HEIG-VD

Projet : Génie Logiciel

Rapport intermédiaire

Righitto Simone, Saam Frédéric, Fröhlich Magali, Melly Calixte

29/04/2014

Fonctionnement générale de l'application

Le but de l'application développée dans le cadre du projet GEN est de recréer le jeu Pictionary. Cette application sera élaborée en vue d'une utilisation sur un réseau local car c'est un jeu par définition multi-joueurs.

Pour rappel, le jeu traditionnel se joue par équipe. A chaque tour, un joueur tente de faire deviner un mot à l'aide d'un dessin si son équipe trouve le mot alors elle gagne un point. Le temps pour effectuer le dessin est limité. De plus, le joueur n'a pas le droit d'écrire le mot à faire deviner. Après un certain nombre de tour, la partie est finie et on compte les points pour désigner la meilleure équipe.

Notre objectif de base est de fournir une application qui doit pouvoir :

- Gérer des parties sur un serveur. Cela comprend une gestion des équipes et donc des joueurs connectés.
- Offrir une interface graphique pour dessiner un mot tiré au hasard.
- Minuter le temps pour le dessin.
- Afficher un chat simple pour afficher les propositions de mots.
- Stocker un dictionnaire qui sera utilisé pour sélectionner un mot.
- Gérer les scores.

Il y aura deux modes de jeux, soit par équipe soit tous contre tous. Commençons par expliquer le premier cas.

Un joueur pourra soit créer une partie, soit en rejoindre une. Le joueur créant une partie pourra spécifier le nombre de tours pour la partie et les catégories qu'il veut.

Les autres joueurs pourront rejoindre une partie créée en indiquant l'IP du serveur et rejoindre une des deux équipes (bleu ou rouge). Une fois tous les joueurs dans la partie, le créateur de la partie pourra décider de commencer la partie. Ensuite une des équipes sera tirée au sort pour savoir qui va commencer à jouer. Au sein de l'équipe qui commence à jouer, il y a un autre tirage au sort pour désigner qui sera le dessinateur. Le dessinateur recevra un message pour lui avertir de son rôle et le mot qu'il doit dessiner. Le dessinateur aura accès au dessin et non au chat tandis que les « devineurs » pourront écrire dans le chat et voir aussi les propositions de l'équipe. Une fois que le dessinateur commence à dessiner, les autres joueurs de la partie (les deux équipes) vont voir le dessin. L'équipe qui commence pourra utiliser le chat et aura 1 minute pour trouver le mot si elle n'y arrive pas l'autre équipe, qui pendant cette minute à son chat désactivé, pourra elle aussi proposer des solutions, à l'aide du chat qui se sera réactivé pour trouver le mot. La partie se termine une fois que le mot a été trouvé ou que le temps général (pour les deux équipes) qui est de 2'30 est écoulé. Ensuite les équipes vont s'échanger de rôle. La partie s'arrête lorsque le nombre de tour a été atteint.

Dans le deuxième cas, tous les joueurs verront le dessin et tous les joueurs pourront utiliser le chat. Il y aura juste un tirage au sort pour savoir qui va être dessinateur.

Les utilisateurs pourront lors du lancement de l'application soit choisir de se connecter, si ils n'ont pas de compte ils pourront en créer un, soit d'être « anonyme ». Les joueurs connectés l'avantage d'une gestion de score à l'aide d'une base de donnée, tandis que les utilisateurs anonyme n'auront pas la fonctionnalité des scores.

Côté serveur, l'administrateur pourra gérer le dictionnaire (ajouter, modifier, supprimer des mots). Il pourra aussi voir les parties en cours.

L'application sera développée en vue d'une utilisation dans un réseau LAN. Ce qui implique que l'adresse IP du serveur sera connue par les utilisateurs qui se connecteront à l'application. On pourrait imaginer un mécanisme où l'application cliente se connecte sur un serveur web par une URL et qui en suite serait redirigée vers un serveur mais cela ne correspond pas à notre environnement de développement.

Certaines règles de jeu pourraient être facilement ignorées.

