|  |
| --- |
| UFR Sciences et techniques |
| Rapport du projet multi-modules |
| Interface Homme-machine |
|  |
| **BRAUD Jérémy && BREMARD Nicolas && LUCAS Boris && KROMMENHOEK Cédric** |
| **11/01/2010** |

Sommaire

Introduction 3

Analyse des besoins des utilisateurs 4

Identification des fonctionnalités 4

Scénarios d’utilisation 5

Cas d’utilisation 7

Conception 8

Fenêtre principale 8

Onglet Faire 10

Onglet Traiter 11

Fenêtre de connexion 13

Fenêtre de création de compte 14

Fenêtre de préférence 15

Fenêtre additionnelle de date 16

Implémentation 17

Pattern de conception : le Pattern MVP 17

Architectures 18

Architecture logicielle globale 18

Architecture de l’interface utilisateur 19

Conclusion 20

# Introduction

Dans le cadre du projet multi-modules, nous avons à charge de développer une application qui puisse mettre en œuvre la méthode GTD (Getting Things Done). Nous devons réaliser pour cette matière l’habillage du logiciel, communément appelée Interfaces Homme-Machine (IHM). Dans les autres matières incriminées par ce projet, nous avons défini les besoins métiers de l’application, comme les cas d’utilisations, la façon dont sont représentées les données ou encore la manière de gérer la persistance des données. Nous reviendrons tout de même sur les cas d’utilisation et essaierons de les détailler par des scénarios afin de délimiter les frontières de l’IHM à construire. Nous détaillerons ensuite le processus de développement, du prototypage papier à l’utilisation de patrons de conception pour la réalisation du code source.

# Analyse des besoins des utilisateurs

## Identification des fonctionnalités

Afin de pouvoir définir correctement l’IHM, nous devons d’abord connaitre son fonctionnement et identifier ses fonctionnalités. Après une recherche approfondie, il en a résulté que la méthode GTD, et donc le logiciel, doit pouvoir réaliser les cas d'utilisation suivants :

* Collecter des informations
* Les traiter en tant que tâches à effectuer
* Les organiser
* Les passer en revue régulièrement (si une tâche devient obsolète par exemple)
* Effectuer une ou plusieurs tâches

Néanmoins, nous ne traiterons pas ici le passage en revue des tâches, celui-ci étant plutôt réservé au code métier. Un indicateur pourra être utilisé lorsque cela est le cas, mais c’est tout.

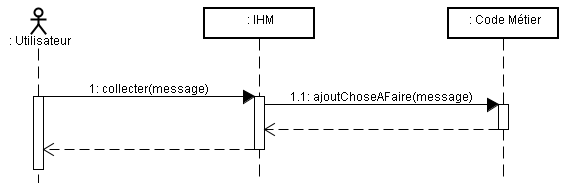
Nous avons aussi décidé que certains cas devaient posséder des contraintes.

* La collecte des informations doit pouvoir s’effectuer à n’importe quel moment
* Plusieurs tâches peuvent faire parties d’un même dossier, appelé projet
* Une tâche peut être dépendante d’une autre, dans ce cas elles sont placées dans un projet qui est dit « ordonné » (la première tâche doit forcément s’effectuer avant la suivante)

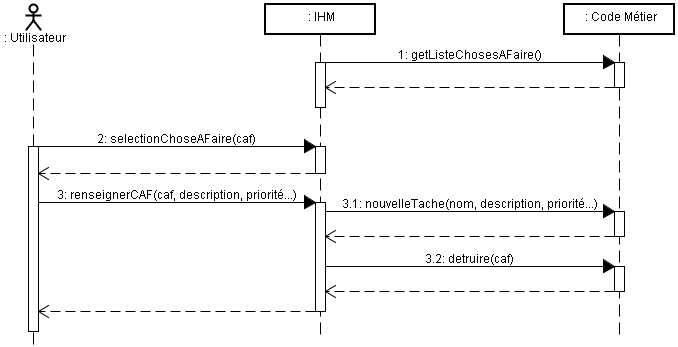
## Scénarios d’utilisation

Essayons de définir quelques scénarios d’utilisation, afin de mieux nous rendre compte de ces différentes fonctionnalités. Les figures suivantes montrent les cas généraux rencontrés lors de l’utilisation du logiciel :

