
Rapport Projet C/SD 2018

Présenté par :
Rémy BANEL,
Ali LABBAIZE ,
Paul-Louis FEULVARCH

DASILFLIX

Encadrants :
Jonathan ARNAULT
Gérald OSTER
Pierre MONNIN
Olivier FESTOR
Soline BLANC
Sabeur ARIDHI
Sébastien DA SILVA

5 mars 2018 - 1 Juin 2018



Sommaire

Remerciements	2
1 Introduction	3
2 État de l'art	4
2.1 Approche 1 : Recommandation Personnalisée	4
2.2 Approche 2 : Recommandation Objet (Content-Based filtering CB)[1]	4
2.3 Approche 3 : Recommandation Sociale[2]	5
2.4 Approche 4 : Recommandation Hybride	6
2.5 Le cas Netflix	6
3 Choix de conception	7
3.1 Le Parsing	7
3.2 Les calculs	8
3.2.1 Les structures de données utilisés	8
3.2.2 calcul de la note	9
3.3 L'interface graphique	11
3.3.1 La SDL	11
3.3.2 Fonctionnement des pages	13
3.3.3 Fonctionnement des polices de texte	13
3.3.4 La création des images	13
4 Résultats obtenus	14
4.1 Architecture et fonctionnement du projet	14
4.2 L'impasse sur le stockage des résultats dans un fichier .json	14
4.3 Exactitude des algorithmes	15
5 Gestion de projet	15
5.1 Priorisation des tâches	15
5.2 Répartition temporelle des tâches	16
5.3 Matrice SWOT	17
5.4 Matrice RACI	18
5.5 Comptes rendus	19
6 Analyse post-mortem	19
6.1 Atteinte des objectifs et efforts individuels réalisés	19
6.2 Temps passé	21
6.3 Points positifs	21
6.4 Difficultés rencontrées	21
6.5 Points d'amélioration et suggestions	22
6.6 Remobilisation des compétences acquises	23
7 Conclusion	23
8 Annexes	23
Bibliographie	24

Remerciements

Nous tenons à remercier MM. DA SILVA et FESTOR pour leur disponibilité et leur écoute durant le projet, car certains points ont pu rapidement être éclaircis grâce à leur aide, et ce dans un court délai.

Les membres de l'équipe projet remercient également l'équipe pédagogique de TELECOM Nancy pour avoir donné aux élèves l'accès au MOOC gestion de projet de M. Rémi BACHELET, ainsi que celui du GitLab de l'école, qui a permis un développement dans un grand confort, avec son interface agréable et ergonomique.

Pour finir, l'équipe souhaite remercier l'ensemble des encadrants du module C et SD pour le choix d'un sujet concret à la fois intéressant et formateur, autant dans le domaine de la programmation que de la gestion de projet, sans oublier l'ouverture au monde de la recherche scientifique.

1 Introduction

Les systèmes de recommandation sont de plus en plus rependus de nos jours sans même que nous nous en apercevions. De Youtube à Amazon, toutes les grandes entreprises se doivent de guider leurs utilisateurs. Ces algorithmes de recommandation sont primordiales pour l'économie de certaines entreprises car ils permettent à l'utilisateur de toujours trouver un objet à son goût et le poussent donc à consommer. On estime ainsi que 80% des contenus visionnés sur le site de vidéos à la demande par abonnement Netflix le sont grâce à son système de recommandation. Ce nombre absolument gigantesque montre bien la nécessité qu'ont les entreprises d'essayer de rendre leurs algorithmes toujours plus performants, car ils peuvent être la clé de leur succès.

L'amélioration de ces algorithmes passent inévitablement par les mathématiques. Les systèmes de recommandation forment donc un véritable objectif pour les chercheurs. Netflix a par exemple organisé un concours en 2006 afin que des gens proposent des améliorations pour son moteur de recommandation.

Notre projet nous a donc amené à apporter notre patte dans ce vaste domaine des systèmes de recommandation en essayant de coder notre propre algorithme. Et cela s'avère ardu de transposer quelque chose qui paraît simple pour un humain (nous pouvons tous facilement conseiller un ami sur un film car l'on connaît sans même s'en rendre compte ses goûts de manière assez précise) sous forme d'algorithme. Réussir à ce qu'une machine "devine" les goûts d'une personne en très peu de temps et quelques choses de plus complexe que l'on pourrait imaginer.