Communication client-serveur

Le rôle du serveur est de gérer les connexions des joueurs et de gérer les parties en cours. Le rôle de l'application client est de fournir une interface pour que l'utilisateur puisse dessiner et utiliser le chat. Le serveur retransmet les informations aux applications clientes.

Lancement du jeu : Les joueurs se connectent au serveur. Une fois connectées, ils ont le choix de créer une partie ou d'en rejoindre une déjà créée.

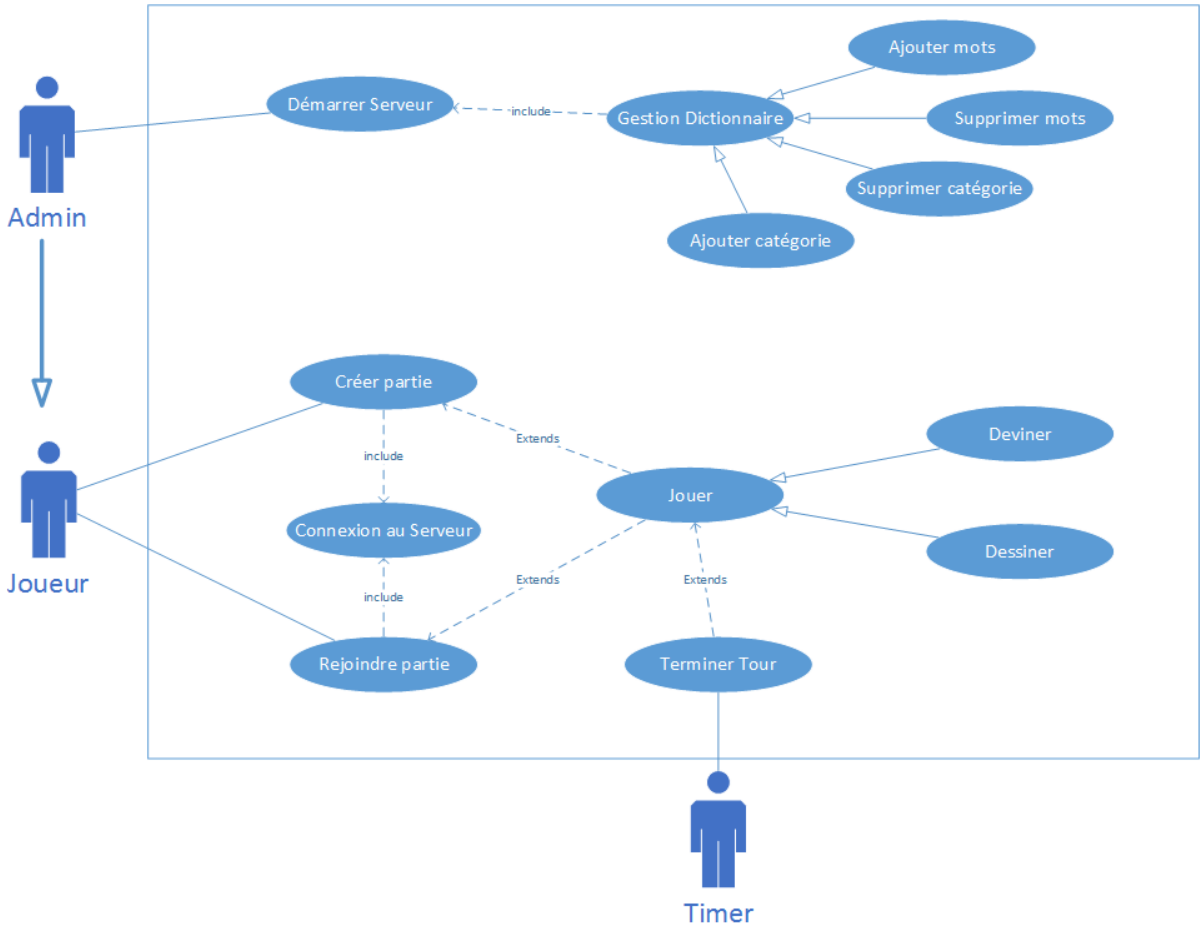
Fin du jeu : Lorsque le nombre de tours est écoulé, la partie prend fin. L'utilisateur est ramené à l'écran d'accueil qui permet de créer une partie ou d'en rejoindre une.