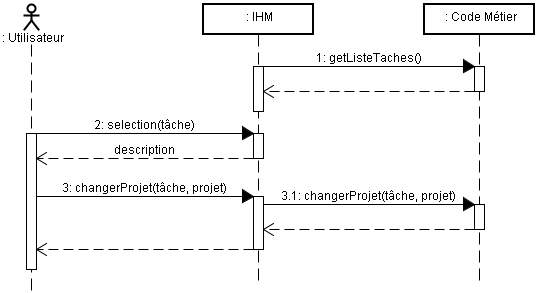
- Collecte d‘une Chose à faire ;



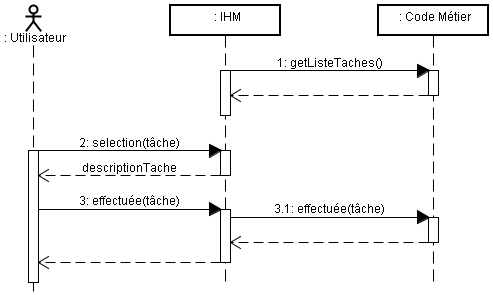
- Traiter une chose à faire ;



- Déplacer une Tâche dans un Projet existant.



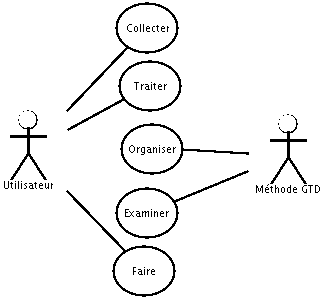
- Faire une Tâche ;



Tous ces scénarios nous permettent de mieux percevoir les actions que l’utilisateur pourra ou non faire à travers l’interface graphique.

## Cas d’utilisation

Nous avons vu, dans la première partie, les fonctionnalités que devaient posséder le logiciel. Puis, dans une seconde partie, les fonctionnalités que l’utilisateur pouvait utiliser ou celles gérer par le système. Nous en avons donc déduit le diagramme des cas d’utilisation.

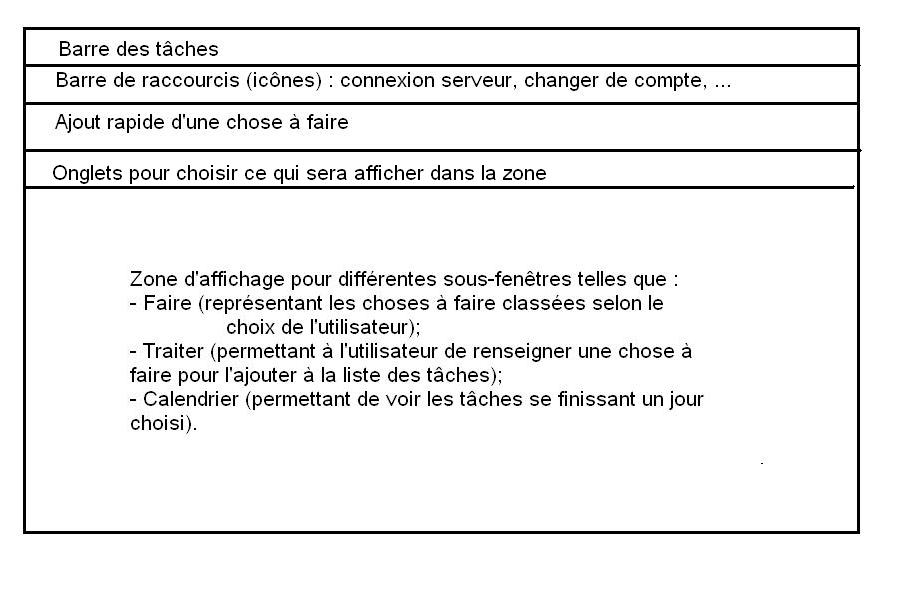


L’analyse des besoins de l’utilisateur et des fonctionnalités du logiciel étant terminée, nous pouvons passer aux détails de la conception de l’IHM.

