P\_OO



Alexandre Samuel King et Thomas Rey – CID2B

ETML

17 Semaines

Aurelie Curchod

Table des matières

[1 Spécifications 3](#_Toc94509098)

[1.1 Titre 3](#_Toc94509099)

[1.2 Description 3](#_Toc94509100)

[1.3 Prérequis 3](#_Toc94509101)

[1.4 Cahier des charges 3](#_Toc94509102)

[1.4.1 Objectifs et portée du projet (objectifs SMART) 3](#_Toc94509103)

[1.4.2 Caractéristiques des utilisateurs et impacts 3](#_Toc94509104)

[1.4.3 Fonctionnalités requises (du point de vue de l’utilisateur) 3](#_Toc94509105)

[1.4.4 Travail à réaliser par l'apprenti 4](#_Toc94509106)

[1.4.5 Méthodes de validation des solutions 5](#_Toc94509107)

[2 Planification Initiale 5](#_Toc94509108)

[3 Analyse 5](#_Toc94509109)

[3.1 Opportunités 5](#_Toc94509110)

[3.2 Document d’analyse et conception 5](#_Toc94509111)

[3.3 Conception des tests 5](#_Toc94509112)

[3.4 Planification détaillée 5](#_Toc94509113)

[4 Réalisation 6](#_Toc94509114)

[4.1 Dossier de Réalisation 6](#_Toc94509115)

[4.2 Modifications 6](#_Toc94509116)

[5 Tests 6](#_Toc94509117)

[5.1 Dossier des tests 6](#_Toc94509118)

[6 Conclusion 6](#_Toc94509119)

[6.1 Bilan des fonctionnalités demandées 6](#_Toc94509120)

[6.2 Bilan de la planification 6](#_Toc94509121)

[6.3 Bilan personnel 6](#_Toc94509122)

[7 Divers 6](#_Toc94509123)

[7.1 Journal de travail 6](#_Toc94509124)

[7.2 Webographie 6](#_Toc94509125)

[8 Annexes 7](#_Toc94509126)

# Spécifications

## Titre

Projet Orienté Objet – Smart Thésaurus

## Description

Le projet consiste à faire un programme qui va indexer les contenus d’un répertoire, d’un site web et un serveur FTP(optionnel). L’utilisateur devra utiliser un moteur de recherche pour afficher les résultats de l’indexation.

## Prérequis

Modules 226 et 120. Modules 326 et 318 en parallèle.

## Cahier des charges

### Objectifs et portée du projet (objectifs SMART)

Réaliser un programme informatique de qualité

* UML (classe, activité, séquence, cas d’utilisation, …)
* MCD – MLD- MPD pour les données
* Design patterns (MVC, Template, …)
* Tests unitaires et d’intégration
* Documentation pertinente

Remplir une documentation professionnelle

* Planification (à jour)
* Journal de travail
  + Format ETML, Trello ou commit log
* Pro-activité
  + Poser des questions au client
  + Faire des démonstrations
  + Utiliser un système de versioning de code (GIT)

### Fonctionnalités requises (du point de vue de l’utilisateur)

Indexation

* Sources possibles
  + K
  + Site web
  + Serveur FTP (optionnnel)
* Choix des éléments à indexer
* Fichiers textes
* Documents Office et OpenOffice
* Documents PDF
* Images (juste le nom et la référence, contenu optionnel)
* Stockage des résultats d’indexation à choix entre
* Base de données (SQLite, MariaDB, MongoDB)
* Fichier CSV/XML
* RAM
* Choix du mode de mise à jour
* Manuel
* Chaque heure

Recherche

* Saisie des critères de recherche
* Support des opérateurs + et – (comme dans Google)
* Affichage des résultats par priorité de pertinence (meilleure correspondance)
* Accès direct à la ressource originale (clic)

Documentation

* Installation (y.c. metériel requis)
* Utilisation

### Travail à réaliser par l'apprenti

1. Créer et maintenir une planification et un journal de travail selon le modèle ETML
2. Concevoir le logiciel à l’aide de
3. Schémas UML (Classe, activité, cas d’utilisation, séquence)
4. Réflexions sur le fonctionnement du logiciel -> Schémas et choix pertinents
5. Petits programmes brouillon pour valider/invalider les idées
6. Implémenter le logiciel à l’aide de C#
7. En appliquant le paradigme OO
8. En utilisant les patterns appropriés (template, factory, …)
9. En protégeant le code avec des tests unitaires
10. En validant la qualité du produit global avec dees tests d’intégration
11. En réalisant un installateur
12. Rédiger la documentation
13. Organisationnelle :
14. Planning et (journal au format ETML, Trello ou commit log)
15. Bilans
16. Planification (différence prévu / réalité)
17. Fonctionnalités (effectué / en cours /non réalisé)
18. Personnels (qu’avez-vous appris)
19. Technique
20. Schémas/explications pour les développeurs
21. Rapports de tests
22. D’installation et d’utilisation

### Méthodes de validation des solutions

Nous allons faire des tests unitaires et d’intégration pour protéger le bon fonctionnement du code.