Le serveur aura un rôle de dieu, il sera tout puissant et c'est lui qui enverra les signaux pour contrôler les joueurs (début et fin de partie). Sauf quand un client trouvera le mot c'est lui qui va envoyer un signal au serveur pour lui dire qu'il peut envoyer un signal de fin aux autres.

Cas d'utilisation

Diagramme de contexte

Voici le fonctionnement général de l'application :



Description des acteurs

Acteur principal : Joueur

- Le joueur de l'application peut démarrer une partie ou rejoindre une partie. Pour cela, il doit obligatoirement être connecté au serveur.
- Le jouer peut avoir deux rôles : Soit dessiner un mot tiré du dictionnaire soit deviné le mot tiré du dictionnaire et faire des propositions sur le chat.

Acteur secondaire : Timer, Administrateur

- Le Timer met fin au tour d'une partie. Une partie est composée de plusieurs tours.
- L'Administrateur peut gérer les mots proposés au joueur et les classer par catégorie.

Scénarios principales de succès

Démarrer Serveur

1. Le serveur démarre
2. Le serveur attend les connexions

Gestion Dictionnaire

1. L'administrateur choisi une opération à faire (Ajouter mots, Supprimer mots, Ajouter catégories, Supprimer catégories)

Ajouter mots

1. L'administrateur saisie la catégorie du mot
2. L'administrateur saisie le mot à ajouter
3. Le mot est ajouté dans la base de données

Supprimer mots

1. L'administrateur saisie la catégorie du mot
2. L'administrateur saisie le mot à effacer
3. Le mot est éliminé de la base de données

Ajouter catégorie

1. L'administrateur saisie le nom de la catégorie
2. La catégorie est ajoutée dans la base de données

Supprimer catégorie

1. L'administrateur saisie le nom de la catégorie
2. La catégorie (avec tous les mots contenue) est supprimée de la base de données

Connexion au Serveur

1. Soit l'utilisateur se connecte en anonyme
2. Soit l'utilisateur saisit son user et mot de passe pour se connecter avec son compte
3. Soit l'utilisateur doit créer un nouveau compte

Créer partie

1. Le joueur se connecte au serveur
2. Le joueur saisie les informations sur la partie à créer
3. La partie est crée

Rejoindre partie

1. Le joueur se connecte au serveur
2. Le joueur se connecte à une partie existante sur le serveur

Jouer

1. Le joueur se voit attribuer un rôle (dessiner / deviner)
2. Le joueur joue son rôle tant que la partie n'est pas terminée

Dessiner

1. Le joueur se voit attribuer un mot à dessiner
2. Le joueur dessine tant que le système n'indique pas le contraire
3. Fin du tour

Deviner

1. Le joueur voit le dessin dans la zone spécifique
2. Le joueur envoie une réponse
3. Les étapes 1 et 2 se répète jusqu'à la fin du tour.

Terminer Tour

1. Le timer fige le dessin lorsqu'il arrive à 1 min
2. Le timer met fin au tour lorsqu'il arrive à 2min 30

Scénario alternatif

Démarrer Serveur

1. Le serveur démarre
    a. Le serveur est déjà démarré
    b. Le port spécifié n'est pas disponible :
        i. Changer de port et redémarré
        ii. terminer les opérations de démarrage

Gestion Dictionnaire

1. Soit ajouter des mots au dictionnaire
2. Soit supprimer des mots au dictionnaire
3. Soit ajouter une catégorie
4. Soit supprimer une catégorie

Ajouter mots

1. L'administrateur saisie la catégorie du mot
    a. La catégorie n'existe pas dans la base de données :
        i. créer une nouvelle catégorie
        ii. retourner à la saisie (1.)
        iii. retourner à « Gestion Dictionnaire »
2. L'administrateur saisie le mot à ajouter
    a. Le mot existe déjà dans la base de données :
        i. afficher un message d'avertissement et retourner à la saisie
        ii. Terminer l'opération « Ajouter mots » et retourner à « Gestion Dictionnaire »
3. Le mot est ajouté à la base de données
    a. Une erreur dans la communication avec la base de données est survenue :
        i. Afficher un message d'erreur

