs Web classiques. La nécessité actuelle d'installer des plug-in pour chaque technologie rencontrée représente un obstacle décourageant pour bien des utilisateurs. Ceux-ci sont également de plus en plus réticents face à l'installation de logiciels provenant de la toile en raison du grand nombre de logiciels malveillants accessible sur Internet.

Les applications basées sur JAVA, comme Shout 3D par exemple, sont actuellement les seules solutions capables d'un affichage 3D sans aucun besoin de déchargement. Ces solutions apportent également une portabilité presque totale, atout rare parmi les technologies connues.

Outre les différents problèmes de format cité précédemment, notre voyage parmi les vastes discussions sur la 3D du Web, nous a permis de relever que l'un des principaux freins à l'utilisation d'affichage 3D réside dans le besoin impératif de création de modèles virtuels. Ceux-ci faisant appel à des compétences étendues dans le domaine du dessin en trois dimensions. Les modeleurs du marché demandent effectivement une prise en main extrêmement longue avant d'obtenir un quelconque résultat. Nous pouvons dresser un parallèle avec la peinture avant l'apparition de l'appareil photographique. La 3D est actuellement du ressort de l'art et non du particulier. Le jour où des objets, voire des paysages, pourront être capturés aussi simplement que de prendre une photographie, la 3D connaitra sa vraie démocratisation. Dans ce sens des scanners 3D commencent à apparaître sur le marché. C'est le cas des produits de la société Worux. Néanmoins, l'utilisation de ceux-ci reste onéreuse puisque l'acquisition d'un tel scanner de taille réduite dépasse les 2’500 euro.

Comme l'a relevé Bill Gates lors de son interview citée précédemment, les interfaces que nous connaissons et utilisons tous les jours que sont le clavier et la souris ne sont pas adaptées aux déplacements dans un monde virtuel. Il existe de tels périphériques, mais l'absence de possibilité d'utilisation pour le grand public les réserve à des cas isolés d'utilisation, dont le prix exclut un emploi familiale.

Malgré ces divers obstacles, l'intérêt porté par les principaux acteurs de l'informatique pour ce domaine est important et le nombre impressionnant de projets communautaires nous promet la venue croissante de projets de ce type sur nos ordinateurs.

Nous pouvons également espérer une plus grande collaboration entre les acteurs majeurs pour éviter les doublons technologiques qui constituent le marché actuel. Une telle réalisation ne se fera sans doute pas du jour au lendemain, mais serait dans la continuation logique du développement de ces technologies. Pour ce faire, une solution Open-Source serait certes idyllique. Malheureusement, ces options ont la réputation de lenteur de mise en place et de développement. Ainsi de nombreux

professionnels mettent leurs espoirs dans les solutions propriétaires des géants du logiciel informatique.

Une autre question débattue à de nombreuses reprises de part le Web concerne l'utilisation ou non de solutions proposant un moteur graphique intégré. Nous l'avons vu lors du projet basé sur la technologie JOGL proposé dans ce travail, la mise en place de l'infrastructure d'affichage constitue en effet une perte de temps notoire pour un développement Web. Il existe de tels moteurs pour Java, mais leur utilisation implique évidemment des contraintes à prendre en compte. Comme un, par exemple, l'alourdissement du logiciel du à un trop large panel d'options. Heureusement, l'évolution des connexions à haut débit permet l'utilisation d'applications Web bien plus lourdes que par le passé. En effet selon l'OFS, la Suisse dispose de plus de deux millions de connexions de ce type.

Pour conclure ce travail, nous reconnaissons que de nombreux aspects doivent être améliorés dans le domaine des interfaces 3D pour le Web. Néanmoins, nous recommandons à toutes les sociétés actives dans le développement Internet de suivre avec attention les évolutions en la matière, et de ne pas risquer de passer à coté d'opportunités de tels développements. Ceux-ci pouvant apporter une plus-value notable à un site commercial qui ne demande qu'un investissement limité selon la technologie utilisée, comme c'est le cas de l'utilisation de Flash ou du format X3D.

3.4 Ce que ce projet nous a apporté

Depuis toujours grandement intéressé par le graphisme en trois dimensions, débutant par la réalisation de modèle d'objets, puis par le développement d'application vidéo ludique, ce travail de diplôme nous a donné l'opportunité de nous plonger dans un domaine actuellement réservé aux passionnés, pourrait bien devenir une facette incontournable des technologies Web dans les années à venir. Malgré une recherche souvent laborieuse d'informations sur le sujet, l'étude de ce domaine ainsi que ses technologies fut extrêmement intéressante. Pour ne citer que quelques points importants, la découverte d'un domaine où tout ou presque reste à faire que ce soit d'un point de vue technologie ou commercial, les diverses techniques graphiques, comme le bump mapping, ainsi que les structures de communication que sont les architectures client-serveur et AJAX, fut un grand enrichissement personnel.