Nous vous souhaitons une bonne lecture.

2 État de l'art

Le premier algorithme de recommandation a vu le jour en 1992 avec Paul Resnick et John Riedl, deux chercheurs en informatique, qui ont proposé le premier système de recommandation, pour les articles d'Usenet¹. Ce système collecte des notes données par les utilisateurs lorsqu'ils lisent des articles. Ces notes sont ensuite utilisées pour prédire à quel point les personnes n'ayant pas lu un article sont susceptibles de l'apprécier. Et ces prédictions sont basées sur le "filtrage collaboratif", dont le principe est simple : si une personne A a des idées similaires à une personne B sur certains sujets, alors il y a des chances que la personne A partage encore l'avis de la personne B sur un autre sujet, plutôt que celui d'une personne tirée au hasard.

Depuis, le sujet des systèmes de recommandation a pris beaucoup d'ampleur surtout avec toutes les applications qu'il a, sur des domaines assez sensibles, tels la politique et l'influence des choix des utilisateurs, avec la création des "bulles informationnelles". Des chercheurs tels que P. Resnick et J. Riedl se sont intéressés à cette dernière, ce qui a donné naissance à des centaines d'algorithmes qui ont été utilisés pour l'implémentation de systèmes de recommandation. La plupart relève de concepts mathématiques avancés, qu'on peut classer selon 4 catégories.

2.1 Approche 1 : Recommandation Personnalisée

Cette approche consiste tout simplement à proposer du contenu à l'utilisateur en se basant uniquement sur ses choix antérieurs, par le biais de listes prédéfinies de recommandations. On catégorise ainsi les utilisateurs de manière assez large et peu précise, car le contenu n'est pas pris en compte dans l'équation. On retrouve cette approche avec certaines annonces publicitaires. Par exemple AdSense de Google est considéré comme un système de recommandation personnalisée qui se base sur le comportement passé de l'utilisateur (navigation, clics, historique de recherche,...)

2.2 Approche 2 : Recommandation Objet (Content-Based filtering CB)[1]

Une seconde approche serait donc d'accorder autant d'importance à l'objet qu'à la cible. Le système se baserait ainsi essentiellement sur les qualités et propriétés intrinsèques de l'objet lui-même tout en les corrélant avec les préférences et intérêts de l'utilisateur. On obtient donc un profil utilisateur (Figure 1) ainsi qu'un profil objet. (Figure 2)

Une fois les profils créés², il est nécessaire d'évaluer à quel point un contenu pas encore vu par l'utilisateur est similaire aux contenus que celui-ci a évalués positivement dans le passé. Cette notion de similarité peut être mesurée de plusieurs façons :

- L'approche la plus simple est de ne prendre en compte qu'une seule valeur binaire. Par exemple, si le livre se trouve dans la liste des genres préférés de l'utilisateur. Dans ce cas la similarité sera de 0 ou 1.
- Une approche plus sophistiquée consiste à s'intéresser à plusieurs données de l'objet (mots clés dans le cas des livres). Il suffit ensuite de mesurer quel objet est le plus proche des goûts de l'utilisateur. Pour cela, on peut par exemple utiliser le coefficient de Dice (donné par la formule $s = \frac{2|X \cap Y|}{|X| + |Y|}$ X et Y étant des ensembles finis quelconques).

1. Usenet est un système en réseau de forums, inventé en 1979. Pour fonctionner dans un environnement Unix, il utilise alors le protocole UUCP. Il devient accessible depuis Internet grâce à l'utilisation du protocole NNTP.

2. Il existe plusieurs façon de collecter les données nécessaires pour créer les profils des utilisateurs (filtrage actif, filtrage passif,...) mais ceci ne fera pas l'objet de cet état de l'art.

Title	Genre	Author	Type	Price	Keywords
...	Fiction, Suspense	Brunonia Barry, Ken Follett	Paperback	25.65	detective, murder, New York

Figure 1. Profile d'un utilisateur.

Title	Genre	Author	Type	Price	Keywords
<i>The Night of the Gun</i>	Memoir	David Carr	Paperback	29.90	press and journalism, drug addiction, personal memoirs, New York
<i>The Lace Reader</i>	Fiction, Mystery	Brunonia Barry	Hardcover	49.90	American contemporary fiction, detective, historical
<i>Into the Fire</i>	Romance, Suspense	Suzanne Brockmann	Hardcover	45.90	American fiction, murder, neo-Nazism
...					