Supprimer mots

1. L'administrateur saisie la catégorie du mot
    a. La catégorie n'existe pas dans la base de données :
        i. afficher un message d'erreur et retourner à la saisie de la catégorie (1.)
2. L'administrateur saisie le mot à effacer
    a. Le mot n'existe pas dans la base de données :
        i. afficher un message d'erreur
3. Le mot est enlevé de la base de données
    a. Une erreur dans la suppression est survenue :
        i. Afficher un message d'erreur

Ajouter catégorie

1. L'administrateur saisie la catégorie
    a. La catégorie existe déjà dans la base de données :
        i. afficher un message d'avertissement et retourner à la saisie
        ii. Terminer l'opération « Ajouter catégorie »

Supprimer catégorie

1. L'administrateur saisie le nom de la catégorie
    a. La catégorie n'existe pas dans la base de données :
        i. afficher un message d'erreur et retourner à la saisie de la catégorie (1.)
2. La catégorie (avec tous les mots contenue) est supprimé de la base de données
    a. Une erreur dans la suppression est survenue :
        i. Afficher un message d'erreur

Connexion au Serveur

1. Soit l'utilisateur se connecte en anonyme
    a. La connexion échoue
        i. Afficher un message d'erreur et retourner à « Connexion au Serveur »
2. Soit l'utilisateur saisit son user et password pour se connecter avec son compte
    a. Le user / password ne sont pas correctes
        i. Afficher un message d'erreur et retourner à « Connexion au Serveur »
3. Soit l'utilisateur doit créer un nouveau compte
    a. Le pseudo n'est pas disponible (déjà présent dans la base de données)
        i. Afficher un message d'avertissement et retourner à la saisie
    b. L'utilisateur n'as pas rempli tous les champs obligatoires
        i. Afficher un message d'avertissement et retourner à la saisie

Créer partie

1. Le joueur saisie les informations sur la partie à créer
    a. Les informations ne sont pas acceptées par le serveur
        i. Afficher un message d'erreur :

            i_1. Retourner à la saisie (2.)

            i_2 Terminer l'application

2. L'utilisateur devient administrateur de la partie.
3. Le serveur attend d'autres connexions pour permettre le lancement de la partie.
4. L'utilisateur lance la partie une fois les autres joueurs connectés.

Rejoindre partie

1. Le joueur se connecte à une partie existante sur le serveur
2. Si l'utilisateur a sélectionné une partie en équipe il choisit son équipe.
3. La partie commencera lorsque l'utilisateur qui a créé la partie la lancera.
    a. L'administrateur de la partie peut quitter la partie avant de la lancer.

Jouer

1. Le joueur se voit attribuer un rôle (dessiner / deviner)
2. Le joueur joue son rôle

Dessiner

1. Le joueur se voit attribuer un mot à dessiner
    a. Le joueur ne connait pas le mot
        i. Il peut demander au serveur un nouveau mot (max 2 change)
2. Le joueur dessine
3. Fin du tour

Deviner

1. Le joueur voit le dessin dans la zone de dessin
2. Le joueur envoie une réponse
    a. La réponse n'arrive pas au serveur
        i. Renvoyer le message avec la réponse
3. Fin du tour

Terminer Tour

Le timer met fin au tour

Protocole de communication entre le client-serveur

Ce chapitre détaille la liste des messages échangés entre l'application cliente et l'application serveur. Le serveur et le client doivent communiquer entre eux pour les cas suivants :

Le joueur se connecte au serveur

1. Le joueur envoie un message au serveur pour lui annoncer une nouvelle connexion. (Les messages échangés ici seront les messages liés au protocole TCP).
2. Le joueur envoie un message pour indiquer s'il veut s'enregistrer auprès du serveur ou s'il veut se connecter en tant qu'utilisateur déjà inscrit.
3. Le joueur envoie en suite un message texte contentant son pseudo et son mot de passe.
   Le joueur à la possibilité d'envoyer un message qui indique qu'il souhaite se connecté en tant qu'utilisateur anonyme, il devra néanmoins indiquer un pseudo.