# Conception

## Fenêtre principale

Cette fenêtre doit permettre à l’utilisateur d’accéder à toutes les fonctionnalités ou aux autres fenêtres donc c’est un élément très important ; c’est pourquoi nous l’avons découpé en différentes zones.



*Figure 3.1 : Prototype papier de la fenêtre principale*

**Barre des tâches**

Elle se décompose en menus. Les menus « communs » tels que Fichier, Edition ou Aide sont implémentés, auxquels nous avons ajouté le menu Serveur.

Tous les menus possèdent des sous-menus :

- Fichier : Nouveau (Compte, Taches, ChosesAFaire…), Déconnexion (qui redirige vers la fenêtre de connexion) et Quitter (déconnecte et sort).

- Edition : Préférences (ouvre une fenêtre permettant la configuration de différents éléments).

- Serveur : Mise à jour (vers et à partir du serveur).

- Aide : Page d’aide, liens Internet, A propos.

Nous avons aussi ajouté des raccourcis claviers « génériques » comme *ctrl+N* pour nouvelle chose à faire.

**Barre des raccourcis**

Un utilisateur inexpérimenté serait obligé d’utiliser les menus ou les raccourcis claviers or ces raccourcis doivent être bien connus pour être réellement efficace ; c’est pourquoi nous avons ajouté cette barre. Elle comportera des images « significatives » de l’action qui sera réalisée quand on cliquera dessus permettant ainsi de guider l’utilisateur.

Nous avons mis dans cette barre les quelques actions qui nous ont parus utiles de pouvoir utiliser n’importe quand d’un seul clic. Ces actions sont regroupées par catégories et séparées par des barres verticales.

Dans une future version, ces icônes seront personnalisables.

**Barre d’ajout rapide d’une chose à faire**

L’utilisateur peut à n’importe quel moment vouloir ajouter une chose à faire ; nous avons donc ajouter cette barre qui ne sera qu’un grand champs texte pour entrer une chose à faire et un bouton « ajout » pour l’ajouter dans la liste des choses à traiter.

**Onglets**

La dernière zone est la plus grande car elle est utilisée pour afficher des listes. Cependant pour un confort de lecture, nous l’avons séparée en plusieurs onglets que l’utilisateur peut sélectionner dans une barre se situant juste au dessus de la zone qui sera modifiée.

**Zone générique**

Cette zone peut représenter plusieurs fenêtres qui auront chacune leurs propres spécificités.

## Onglet Faire

FenetrePrincipaleProto.tiff

*Figure 3.2 : Prototype « en dur » de l’onglet Faire dans la fenêtre principale.*

Cet onglet se sépare en deux parties séparées verticalement car pour afficher des éléments se trouvant dans des dossiers (logiciels de messagerie par exemple ou explorateur de fichiers), il est souvent choisi d’afficher à gauche les différents dossiers.

La partie gauche contient des boutons permettant de sélectionner des tâches à voir en fonction dans une première partie des dossiers et dans une seconde des contextes. On ne peut voir qu’un dossier à la fois, mais on peut regarder plusieurs contextes simultanément car des contextes différents peuvent se produire en même temps (ex : « au téléphone » et « au travail »).