# Planification Initiale

Ce paragraphe présente le planning d'origine (date de début, date de fin, vacances et congés, liste hiérarchique des tâches ou GANTT, jalons, durée totale)

Toutes les mises à jour subies par le planning sont à reporter (avec date de mise à jour) et peuvent déboucher sur plusieurs versions de plannings.

# Analyse

## Opportunités

Ce paragraphe énumère la liste des difficultés potentielles de tout ordre :

Liste des compétences à acquérir ou approfondir

Liste du matériel à exploiter

Recherche d’informations particulières

Gestion du travail en équipe & collaboration

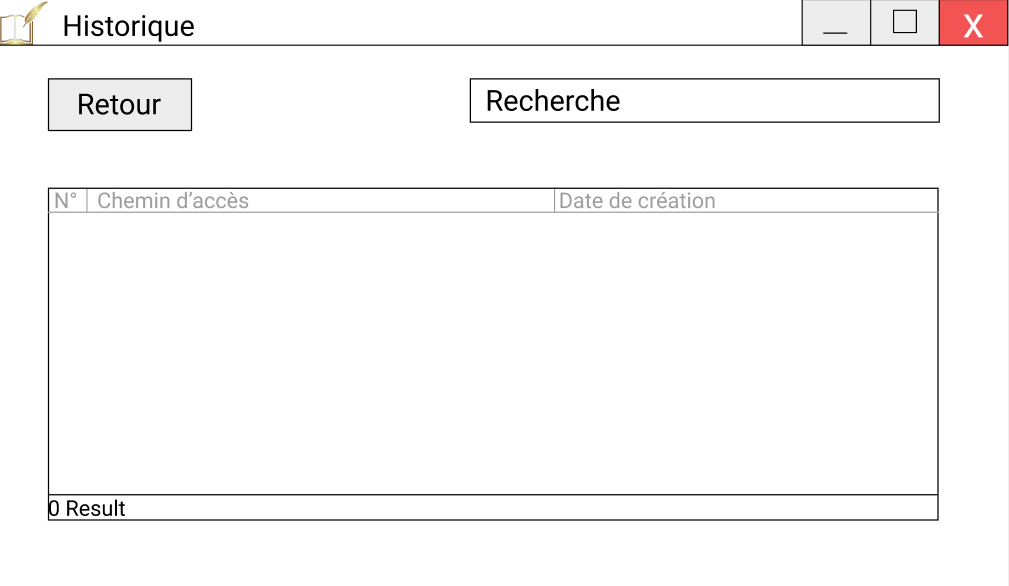
Ainsi que les solutions possibles

Si les spécifications de départ ne laissent pas de doutes sur la manière de réaliser un projet, ce chapitre ne fera que renvoyer le lecteur aux spécifications.

## Document d’analyse et conception

Le projet ressemblera à ça.





On pourra choisir le répertoire et le chemin d’accès.



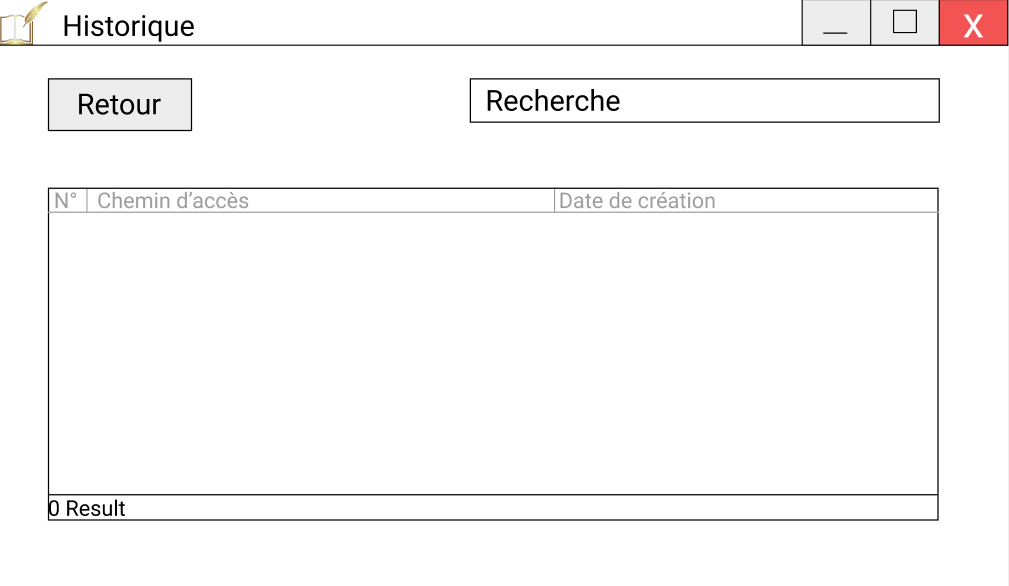
On pourra aussi choisir un dossier/ fichier grâce à la barre de recherche et ses supports des opérateurs. Il sera aussi possible de choisir le type de fichier voulu.