Le temps accordé à ce travail n'est malheureusement pas suffisant pour parcourir tous les recoins du Web 3D. Ce champ est aussi vaste que disparate et de nombreuses technologies non abordées dans cette travail, mériteraient, nous en sommes convaincus, une étude approfondie. Ce petit monde étant amené à évoluer dans les prochaines années, il est également ardu de faire ressortir les solutions qui feront le web de demain. Cette volubilité, loin de nous décourager, nous incite au contraire à continuer la découverte de ce monde fascinant et nous n'hésiterons pas nous replonger dans ce sujet si l'occasion se présente.

Glossaire

3DStudio Max (Autodesk)
Logiciel de CAO dédié au dessin en trois dimensions (www.autodesk.fr)
Acrobat3D (Adobe) Lecteur de fichier .pdf dédié à l'échange de données (www.acrobat.com)

AJAX
Asynchronous JavaScript And XML, représente une structure de communication pour site Internet

Alternativa 3D
Environnement de développement 3D basé sur Adobe Flash(alternativaplatform.com)

API
Application Programming Interface, interface de programmation pour la communication vers un composant informatique

BSCONTACT (Bitmanagement)
L'un des leaders des lecteurs X3D (www.bitmanagement.com)

Blender
Modeleur 3D Open Source (blender.org)

CAO
Conception Assisté par Ordinateur

COLADA
Format d'échange de scène en trois dimensions

CPU
Central Processing Unit, appelé généralement processeur

Director (Adobe)
Logiciel de traitement multimédia (www.adobe.com/products/director)

DirectX (Microsoft)
API graphique dédié à l'affichage en trois dimensions sous Windows

DOM
Document Objet Model, standard du W3C concertant l'accès entre les langages de programmation

Flash (Adobe)
Technologie d'affichage vectoriel dédiée à internet comprenant un lecteur, un logiciel de conception et utilisant le format .swf (www.adobe.com/products/flash)

FLUX
Logiciel de lecture de format X3D Open Source

FTP
File Transfer Protocol, Protocole de communication pour le transfert de donnée et utilisant le port 21

GPL GNU
General Public Licence, licence fixant les conditions de distribution pour des logiciels libres (www.gnu.org/copyleft/gpl.html)

GPU
Graphics Processing Unit, Unité de calcul dédié à l'affichage en trois dimensions

HTML
Hypertext Markup Language, format de donnée standardisé utilisé pour l'affichage de page sur internet

IANA
Internet Assigned Numbers Authority, association en charge de la gestion des adresses IP pour internet (www.iana.org)

JAVA
Technologie de programmation développée par SUN Microsystems (www.java.net)

JAVA 3D
API graphique pour JAVA (www.java3d.org)

JavaScript
Language de script dédié au développement de pages Web interractives (developer.mozilla.org/en/JavaScript)

JOGL
API graphique pour JAVA basé sur OpenGL (jogl.dev.java.net)

JNLP
Format associé à la technologie Java Web Start pour l'exécution de contenu Java sur Internet

JVM
Java Virtual Machine, interpréteur permettant l'exécution de code Java

MIME
Multipurpose Internet Mail Extensions, Un standard de codage de caractère initialement dédié au courrier virtuel, il est également utilisé par les serveurs internet pour déterminer le type de codage de fichier comme le XML

MMORPG
Massively Multiplayer Online Role-Playing Game, application vidéo ludique, basée sur l'interaction entre un grand nombre de joueur via Internet

PHP
Hypertext Preprocessor, langage de programmation dédié à la conception de page web (www.php.net)

OFS
Office Fédéral de la Statistique (www.statistique.admin.ch)

OpenGL
API graphique Open Source dédié à l'affichage en trois dimensions (OpenGL.org)

OpenWorlds
Société développant un lecteur X3D (www.openworlds.com)

Papervision3D
Interface d'affichage en trois dimensions dédié à Adobe Flash (www.papervision3D.org)

SAI
Scene Access Interface, Interface d'accès au contenu de lecteur X3D prévu par la technologie AJAX3D (www.ajax3d.org)

Sandy
Moteur 3D basé sur la technologie Adobe Flash (www.flashsandy.org)

SDK
Software Developpment Kit, outils et documentation nécessaire au développement informatique

Shockwave3D (Adobe)
Lecteur multimédia permettant la diffusion de contenu Adobe Director sur Internet (www.adobe.com/products/shockwaveplayer)

SQL
Structured Query Language, standard pour l'accès aux données stockées dans une base de données

Swift3D (Electric Rain)
Modeleur et moteur graphique basée sur Adobe Flash (www.erain.com/product/swift3d)

SwirlX3D (Pinecoast Software)
Lecteur de format X3D (www.pinecoast.com)