## Onglet Traiter

FenetrePrincipaleOngTraiterProto.tiff

*Figure 3.3 : Prototype « en dur » de l’onglet Traiter dans la fenêtre principale.*

FenetrePrincipaleOngTraiterProto2.tiff

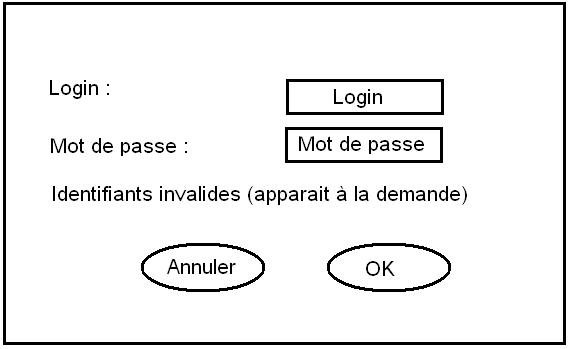
*Figure 3.4 : Prototype « en dur » de l’onglet Traiter dans la fenêtre principale.*

Cette fenêtre est à double usage : elle permet de renseigner les choses à faire pour créer une tâche ou un projet et ensuite de placer ou déplacer les tâches et les projets dans des projets ou sous projets.

Ainsi elle permet d’organiser ses tâches en fonction d’autres tâches ou d’autres projets. Sur la figure 3.3, on peut voir sur la gauche deux projets et une tâche qui sont développés sur la figure 3.4. On peut ainsi remarqué qu’un des projets contient un sous projet.

De même la figure 3.3 permet d’observer l’ajout d’une tâche alors que la figure 3.4 permet d’observer l’ajout de projet.

## Fenêtre de connexion



*Figure 3.6 : Prototype papier de la fenêtre de connexion.*

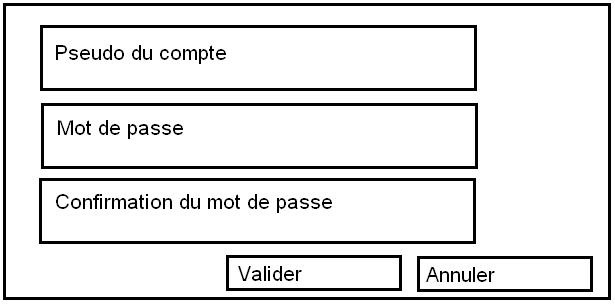
Pour concevoir cette fenêtre, nous nous sommes inspirés des fenêtres de connexions du type des logiciels de messagerie instantanée.

Elle est composée de deux champs texte (login et mot de passe), un texte qui n’apparaît que pour indiquer que les identifiants ont été mal saisis et de deux boutons (Valider et Annuler). Valider permet de vérifier les identifiants, si ils sont valides, l’utilisateur est dirigé vers la fenêtre principale dans lequel son compte a été chargé, sinon il reste sur la fenêtre connexion et le texte d’identifiants invalides apparaît. Annuler permet de quitter le programme.

FenetreConnexion.tiff

*Figure 3.7 : Prototype « en dur » de la fenêtre de connexion.*

## Fenêtre de création de compte



*Figure 3.8 : Prototype papier de la fenêtre de création de compte.*

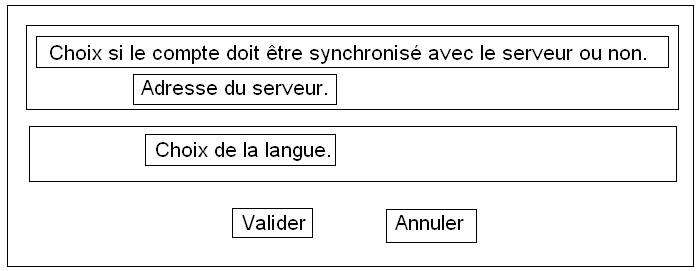
Cette fenêtre permet à un utilisateur qui n’a pas de compte de s’inscrire pour pouvoir utiliser le logiciel.

*FenetreNouveauCompte.tiff*

*Figure 3.9 : Prototype « en dur » de la fenêtre de création de compte.*

## Fenêtre de préférence

L’utilisateur doit avoir la possibilité de fixer certains paramètres de son compte tels que la connexion au serveur ou non.



