## D) Le serveur (Benjamin MAURIN et Steve GINER)

### 1) Le choix du langage Java

Nous avons choisi Java pour différentes raisons :

* C’est un langage qui permet de programmer avec moins d’erreur qu’un langage comme le C++.
* C’est un langage multiplateforme, ce qui permet de reprendre le code sans aucun problème.
* C’est un langage orienté objet qui permet d’implanter facilement des applications multitâches.
* - En ce moment nous étudions le C++ en cours et le projet nous permet de ne pas perdre la main avec le Java et même mieux observer les différences entre ces deux langages.
* - Et pour finir, étant donné que les trois autres clients dépendent de notre avancement, il était plus raisonnable d’utiliser un langage plus structuré, sur lequel nous sommes plus compétents et avec lequel nous avons déjà expérimenté le multitâches sérieusement.

### http://www.agentgroup.unimo.it/pppj08/images/sun_logo.pnghttp://www.cahierdescharges.com/blog/wp-content/uploads/2011/10/java-langage-programmation.png2) Présentation de Java

Java est un langage de programmation né en 1995. Il a été par James Gosling et Patrick Naughton, employés de Sun Microsystems, avec le soutien de Bill Joy (cofondateur de Sun Microsystems en 1982).

C’est un langage orienté objet semi-interprété qui est facilement portables sur différents systèmes d’exploitation tel qu’UNIX, Windows, Mac OS … Sa portabilité est garantie par une plate-forme spécifique à chaque système.

Cette plate-forme est la [machine virtuelle](http://fr.wikipedia.org/wiki/Machine_virtuelle) qui effectue la [traduction](http://fr.wikipedia.org/wiki/Interpr%C3%A8te_(informatique)) et l'exécution du bytecode (code intermédiaire plus proche des instructions machines que le code source) en code natif.

1. Présentation de NoSQL

Présentation rapide :

Le noSQL est une catégorie de *SGBD* non-relationnelle et utilisé par de nombreux acteurs de l’informatique tels que Google, Amazon.com, Facebook ou eBay.

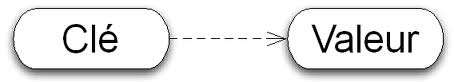
Il existe plusieurs architectures de base de données noSQL, mais la plus courante et la plus simple est le système de clé-valeur.

Les données sont simplement représentées par un couple clé/valeur. La valeur peut être une simple chaîne de caractères, un objet sérialisé… Les valeurs sont ensuite retrouvées comme dans une table de hachage grâce à sa clé.

Le *SGBD* choisit pour le projet est de type document, c'est-à-dire qu’il ajoute au modèle clé-valeur l’association d’une valeur à structure libre, c’est-à-dire qui nécessiterait un ensemble de jointures en logique relationnelle.

La structure orientée document permet :

* Ajout, modification, lecture ou suppression de seulement certains champs dans un document.
* Indexation de champs de document permettant ainsi un accès rapide sans avoir recours uniquement à la clé.
* Requêtes élaborées pouvant inclure des prédicats sur les champs.



Pourquoi le noSQL ?

* Simplicité : pas de tables à créer, pas de conception préliminaire et rajout de nouveaux types de valeurs sans problèmes.
* Performance : la complexité pour accéder à une valeur dans la base de donnée est presque de O(1) (constante).
* Montée en charge élevée : Même en rajoutant un grand nombre de valeurs, la base de données reste performante et fiable.

Choix du SGBD : MongoDB : 

MongoDB est un SGBD noSQL libre développé depuis 2007 par 10gen. Il permet de manipuler des objets structurés au format BSON (JSON binaire).

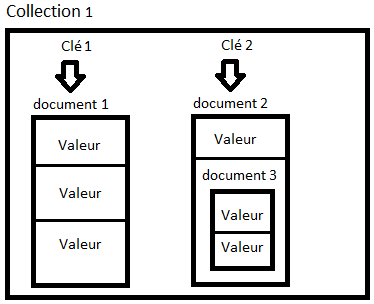
Concrètement les données prennent la forme de documents enregistrés dans des collections, une collection pouvant ainsi contenir un nombre quelconque de documents.

Si on souhaite faire une comparaison avec les bases de données SQL, on peut dire que les collections sont l'équivalent des tables, et que les documents sont l'équivalent des enregistrements.

Cependant il y a une différence fondamentale entre les deux approches : si les enregistrements d'une table SQL et les documents d'une collection MongoDB se composent tous deux de champs, dans les bases SQL chaque enregistrement de la table contient exactement les mêmes champs, dont seul le contenu varie.

A contrario dans une collection MongoDB, les documents peuvent avoir des champs totalement différents.

De plus, il est possible d'imbriquer des documents. (Un champ peut contenir plusieurs champs).



Pourquoi MongoDB ?

- Libre et gratuit : idéal pour un projet étudiant.

- Simplicité et documentation : MongoDB possède une large documentation et une *API* simple à utiliser, permettant de vite programmer et de pouvoir faire reprendre la gestion de la base de données par d’autres personnes.

- Performance et amélioration : la structure orienté document permet d’avoir de très bonnes performances même dans le cas d’une complexification ultérieure de la base de données.

### http://i1-news.softpedia-static.com/images/news2/Eclipse-IDE-Evolves-into-a-First-Class-Windows-7-Citizen-2.png4) L’environnement de développement (Eclipse)

Eclipse est un environnement de développement (IDE)  libre, extensible, universel et polyvalent. Ceci permet donc au programmeur de disposer de différents outils et d’une interface pour développer plus facilement et rapidement des applications.

Nous avons choisi Eclipse pour plusieurs raisons :

* Tout d’abord, c’est un outil que nous avons utilisé durant notre cursus, on est donc familiarisé avec son interface et ses fonctionnalités et sûr de sa fiabilité.
* Il est libre, gratuit et dispose d’un plugin Java.
* Il permet de générer une Javadoc d’une très bonne qualité en quelques clics.
* Il permet l’auto-complétion, la coloration et correction du code en direct, de compiler et d’exécuter plus simplement que dans un terminal et d’accéder plus rapidement aux erreurs dans le code à l’aide de lien dans la console. Il permet aussi de générer des codes génériques tel que le ‘try/catch’ pour la gestion des exceptions ou les importations de librairies.
* Il permet aussi la gestion de projet, paquetage et l’importation de librairies supplémentaires en quelques clics.