Telnet
TELecommunication NETwork, protocole de communication utilisé pour la gestion d'équipement à distance et utilisant le port 23

TCP/IP
Transmission Control Protocol/Internet Protocol, ensemble de protocole de communication permettant l'échange de donnée via un réseau informatique

U3D
Universal 3D, format développé par le 3D Industry Forum dédié à l'échange de donnée en trois dimensions

URL
Uniform Resource Locator, chaine de caractère représentant l'adresse d'une ressource disponible sur Internet

VRML
Virtual Reality Modeling Language, Format dédié à la diffusion Internet, représentant de scène en trois dimensions (www.web3d.org/x3d/vrml)

W3C

World Wide Web Consortium, Organisation principale en charge du développement de standard pour Internet (www.w3.org)

X3D

Format de fichier développé par le Web 3D Consortium représentant une scène en trois dimensions et utilisant un format de balises XML (www.web3d.org/x3d)

XAML

Extensible Aplications Markup Language, format développé par Microsoft pour l'échange d'information graphique.

XJ3D

Projet Open Source de lecteur X3D basé sur la technologie JAVA (www.xj3d.org)

XML

Extensible Markup Language, format de fichier développé par le W3C utilisant un system de balises extensibles

Bibliographie

Portails et articles

www.3d-test.com (19.07.2008)
Un site français sur le monde de la 3D, comporte de nombreuses interviews très intéressantes
www.web3d-fr.com (01.08.2008)
Un site français dédié à la 3D web, comportant notamment des interviews, des exemples et des tutoriaux
www.3dtrue.com (01.08.2008)
Portail référençant de nombreux exemple et tutoriaux concernant le monde de la 3D
www.future-internet.eu (01.08.2008)
Portail européen sur le développant du future d'internet
www.ideasint.blogs.com/ideasinsights/2007/10/is-the-future-o.html (01.08.2008)
Article discutant du future de la 3D sur Internet
www.techlearning.com/blog/2007/03/as_i_read_about_the.php (24.06.2008)
Article sur le web 3d et notamment sur les univers virtuels
www.neteco.com/67714-web-3d.html (24.06.2008)
autre article sur le web 3D
www.pperes.wordpress.com (24.06.2008)
Article sur les mondes virtuels on-line
www.casa.ucl.ac.uk/martin/virtual_cities.html (20.6.2008)
Article sur les villes virtuelles
www.modelbenders.com/2008/02/server-side-3d-rendering.html (20.08.2008)
Article proposant une utilisation de la 3d dont le rendu est effectué coté serveur
www.metaverseroadmap.org (27.08.2008)
Site proposant une large vue des mondes virtuelle exitant, ainsi que d'une prévision pour ceux ci en 2016
www.pixelsebi.com/2008-06-04/one-metaverse-different-visual-layers (27.08.2008)
Article présentant une vision multicouches de l'utilisation d'internet et des mondes virtuels
www.itnews.com.au/News/NewsStory.aspx?story=42704 (16.06.2008)
Article développant la 3d internet
www.itnews.com.au/News/42140,ibm-to-fund-3d-internet-project.aspx (16.06.2008)
Article d'IBM sur le net 3d
www.itnews.com.au/News/42140,ibm-to-fund-3d-internet-project.aspx (04.08.2008)
Article sur ce même sujet
www.builderau.com.au/news/soa/Intel-s-3D-divorce-rate/0,339028227,339129580,00.htm (27.08.2008)
Article sur la séparation d'Intel du Web 3D Consortium
iit-iti.nrc-cnrc.gc.ca/vit-tiv/display-exhibit_e.html (02.07.2008)
Article relatent de l'utilisation de la 3D pour des musées

Exemples

Entreprises

www.idees-3com.com (01.08.2008)
Entreprise spécialisée dans la conception de vitrines on-line
www.unity3d.com (13.08.2008)
Compagnie proposant un moteurs de jeux on-line extrêmement puissant.
www.victoriacouture.com (01.08.2008)
Exemple de visite de shopping virtuel

www.geosim.co.il (04.08.2008)
Société de création de ville virtuelle
www.planet9.com (08.08.2008)
Une autre société spécialisée dans la création de ville virtuelle
www.madlix.com (02.08.2008)
Compagnie de création de vitrine virtuelle
www.cybertown.com (08.08.2008)
Offre de divers espace de discussion en 3D
www.internet3d.fr (20.07.2008)
Entreprise de déploiement d'élément 3D web sur internet