Ecran d'accueil du serveur pour créer une partie

1. Le serveur envoie à l'application cliente la liste des parties créées sur le serveur. Le serveur envoie plusieurs messages contenant le nom de la partie et le nom du joueur qui l'a créé.
2. Le joueur envoie un message au serveur pour indiquer s'il veut créer une partie ou s'il veut rejoindre une partie.
3. Le joueur envoie un message texte au serveur indiquant les informations liées à la partie sélectionnée.

Création d'une partie

1. Le joueur envoie le nom de la partie au serveur.
2. Le serveur envoie au joueur le nom des joueurs inscrits.
3. Le joueur envoie un message pour lancer la partie.

Rejoindre une partie

1. Le joueur envoie le nom de la partie sélectionnée au serveur.
2. Le serveur envoie le pseudo du joueur à l'application cliente.

Dessiner durant un tour

1. Le serveur envoie un mot tiré au hasard à un joueur au hasard.
2. Le joueur qui a reçu le mot envoie un message au serveur pour commencer le tour.
3. Le serveur démarre le timer chez les autres joueurs.
4. Le joueur commence son dessin qui est retransmis au serveur.
5. Le serveur envoie le dessin aux autres joueurs.

Deviner le mot d'un tour

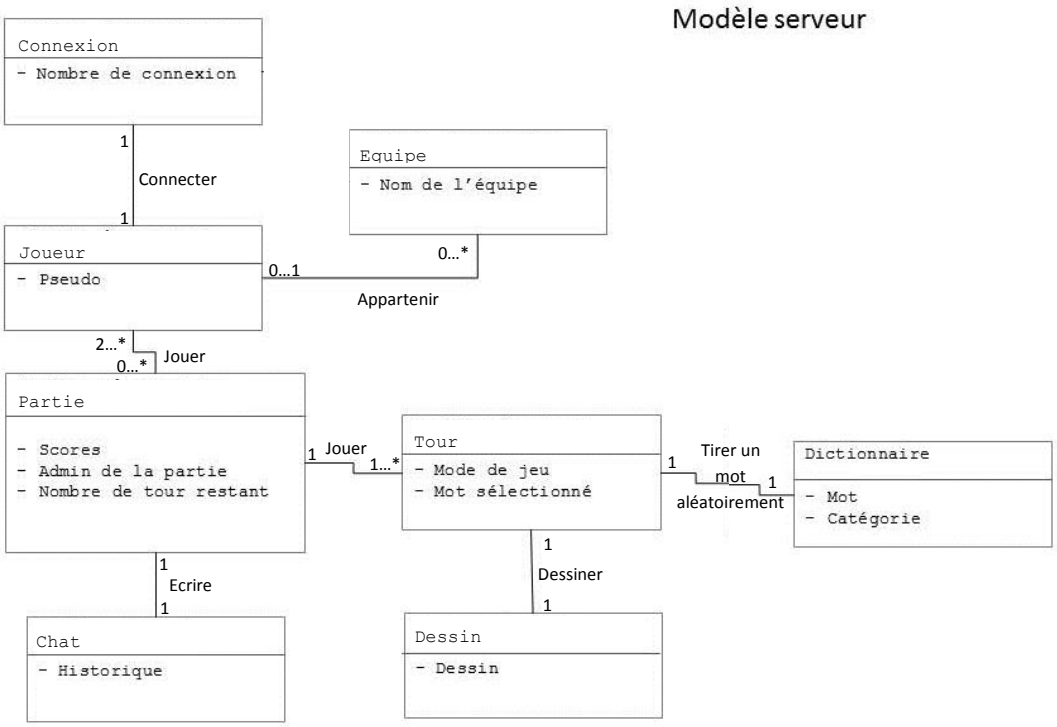
1. Le joueur reçoit le départ du timer.
2. Le joueur fait ses propositions sur le chat, le joueur envoie un message texte au serveur.
3. Le serveur retransmet le message aux autres joueurs de la partie.
4. Si une réponse est correcte le serveur envoie un message aux autres joueurs pour annoncer la fin du tour.

