Document Architecture Logiciel

Version <1.0>

FileFinder

Historique des révisions

Date	Version	Description	Auteur
19/5/2015	1.0		El koutbi Saad

Table des matières

1. Introduction

Ce document présente l'organisation générale du logiciel et les modèles de conception mis en oeuvre pour la réaliser.

2. Objectif du logiciel

Notre logiciel a pour principal objectif la recherche de localisation des rousources numerique qui se trouve dans ses supports de stockage.

2.1 Contexte

Le logiciel doit permettre lister les fichier des support d'un utillisateur ,ainsi l'utillisateur en se conn

2.2 Besoins fonctionnels

Notre application a pour principalefonctionalité :

la recheche de la localisation des resources numeriques.

Ajout suppression de support de stockage.

partager les resultat de recherche entre utilisateur.

Syncronisation des resource.

2.3 Besoins non fonctionnels

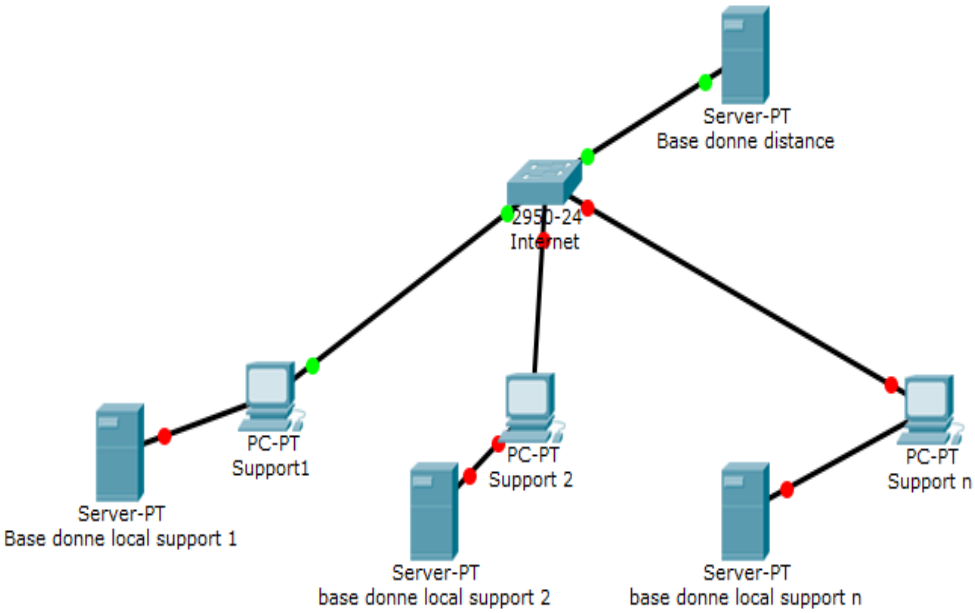
Ne sont présentés ici que les besoins importants pour l'architecture du logiciel:

Utilisabilité - Préférences : Les outils doivent posséder un menu Préférences dans lequel on peut au moins choisir les type de fichier a ne pas listée.

Performances - lancement : L'outil doit se lancer rapidement.

Supportabilité; l'application d'oi etre superter par n'importe qu'elle systeme d'exploitation.