Divers

www.abrahamjoffe.com.au/ben/canvascape (08.08.2008)
Labyrinthe en 3D
www.intoronto.com (04.08.2008)
Visite virtuelle de Toronto
www.jmol.sourceforge.net (08.08.2008)
Projet de représentation moléculaire développé en java.
www.polyhedra.org (08.08.2008)
Collection de polygone 3D
www.casa.ucl.ac.uk/planning/cities.htm (08.08.2008)
Lient vers divers villes possédants leur homologue virtuel
www.web3dart.org (08.08.2008)
Sélection des meilleurs animation 3d sur le web
www.tetedechoux.ifrance.com/espace_gainsbourg.htm (08.08.2008)
Balade virtuel sur le thème de Gainsbourg
maquette.virtuelle.free.fr/expo.html (02.07.2008)
Visite virtuelle du projet urbain du parc des expositions
www.wurmonline.com (05.07.2008)
MMORPG 3D, gratuit et réalisé avec des technologies Java

Musées

www.museevirtuel.ca (20.6.2008)
Le musée virtuel du canada
www.civilisations.ca/aborig/inuit3d/vmcinuit_f.html (02.07.2008)
Musée virtuel sur les Inuits
3dmuseum.geology.ucdavis.edu/contents.html (02.07.2008)
Exposition d'ossement en trois dimensions

Technologies

www.uselesspickles.com/triangles/demo.html (17.07.2008)
Démonstration d'affichage 3D à l'aide unique de balises HTML et JavaScript

Flash

www.tutorialized.com (20.06.2008)
Site regroupant de nombreux tutoriels, dont en autre sur les technologies 3D basées sur Flash
www.pv3d.org (01.08.2008)
Site dédié à Papervision offrant de nombreux exemples
www.scolring.org (01.08.2008)
Portail communautaire de la technologie SCOL

X3D

www.extremetech.com/article2/0,1697,2041418,00.asp (04.08.2008)
Article concernant les lecteurs X3D

www.ajax3d.org (04.08.2008)
Site officiel de la technologie AJAX3D développé par Media Machine
www.web3d.org/x3d/content/examples/Basic/development (08.08.2008)
Une grande série d'exemple de code X3D
www.bitmanagement.com/developer/contact_j/doc/3dauthoring/3dapi3.html
(07.08.2008) Spécification des commandes DOM pour le lecteurs X3D BSContact J
www.web3d.org/x3d/specifications/ISO-IEC-19777-1-X3DLanguageBindings-ECMAScript
(07.08.2008)
Définition ISO du format X3D
www.cic.nist.gov/vrml/vbdetect.html (07.08.2008)
Revue des lecteurs VRML/X3D du marché
www.web3d.org/tools/viewers_and_browsers (22.08.2008)
Description des divers plug-ins X3D existant

AJAX3D

www.3donthewebcheap.blogspot.com/2007/04/ajaxing-x3d-sequencer.html (08.08.2008)
Article très complet sur la technologie AJAX3D
www.ajax3d.org (14.08.2008)
Site officiel d'Ajax3D
www.vrmlcreations.com/AjaxExamples/AjaxPlaneSensor.htm (08.08.2008)
Différent exemple de réalisation avec Ajax3D
www.hypermultimedia.com/ajax3d (08.08.2008)
Tutoriel de base Ajax3D
www.pinecoast.com/tutorial001.htm (13.08.2008)
Site de Pinecoast Software regroupant qulques tutoriaux simples

JAVA

www.java.sun.com/javase/technologies/desktop/javawebstart/index.jsp (20.06.2008)
Site officiel de la technologie JavaWebStart
www.java.sun.com/applets (01.05.2008)
Page du SUN concernant les applets Java
www.java.sun.com/j2se/1.5.0/docs/guide/javaws/developersguide/syntax.html (04.08.2008)
Documentation du format JNLP
www.java3d.org (17.07.2008)
Site officiel de la technologie Java3D
www.felixgers.de (23.06.2008)
Contient de nombreux tutoriel et référence de lien pour OpenGL et JOGL
www.nehe.gamedev.net (04.06.2008)
Une série de tutoriel sur OpenGL(pour C++) parmi les plus complets du net.
www.pepijn.fab4.be/software/nehe-java-ports (08.07.2008)
Traduction des exemples du site nehe.gamedev.net pour Java
www.jerome.jouvie.free.fr/OpenGl/index.php (09.07.2008)
Une autre très bonne séries de tutoriels sur JOGL
www.jogl.dev.java.net (01.05.2008)
Site officiel du projet Jogl

OpenGL

www.perception.inrialpes.fr/people/Boyer/Teaching/OpenGL.pdf (04.08.2008)
Un tutoriel sur OpenGL format pdf
www.alrj.org/docs/3D/modelview.pdf (04.08.2008)
Document détaillant l'utilisation de la modèle vue
www.polytech.unice.fr/~buffa/cours/synthese_image/MINI_PROJETS/1999/essi/navarro-chaput-glskel-divers/opengl.html (04.08.2008)
Description du fonctionnement d'OpenGL avec quelques exemple
www.paulsprojects.net/tutorials/simplebump/simplebump.html (24.08.2008)
Nombreux tutoriels sur OpenGL dont un concernant le Bump Mapping