Fin du tour du timer

1. Lorsque le timer du serveur arrive à 1 minute le serveur envoie un message d'arrête au joueur qui dessine.
2. Lorsque le timer du serveur arrive à 2 minutes 30 alors le serveur envoie un message aux autres joueurs pour mettre fin au tour.

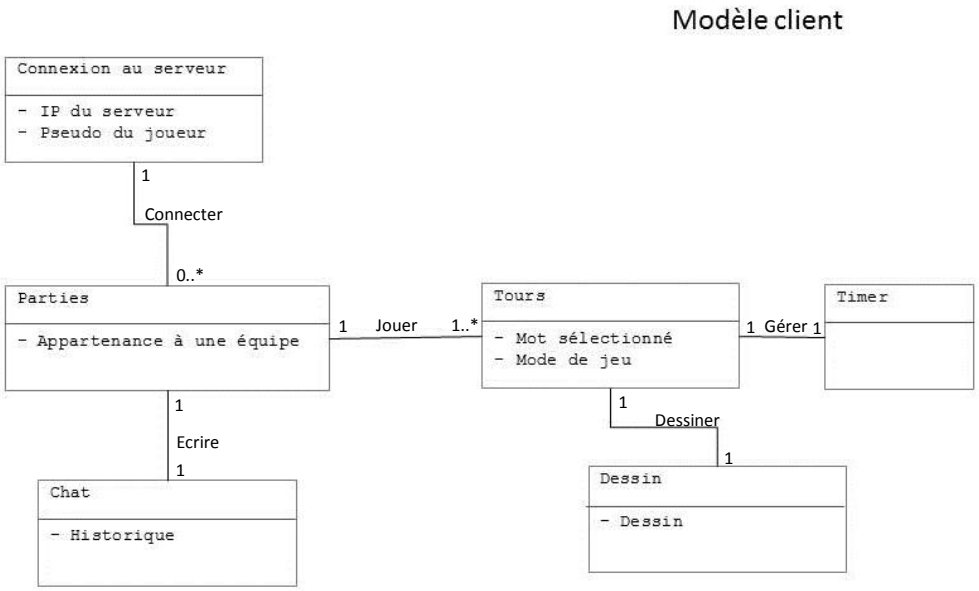
Modèle de domaine

Ce modèle de domaine décrit le fonctionnement de l'application serveur.



- Connexions : Gère les connexions des joueurs.
- Equipes : Contient le nom de l'équipe et la liste des joueurs appartenant à l'équipe. L'équipe n'est valable que pour une partie.
- Joueurs : Contient le pseudo du joueur.
- Parties : Contient les scores, le nombre de tours qu'ils restent avant la fin de la partie, ainsi que les informations pour retrouver l'administrateur du jeu.
- Tours : Ainsi qu'un mot sélectionné par tour. Ici, le mode de jeu sélectionné représente le joueur qui effectue le dessin.
- Dictionnaire : Base de données contenant les mots ainsi que les catégories à disposition.
- Timer : Contient le nombre de minutes restantes, ainsi que les secondes s'y rapportant.
- Chat : Zone de discussion, contient l'historique des messages.
- Dessin : Gère la transmission du dessin.

Ce modèle de domaine décrit le fonctionnement de l'application cliente.

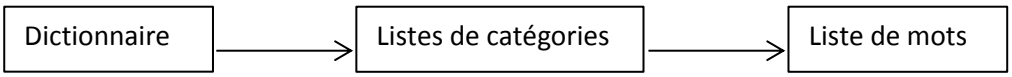


- Connexion au serveur : Contient l'IP du serveur ainsi que le pseudo du joueur.
- Parties : Contient l'équipe du joueur.
- Chat : Zone de discussion, contient l'historique des messages.
- Tours : Contient le mode de jeu qui indique si c'est au tour du joueur de dessiner. Dans ce cas, il y a également un mot sélectionné par tour.
- Dessin : Zone de dessin.