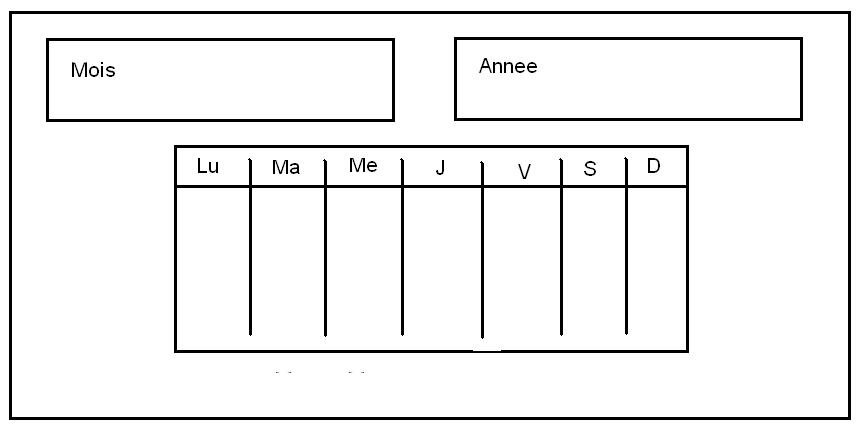
*Figure 3.10 : Prototype papier de la fenêtre de préférences.*

FenetrePreferencesProto.tiff

*Figure 3.11 : Prototype « en dur » de la fenêtre de préférences.*

## Fenêtre additionnelle de date

Basé sur nos expériences personnelles (principalement sur internet), nous nous sommes mis d’accord que pour les choix des dates, il est plus facile d’utiliser un « pseudo calendrier » qui permet dans une mini fenêtre de choisir rapidement une date.



*Figure 3.12 : Prototype papier de la fenêtre du pseudo calendrier.*

FenetreDate.tiff

*Figure 3.13 : Prototype « en dur » de la fenêtre de pseudo calendrier.*

# Implémentation

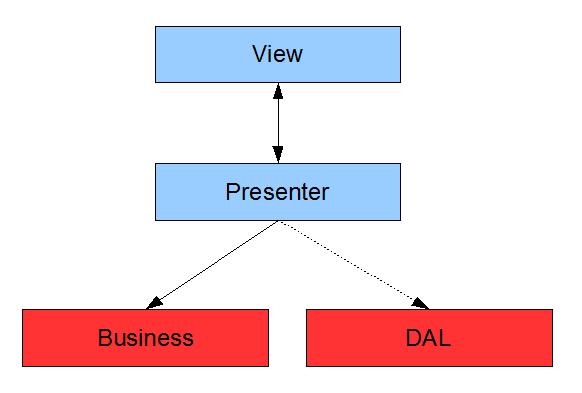
## Pattern de conception : le Pattern MVP

Dans le cadre de l’implémentation de cette IHM, nous avions d’abord pensé à utiliser le patron « classique » Modèle-Vue-Contrôleur.



Cependant après réflexion et implémentation du code-métier, il nous a paru illogique que la Vue puisse modifier directement le Modèle sans contrôles des respects de la méthode « Getting Things Done ». C’est pourquoi nous avons cherché un patron de conception plus adapté à notre cas.

Nous avons ainsi trouvé le patron « Model-View-Presenter ». Ce patron est un dérivé du patron MVC. Il supprime l’interaction entre le Modèle et la Vue et rend les interactions restantes bidirectionnelles.



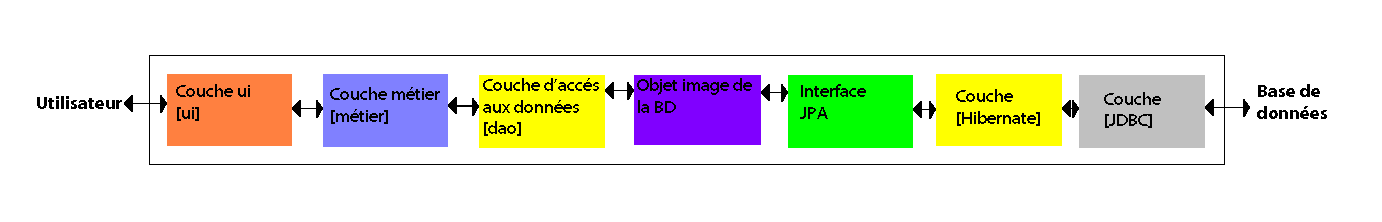
Nous pouvons ainsi gérer toutes les modifications ou interactions entre la Vue (interface utilisateur) et le Modèle (compte).