3. Structure



3.1 Vue des couches

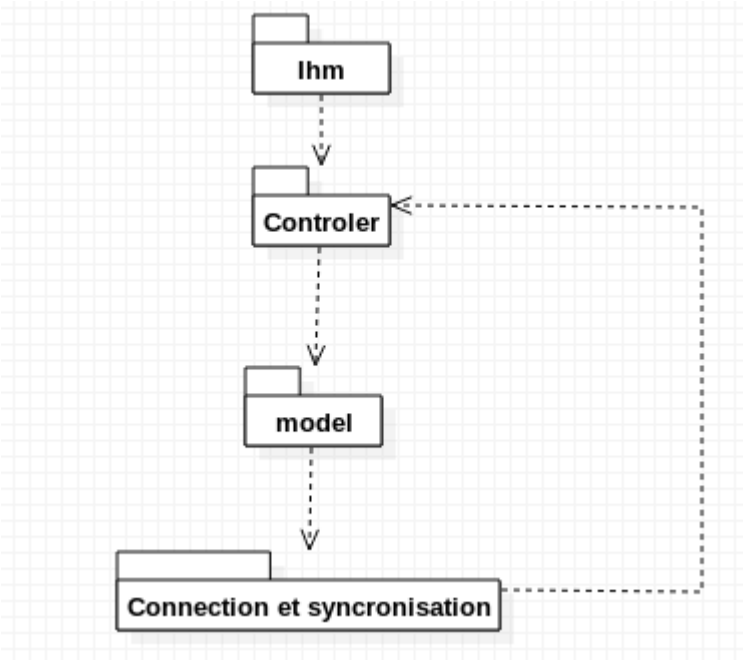
La couche Interface contient toutes les fenêtres et objets graphiques utilisés dans l'application. Elle intègre aussi les contrôleurs liés à l'interface, puisque aucune couche spécifique n'apparaît dans le logiciel.

La couche Application contient les classes effectuant les traitements de l'application. Ces classes gèrent aussi les règles de cohérence de l'application.

La couche Domaine contient les classes métiers qui utilisent leurs règles de gestion spécifiques. La couche Infrastructure contient les classes permettant de réutiliser certains composants, dont les classes chargées de gérer la persistance.

Dans ce cadre, c'est la couche application et non de celle des données qui accède à la couche infrastructure : les classes de traitement s'occupent de sauvegarder les données , ce qui permet de séparer la définition des données de leur mode de stockage.

3.2 Sous-systèmes et paquetages



4. Comportement

4.1 Réalisation des cas d'utilisation

L'utilisateur en recherchant une resource donne le nom du fichier recherche le serveur recherche déjà dans le support lui meme puis selon les cas ou la conextion est presente ou pas va rechercher dans la base donné distance ou sa copie local.

L'utilisateur pour ajouté un support vas installe l'apllication une liste des support qu'il posséde sur se pc apparet il choisi alors ceux qu'il veut partagé et valide l'apllication fait son premier listage de resource et l'envois a la base donné externe.

Achaque fois que l'utilisateur est connecté a l'aide d'un support le listage des et envoyé a la base donnémise ajours sont envoyer aux serveur externe.

A chaque modification dela base donné externe une nouvelle copie est envoyé aux resource conécté.

5. Qualités de l’architecture

Notre architercture a comme avantage que :

Toutes les données sont centralisées sur un seul serveur, ce qui simplifie les contrôles de sécurité, l'administration, la mise à jour des données et des logiciels.

Les technologies supportant l'architecture client-serveur sont plus matures que les autres.

La complexité du traitement et la puissance de calculs sont à la charge du ou des serveurs, les utilisateurs utilisant simplement un client léger sur un ordinateur terminal qui peut être simplifié au maximum.

les serveurs étant centralisés, cette architecture est particulièrement adaptée et véloce pour retrouver et comparer de vaste quantité d'informations (moteur de recherche sur le Web), ce qui semble être rédhibitoire pour le P2P beaucoup plus lent, à l'image de Freenet.

Neaumoin elle comporte les quelques soucie suivant :

Si trop de clients veulent communiquer avec le serveur au même moment, ce dernier risque de ne pas supporter la charge (alors que les réseaux pair-à-pair fonctionnent mieux en ajoutant de nouveaux participants).

Si le serveur n'est plus disponible, plus aucun des clients ne fonctionne (le réseau pair-à-pair continue à fonctionner, même si plusieurs participants quittent le réseau).

Les coûts de mise en place et de maintenance peuvent être élevés.

En aucun cas les clients ne peuvent communiquer entre eux, entrainant une asymétrie de l'information au profit des serveurs.