Livres consultés

Open GL 2.0 : Guide officiel, (Broché - 12 juillet 2006)
par Jackie Neider, Mason Woo, Tom Davis, et Dave Shreiner

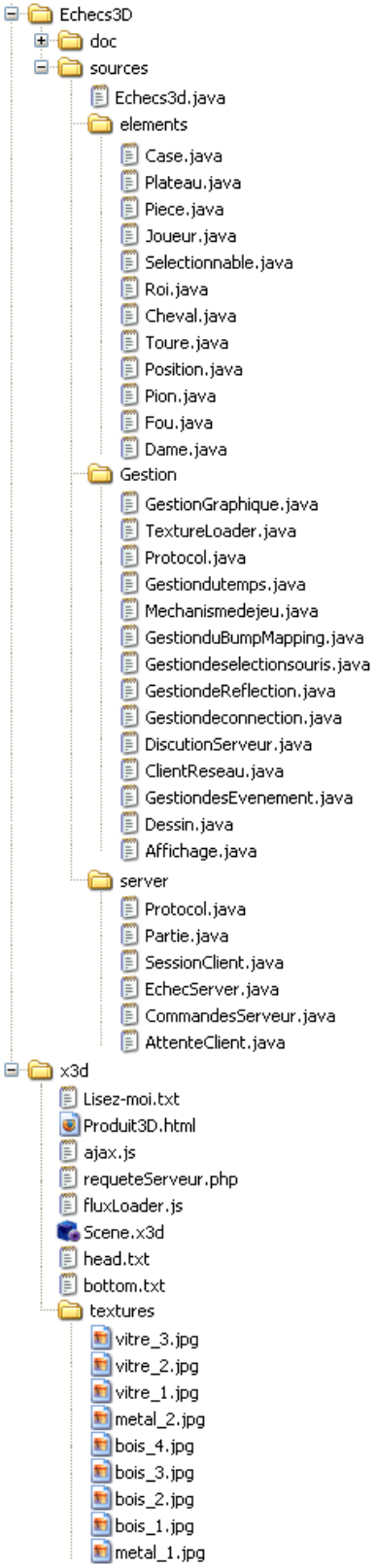
Core Web 3D, (Broché - 25 septembre 2000)
par Aaron E. Walsh et Mikael Bourges-Sevenier

X3D: Extensible 3D Graphics for Web Authors, (Broché - 15 janvier 2006)
par Don Brutzman et Leonard Daly

Au coeur de Java 2 : Tome 1, Notions fondamentales, (Broché - 15 décembre 2004) par Gary Cornell, Cay Horstmann, Christiane Silhol, et Nathalie Le Guillou de Penanros