## Architectures

### Architecture logicielle globale

Le patron de conception utilisé pour l’interface utilisateur nous met déjà sur la voie de l’aspect globale de l’architecture : 1ere couche : Vue.

Les autres couches ont été déterminées lors de la phase d’analyse et se positionnent comme sur le schéma suivant :

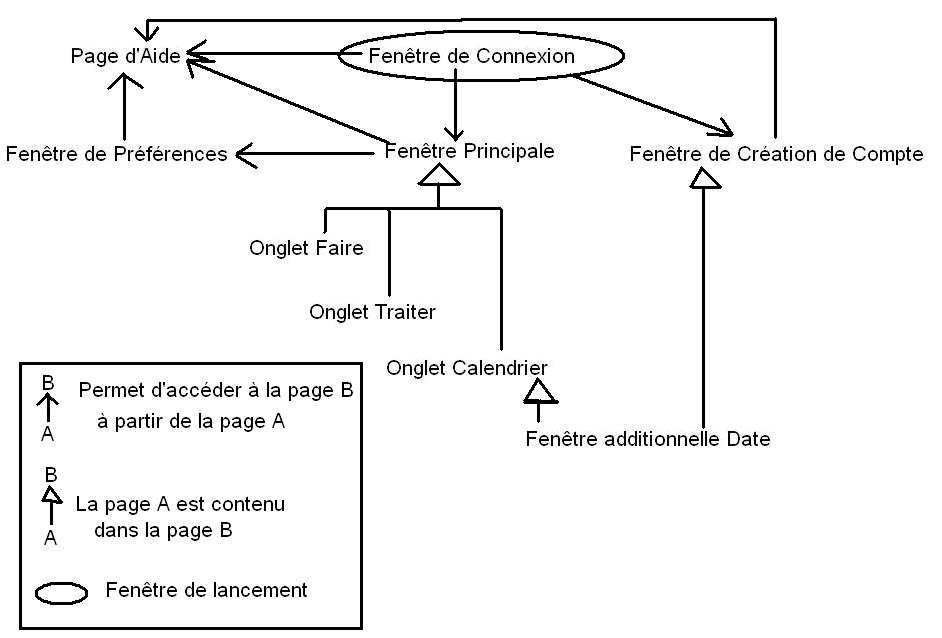


Les couches inférieures ne sont pas visibles de l’utilisateur. Elles servent pour des fonctionnalités du programmes : persistance, stockage …

La couche Métier est la partie Contrôle, la couche UI (User Interface) est la Vue et les autres représente le Modèle (et sa persistance).

### Architecture de l’interface utilisateur

On peut aussi admettre une certaine architecture (hiérarchie ou « organigramme d’interactions ») entre nos fenêtres.



Notre interface comprend aussi bien des fenêtres du logiciel que des pages Internet. Effectivement certaines fonctionnalités, comme l’aide, sont plus facilement et plus communément créées avec des pages HTML. Certains menus auront aussi des liens externes comme les liens vers les tutoriaux vidéos mis en ligne ; le logiciel est ainsi moins volumineux mais nécessite une connexion internet que de toutes manières, il fallait pour la connexion au serveur.

# Conclusion

Ce projet nous aura permit de mettre en œuvre de façon beaucoup plus poussé la création d’IHM ainsi que les interactions entre différents types d’interfaces (fenêtres logicielles et HTML).

Nous avons aussi appris à mieux expliciter nos choix de conception qui se faisaient dans les projets précédents de manière plus implicite/inconsciente comme l’ergonomie ou le choix des couleurs. Ce qui nous rendait le travail difficiles au moment de la justification.

Nous pourrons à l’avenir mieux détaillé les IHM que nous concevrons et mieux les évaluer.