Base de donnée et conception du dictionnaire

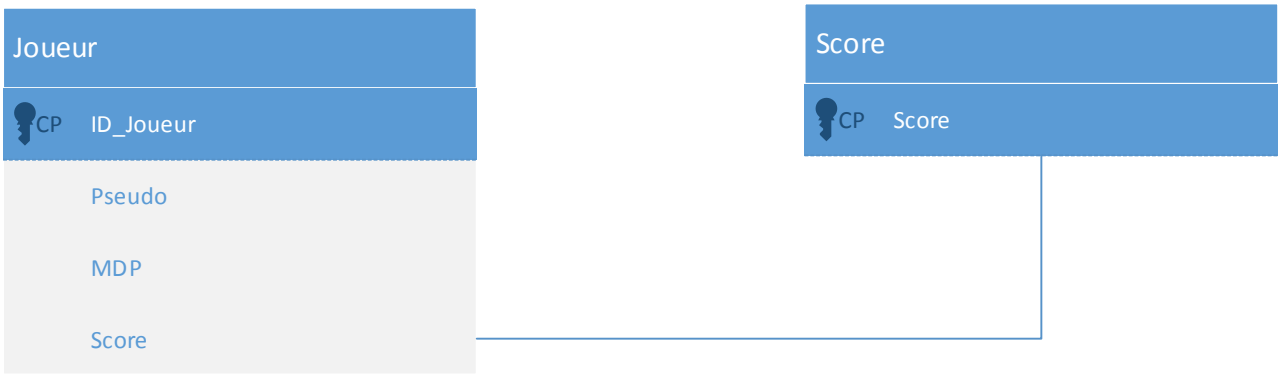
Pour le fonctionnement de l'application, il faut tirer un mot aléatoirement issu du dictionnaire. Les mots tirés au hasard sont classés par catégorie et le joueur peut choisir la catégorie pour son tour.

La structure de donnée à mettre en place n'est pas particulièrement complexe à mettre en place. Les données seront donc stockées dans un fichier XML.



Les joueurs inscrits sur le serveur seront enregistrés sur dans une base de donnée. Les joueurs enregistrés pourront publier leurs scores et les consulter.

Base de donnée pour les scores



Groupe de développement

Il y a 4 personnes qui composent le groupe de développement.

Rôle standard	Responsabilités	Responsable
Représentants des utilisateurs	• Collecte des besoins pour le projet. • Spécifications des tests de fonctionnalités • Explication des aspects métiers	Melly Calixte
Chef de projet	• Planification • Coordinations avec les utilisateurs	Fröhlich Magali
Analyste	• Spécifications • Collecte des demandes de changement	Righitto Simone
Architecte, concepteur en chef	• Conception de l'architecture du produit	Saam Frédéric, Fröhlich Magali
Programmeur	• Participe à la conception du produit • Ecrit les tests unitaires • Codage	Righitto Simone, Saam Frédéric, Fröhlich Magali, Melly Calixte
Responsable des tests	• Participe à l'intégration continue des composants • Ecrit les tests fonctionnels • Met en place l'architecture permettant de lancer régulièrement les tests fonctionnels	Righitto Simone
Responsable de la configuration	• Gestion de la base des artefacts du projet • Gestion des releases • Allocation des droits • Responsable de la configuration (logicielle & matérielle) • Intégration des changements	Saam Frédéric