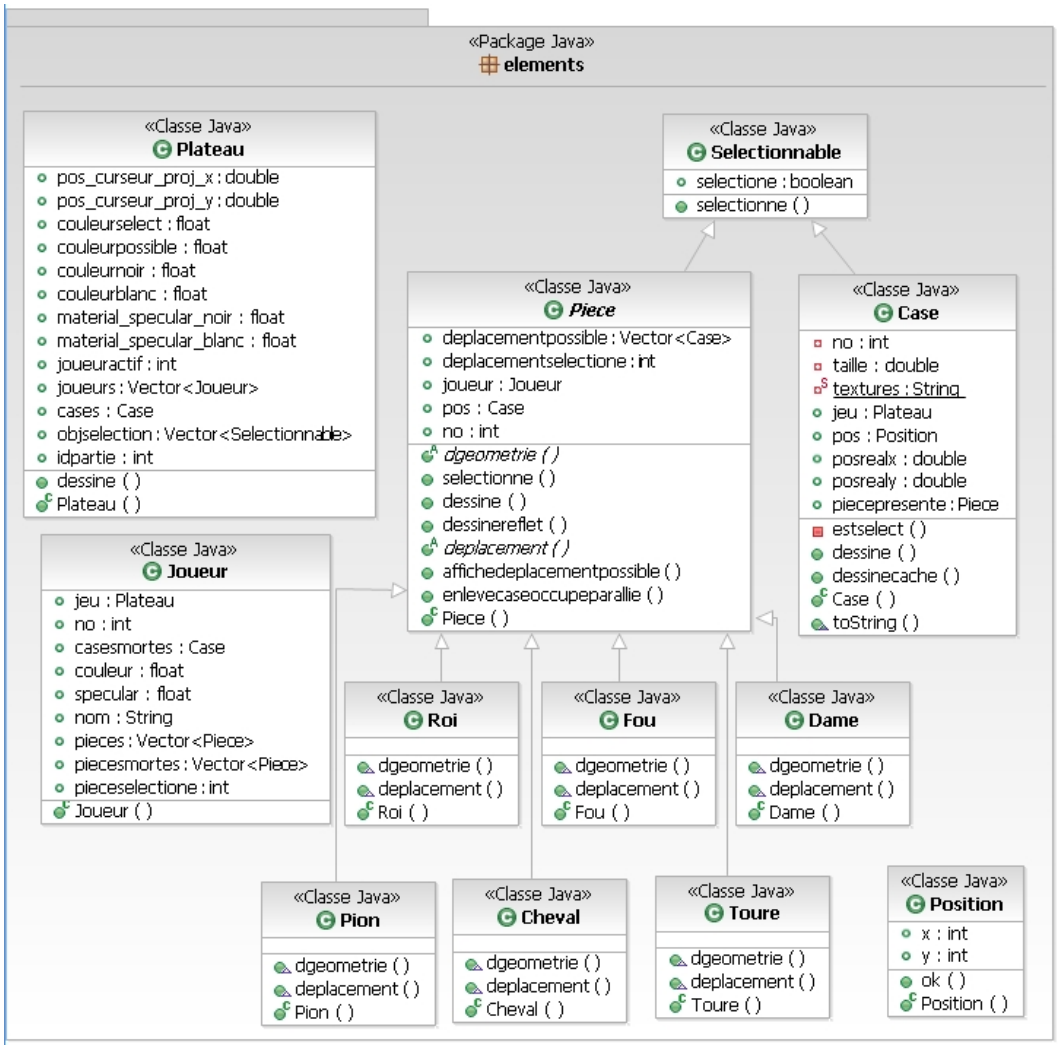
Annexe 1

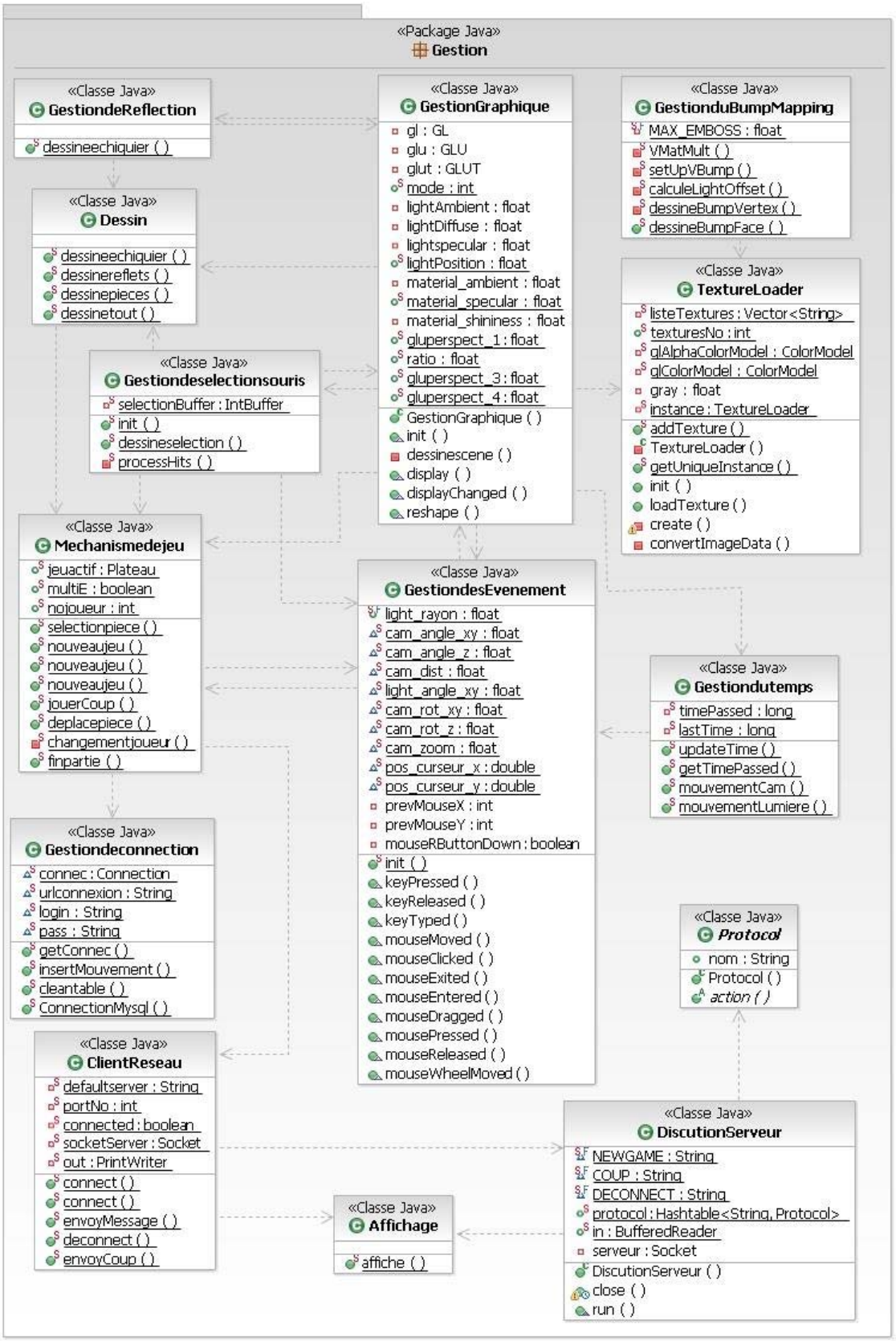
Liste des fichiers

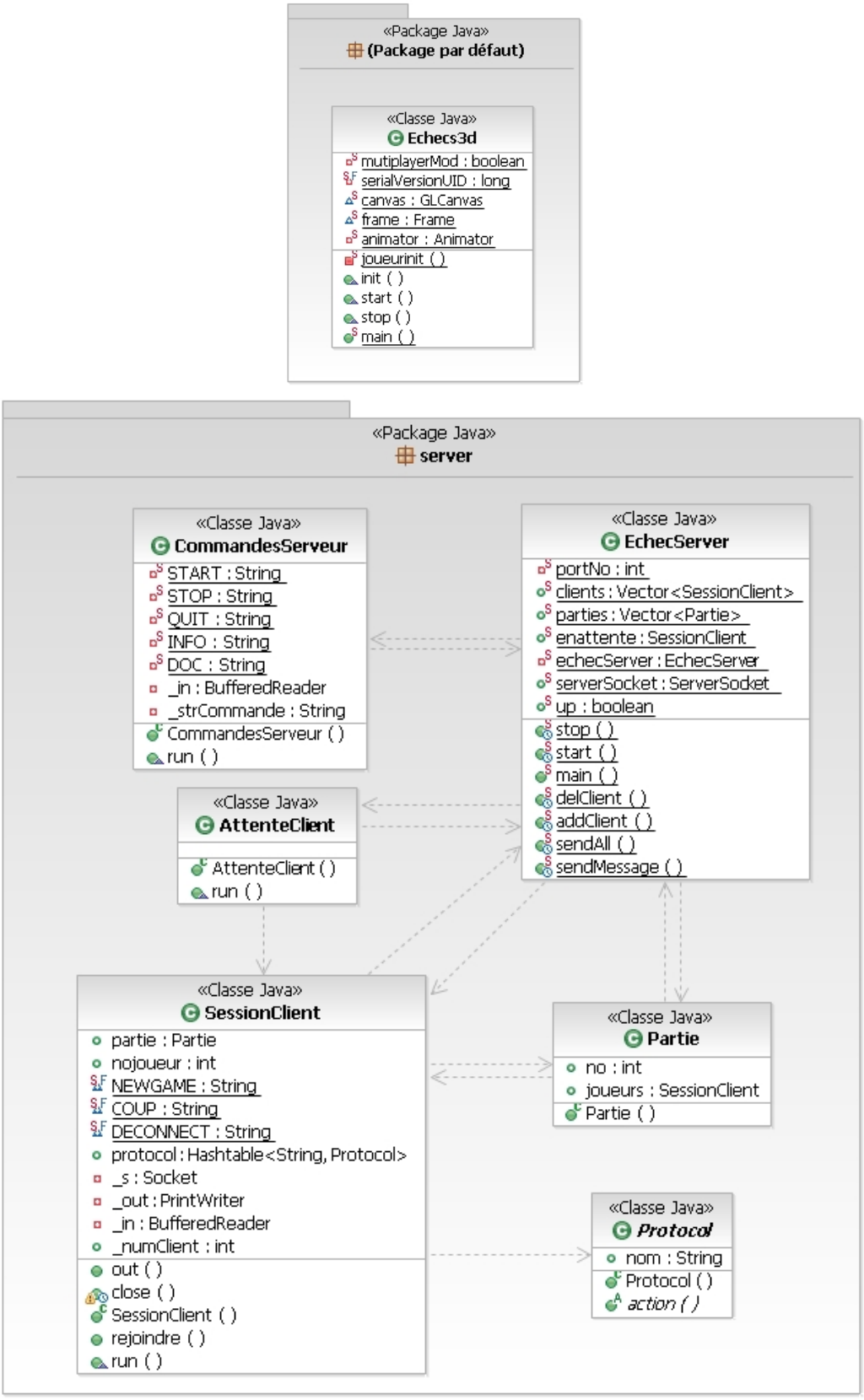


Annexe 2

Diagramme de classe du projet de jeu d'échecs




«Package Java»
Gestion
«Classe Java»
GestiondeReflection
dessineechiquier ( )
«Classe Java»
GestionGraphique
gl : GL
glu : GLU
glut : GLUT
mode : int
lightAmbient : float
lightDiffuse : float
lightspecular : float
lightPosition : float
material_ambient : float
material_specular : float
material_shininess : float
gluperspect_1 : float
ratio : float
gluperspect_3 : float
gluperspect_4 : float
GestionGraphique ( )
init ( )
dessinescene ( )
display ( )
displayChanged ( )
reshape ( )
«Classe Java»
GestionduBumpMapping
MAX_EMBOSS : float
VMatMult ( )
setUpVBump ( )
calculeLightOffset ( )
dessineBumpVertex ( )
dessineBumpFace ( )
«Classe Java»
Dessin
dessineechiquier ( )
dessinereflets ( )
dessinepieces ( )
dessinetout ( )
«Classe Java»
TextureLoader
listeTextures : Vector<String>
texturesNo : int
glAlphaColorModel : ColorModel
glColorModel : ColorModel
gray : float
instance : TextureLoader
addTexture ( )
TextureLoader ( )
getUniqueInstance ( )
init ( )
loadTexture ( )
create ( )
convertImageData ( )
«Classe Java»
Gestiondeselectionsouris
selectionBuffer : IntBuffer
init ( )
dessineselection ( )