Figure 2. Profile d'un livre à recommander.

2.3 Approche 3 : Recommandation Sociale[2]

Ici, l'idée n'est plus de considérer un utilisateur unique, mais plutôt un ensemble d'utilisateurs. La recommandation se basant ainsi sur le comportement passé des utilisateurs similaires à l'utilisateur traité. On considère donc avec cette méthode que si deux individus ont présenté des intérêts similaires par le passé, alors ils partageront des goûts communs dans le futur.

Ce type de recommandation se divise en 2 sous catégories :

- On peut traiter la recommandation du point de vue de l'utilisateur, cette méthode sera dite user-centric. On analyse quels utilisateurs présentent des intérêts communs afin de déterminer le voisinage proche de l'utilisateur traité. On se base sur ce voisinage et des objets qu'ils ont aimés afin de déterminer quel objet pourrait plaire à l'utilisateur analysé. Afin de déterminer à quel point deux utilisateurs peuvent être proches, on utilise généralement le coefficient de corrélation de Pearson qui se présente sous la forme

$$P_{a,u} = \frac{\sum_{i=1}^m (r_{a,i} - \bar{r}_a) \times (r_{u,i} - \bar{r}_u)}{\sqrt{\sum_{i=1}^m (r_{a,i} - \bar{r}_a)^2 \times \sum_{i=1}^m (r_{u,i} - \bar{r}_u)^2}}$$

où :

- a et u sont des utilisateurs
- $r_{a,i}$ est la note donnée par l'utilisateur a à l'objet i
- \bar{r}_a est la moyenne des notes données par a
- m est le nombre total d'objets

On prend ensuite les utilisateurs possédant le coefficient de corrélation le plus élevé avec l'utilisateur traité afin de créer son voisinage. Il ne reste plus qu'à calculer l'intérêt de l'utilisateur pour un objet donné à l'aide de la formule $p_{a,i} = \bar{r}_a + \frac{\sum_{u=1}^n (r_{u,i} - \bar{r}_u) \times P_{a,u}}{\sum_{u=1}^n P_{a,u}}$ où :

- $p_{a,i}$ est la prédiction de l'objet i pour l'utilisateur a traité
- n est le nombre de personnes dans le voisinage de l'utilisateur a traité
- Mais on peut également utiliser le point de vue de l'objet, la méthode sera donc dite item-centric. L'idée est ici de mesurer la corrélation entre les objets à partir des notes attribuées par les utilisateurs. On Relis ainsi plusieurs objets entre eux et si l'utilisateur traité a aimé un

objet donné, il lui sera conseillé un objet proche de celui en question. Cependant ce système présente un défaut majeur, il a besoin de nombreux utilisateurs pour être pertinent. Ainsi le lancement de tels systèmes souffre de l'absence d'utilisateurs afin d'obtenir des recommandations pertinentes.

2.4 Approche 4 : Recommandation Hybride

Cette dernière approche des systèmes de recommandation consiste en un mélange des 3 précédentes. De plus, on va chercher à palier le cas où deux utilisateurs possédants les mêmes goûts n'ont pas évalué des objets en communs. Un filtrage par recommandation sociale (ou filtrage collaboratif) ne les considérera donc pas comme voisin et ne pourra donc pas se servir de leurs similarités afin de recommander un nouvel objet. Ici, l'idée sera donc d'assigner une valeur par défaut sur les objets notés seulement par l'un des deux utilisateurs. On améliore ainsi la prédiction en cas de rareté. Ce cas intervient lorsque la base de données du site est très grande par rapport au nombre d'utilisateurs. Dans ce cas, il est rare de trouver des utilisateurs ayant aimé les mêmes objets, la tâche de corrélation en devient très ardue. Ainsi lorsqu'un utilisateur n'a pas noté un objet, on va appliquer un filtrage item-centrique, basé sur l'objet, afin de lui attribuer une pseudo note. Il sera donc plus facile de pouvoir comparer cet utilisateur à un autre à l'aide d'un filtrage user-centrique. L'intérêt de ce système est de pouvoir compenser le lancement du système de recommandation. En effet, lors du lancement, il y a très peu d'utilisateurs et le filtrage collaboratif se retrouve fortement biaisé. Cette approche est de plus en plus utilisée et de grands groupes tels qu'Amazon et Netflix utilise une forme de celle-ci.