Après être dans un dossier, on pourra indexer le contenu du dossier. Après avoir fait cela, il sera possible d’accéder à cette indexation grâce à un historique qui nous ressort tous les indexations que nous avons faites jusque-là. De plus, il sera aussi possible d’ouvrir le fichier sélectionner directement dans l’explorateur de fichier.



Une fois l’historique ouvert, l’utilisateur pourra sélectionner une indexation déjà crée.



Ce paragraphe décrit le fonctionnement de manière détaillée.

Autant que possible de manière graphique, imagée, tableaux, etc.

Tous les cas particuliers devraient y être spécifiés…

Il s’agit d’y présenter les fonctionnalités à développer :

Découpage en étapes, en modules, en fonctionnalités, etc.

Formulaires, interfaces graphiques, pages web, etc.

Schémas de navigation, schémas événementiels, structogramme, pseudocode, etc.

Si le projet inclut une base de données :

Dictionnaire des données

Modèle conceptuel des données, modèles logique des données.

## Conception des tests

Ce paragraphe permet de spécifier la stratégie de test qui sera menée au point 5.1

Qui, quand, avec quelles données, dans quel ordre, etc.

## Planification détaillée

A ce stade, après l’analyse complète du projet, un planning détaillé et complet (avec tâches, sous-tâches, dépendances, durée, …) peut être finalisé.

Le planning détaillé doit s’inscrire dans le planning initial. Il faut que l’on puisse situer cette planification détaillée par rapport à la planification initiale.

# Réalisation

## Dossier de Réalisation

Cette partie permet de reproduire ou reprendre le projet par un tiers.

Pour chaque étape, il faut décrire sa mise en œuvre. Typiquement :

Versions des outils logiciels utilisés (OS, applications, pilotes, librairies, etc.)

Configurations spéciales des outils (Equipements, PC, machines, outillage, etc.)

Code source commenté des éléments logiciels développés.

Modèle physique d’une base de données.

Arborescences des documents produits.

Il faut décrire le parcours de réalisation et justifier les choix.

## Modifications

Historique des modifications demandées (ou nécessaires) aux spécifications détaillées.

Date, raison, description, etc.

# Tests

## Dossier des tests

On dresse le bilan des tests effectués (qui, quand, avec quelles données…) sous forme de procédure. Lorsque cela est possible, fournir un tableau des tests effectués avec les résultats obtenus et les actions à entreprendre en conséquence (et une estimation de leur durée).

Si des tests prévus dans la stratégie n'ont pas pu être effectués :

raison, décisions, etc.

Liste des bugs répertoriés avec la date de découverte et leur état:

Corrigé, date de correction, corrigé par, etc.

# Conclusion

## Bilan des fonctionnalités demandées

Il s’agit de reprendre point par point les fonctionnalités décrites dans les spécifications de départ et de définir si elles sont atteintes ou pas, et pourquoi.

Si ce n’est pas le cas, estimer en « % » ou en « temps supplémentaire » le travail qu’il reste à accomplir pour terminer le tout.

## Bilan de la planification

Distinguer et expliquer les tâches qui ont généré des retards ou de l'avance dans la gestion du projet. Indiquer les différence entre les planifications initiales et détaillées avec le journal de travail.

## Bilan personnel

Si c’était à refaire:

Qu’est-ce qu’il faudrait garder ? Les plus et les moins ?

Qu’est-ce qu’il faudrait gérer, réaliser ou traiter différemment ?

Qu’est que ce projet m’a appris ?

Suite à donner, améliorations souhaitables, …

Remerciements, signature, etc.

# Divers

## Journal de travail

Date, activité (description qui permet de reproduire le cheminement du projet), durée, liens et références sur des documents externes. Lorsqu’une activité de recherches a été entreprise, il convient d’énumérer ce qui a été trouvé, avec les références.

## Webographie

Références des sites Internet consultés durant le projet.

# Annexes

Listing du code source (partiel ou, plus rarement complet)

Guide(s) d’utilisation et/ou guide de l’administrateur

Etat ou « dump » de la configuration des équipements (routeur, switch, robot, etc.).

Extraits de catalogue, documentation de fabricant, etc.