processHits ( )
«Classe Java»
Mechanismedejeu
jeuactif : Plateau
multiE : boolean
nojoueur : int
selectionpiece ( )
nouveaujeu ( )
nouveaujeu ( )
nouveaujeu ( )
jouerCoup ( )
deplacepiece ( )
changementjoueur ( )
finpartie ( )
«Classe Java»
GestiondesEvenement
light_rayon : float
cam_angle_xy : float
cam_angle_z : float
cam_dist : float
light_angle_xy : float
cam_rot_xy : float
cam_rot_z : float
cam_zoom : float
pos_curseur_x : double
pos_curseur_y : double
prevMouseX : int
prevMouseY : int
mouseRButtonDown : boolean
init ( )
keyPressed ( )
keyReleased ( )
keyTyped ( )
mouseMoved ( )
mouseClicked ( )
mouseExited ( )
mouseEntered ( )
mouseDragged ( )
mousePressed ( )
mouseReleased ( )
mouseWheelMoved ( )
«Classe Java»
Gestiondutemps
timePassed : long
lastTime : long
updateTime ( )
getTimePassed ( )
mouvementCam ( )
mouvementLumiere ( )
«Classe Java»
Gestiondeconnection
connec : Connection
urlconnexion : String
login : String
pass : String
getConnec ( )
insertMouvement ( )
cleantable ( )
ConnectionMysql ( )
«Classe Java»
Protocol
nom : String
Protocol ( )
action ( )
«Classe Java»
ClientReseau
defaultserver : String
portNo : int
connected : boolean
socketServer : Socket
out : PrintWriter
connect ( )
connect ( )
envoyMessage ( )
deconnect ( )
envoyCoup ( )
«Classe Java»
DiscutionServeur
NEWGAME : String