Itération

Semaine	Objectif et déroulement
28 avril 2014	Rendu du rapport intermédiaire
05 mai 2014	Objectif : Connexion au serveur, Dictionnaire Le but de cette itération est de permettre à l'administrateur de démarrer le serveur et de le configurer. En parallèle, les fichiers XML qui constituent le dictionnaire seront développés. Le serveur pourra gérer en suite le dictionnaire. Righitto Simone : Analyse et création du fichier XML pour la gestion des mots qui composent le dictionnaire. Saam Frédéric : Conception du serveur et de l'application cliente. Fröhlich Magali : programmation des classes pour la gestion du dictionnaire sur le serveur. Melly Calixte : Programmation des classes de base pour l'application cliente. Fonctionnalités produites à la fin de l'itération : Démarrer le serveur et gérer le dictionnaire. Gérer le dictionnaire correspond aux actions : ajouter, modifier, supprimer un mot ou une catégorie.
12 mai 2014	Objectif : Gestions des joueurs Le but de cette itération est de pouvoir gérer l'enregistrement des joueurs sur le serveur ainsi que leur authentification à travers la base de données. Righitto Simone : Programmation des classes pour la gestion des joueurs sur le serveur. Saam Frédéric : Conception du modèle observable-observé pour préparer les interfaces graphiques. Programmation des interfaces du côté serveur. Fröhlich Magali : Programmation des classes pour gérer les connexions des joueurs au serveur. Melly Calixte : Programmation de la base de données et des interfaces graphiques côtés client. Fonctionnalités produites à la fin de l'itération : Gérer les joueurs correspond à la possibilité de supprimer un joueur et d'afficher la liste des joueurs enregistrés sur la base de données.
19 mai 2014	Objectif : Création d'une partie, Rejoindre une partie Le but de cette itération est de pouvoir gérer la création des parties par un joueur sur le serveur. Les joueurs doivent pouvoir se connecter au serveur pour rejoindre une partie créée. Righitto Simone : Programmation des classes pour la gestion des parties : Création des parties et Rejoindre une partie. Saam Frédéric : Conception des classes pour la gestion des parties. Fröhlich Magali : Gestion des connexions et programmation des messages échangés pour la création de la partie. Cela comprend également la possibilité pour le joueur de rejoindre une partie. Melly Calixte : Programmation des classes pour la gestion des parties : Rejoindre une partie. Fonctionnalités produites à la fin de l'itération : Les joueurs peuvent se connecter au serveur et gérer des parties. Cela comprend la partie authentification, la création d'une partie et la possibilité de rejoindre une partie. Dans un premier temps, on ne considérera que le mode sans équipe.

26 mai 2014	Objectif : Dessiner, Chat Le but de cette itération est de pouvoir voir l'interface finale de l'application cliente. Il doit être possible d'effectuer un dessin sur l'application graphique. Righitto Simone : Programmation de l'interface graphique générale. Saam Frédéric : Conception des classes pour le dessin et le chat. Fröhlich Magali : Programmation de l'interface graphique pour dessiner. Melly Calixte : Programmation de l'interface graphique pour écrire dans le chat. A la fin de cette itération, il sera possible dessiner et d'écrire un message dans le chat. On ne considérera que le mode sans équipe.
02 juin 2014	Objectif : Dessiner, Chat Le but de cette itération est de pouvoir utiliser l'interface graphique de l'application cliente pour effectuer un dessin retransmis sur les autres applications clientes. Il doit être également possible d'utiliser le chat entre les différentes applications. Righitto Simone : Programmation de l'affichage des scores sur le serveur. Saam Frédéric : Conception et programmation du Timer et de la gestion des scores. Fröhlich Magali : Programmation des échanges pour afficher le dessin Melly Calixte : Programmation des échanges pour afficher le chat A la fin de cette itération, il sera possible dessiner et d'écrire un message dans le chat. Ces données seront retransmises aux autres applications clientes. Il sera également possible d'afficher les scores sur le serveur. On ne considérera que le mode sans équipe.
09 juin 2014	Objectif : Gérer les tours Le but de cette itération est de pouvoir utiliser l'application cliente/serveur pour jouer une partie. Il doit être possible de tirer un mot du dictionnaire aléatoirement pour qu'un utilisateur le dessine. Righitto Simone : Test générale de l'application. Saam Frédéric : Programmation des messages échangés entre le timer et les applications. Fröhlich Magali : Programmation des messages échangés pour la gestion des équipes. Melly Calixte : Test générale de l'application. A la fin de cette itération, il sera possible de jouer une partie en équipe ou en mode tous contre tous.
16 juin 2014	Objectif : Conception final de l'application. A présent, le serveur doit pouvoir gérer plusieurs connexions pour gérer plusieurs parties en même temps. Righitto Simone : Découverte réseau. Saam Frédéric : Gestion de plusieurs parties lancées en même temps sur le serveur. Fröhlich Magali : Documentation. Melly Calixte : Test final de l'application. A la fin de cette itération, le développement de l'application sera terminé.
23 juin 2014	Présentation et rendu final