COUP : String
DECONNECT : String
protocol : Hashtable<String, Protocol>
in : BufferedReader
serveur : Socket
DiscutionServeur ( )
close ( )
run ( )
«Classe Java»
Affichage
affiche ( )


«Package Java»
(Package par défaut)
«Classe Java»
Echecs3d
mutiplayerMod : boolean
serialVersionUID : long
canvas : GLCanvas
frame : Frame
animator : Animator
joueurinit ( )
init ( )
start ( )
stop ( )
main ( )
«Package Java»
server
«Classe Java»
CommandesServeur
START : String
STOP : String
QUIT : String
INFO : String
DOC : String
_in : BufferedReader
_strCommande : String
CommandesServeur ( )
run ( )
«Classe Java»
EchecServer
portNo : int
clients : Vector<SessionClient>
parties : Vector<Partie>
enattente : SessionClient
echecServer : EchecServer
serverSocket : ServerSocket
up : boolean
stop ( )
start ( )
main ( )
delClient ( )
addClient ( )
sendAll ( )
sendMessage ( )
«Classe Java»
AttenteClient
AttenteClient ( )
run ( )
«Classe Java»
SessionClient
partie : Partie
nojoueur : int
NEWGAME : String
COUP : String
DECONNECT : String
protocol : Hashtable<String, Protocol>
_s : Socket
_out : PrintWriter
_in : BufferedReader
_numClient : int
out ( )
close ( )
SessionClient ( )
rejoindre ( )
run ( )
«Classe Java»
Partie
no : int
joueurs : SessionClient
Partie ( )
«Classe Java»
Protocol
nom : String
Protocol ( )
action ( )