2.5 Le cas Netflix

Netflix est une entreprise américaine fondée en 1997. Elle propose aujourd'hui le visionnage de milliers de films et séries. Avec ses 125 millions d'abonnées, elle est aujourd'hui leader sur le secteur de la SVOD (la vidéo à la demande par abonnement). Cela est en grande partie dû à son algorithme de recommandation qui permet aux utilisateurs de rester sur la plateforme sans avoir besoin de chercher le contenu qui les intéresse. Ainsi, 80% du contenu lu sur Netflix provient du système de recommandation, ce qui permet de rendre l'utilisateur passif, ce qui l'encourage à rester plus longtemps et à consommer du contenu.

L'algorithme se base sur du machine learning en prenant en compte un certain nombre de paramètres :

- la navigation de l'utilisateur sur la plateforme
- le temps de lecture d'une série, ceci dans le but d'en déterminer le niveau d'appréciation de l'utilisateur
- les séries que l'utilisateur a arrêté de regarder et au bout de combien de temps
- les séries que l'utilisateur a regardé précédemment
- l'heure à laquelle vous avez regardé telle ou telle série
- ce que l'utilisateur a regardé la semaine précédente, le mois précédent, voire l'année précédente la même époque

De plus, Netflix appose à chaque contenu différents "tags". Ceux-ci permettent de déterminer le "champ lexical" du contenu en permettant par exemple de savoir si la série se déroule dans le futur, dans une grande ville, quels sont les personnages principaux (une femme, des enfants,...)

Avec tout ceci, Netflix a établi 6 algorithmes principaux :

- le Personalized Video Ranker qui propose pour chaque utilisateur un choix personnalisé en fonction des tags et de la popularité du contenu
- le Top-N Video Ranker qui fournit des recommandations pour chaque utilisateur en se basant uniquement sur les contenus les mieux classés par l'ensemble des utilisateurs
- l'algorithme de tendances récentes fonctionne comme le précédent mais en ne prenant en compte que les vidéos qui sont les mieux classés récemment, c'est-à-dire les tendances à court terme

- Le Continue Watching va lui se baser sur le contenu déjà visionné tout en corrélant notre point d'abandon d'une série
- l'algorithme de similitude vidéo à vidéo recommande un contenu en fonction de ce que l'on regarde actuellement. Cette algorithme est non personnalisé et se contente de renvoyer vers des vidéos prédéfinies similaires à celle que l'on regarde

Ceux-ci représentent les principaux algorithmes de recommandation de Netflix, mais ils ne sont pas seuls et c'est grâce à ces nombreux algorithmes faisant chacun des tâches précises que Netflix est parvenu à créer son système de recommandation. Celui-ci nécessite encore des améliorations bien évidemment. La société américaine souhaitant améliorer plusieurs points comme le fait de ne pas polluer nos recommandations avec une vidéo vue qui ne nous intéresse pas (par exemple des parents qui laissent leurs enfants regarder des vidéos). De même, Netflix souhaite améliorer la prédiction de l'évolution des goûts des enfants afin de toujours rester pertinent dans les choix de recommandation adressés aux enfants.

3 Choix de conception

3.1 Le Parsing

Le parsing, ou extraction de données, est l'étape principale pour bien commencer le projet. La première remarque qui peut être faite en lisant notre code de parsing *fonction_transfers.c*, est que celui-ci ne fait pas loin de 1000 lignes, ce qui est évidemment très long. Il nous semble donc important d'expliquer en détail les raisons de la longueur de ces fonctions de parsing.

Tout d'abord il faut noter que toutes les recherches sont faites grâce à une seule fonction à savoir *fgetc()*, un choix contraignant lorsque le code s'allonge, mais qui offre une sécurité contre les attaques passants par les fichiers texte, comme cela a été présenté dans le TP de C sur le sujet. On évite ainsi l'utilisation de *gets()* ou *fscanf()* qui n'offrent pas de protection contre le buffer overflow[3].

Certes le code s'en voit un peu rallongé, mais il en devient facilement compréhensible dans un même temps, dans le sens où toutes les fonctions de parsing s'appuient sur le même principe de parcours caractère par caractère.

```
1 char *title (char* source, int id);
2 char **actors (char* source, int id);
3 char** tags (char* source, int id);
4 int year (char * source, int id);
5 char * type(char * source, int id);
6 char * director (char* source, int id) ;
7 int duration (char* source, int id);
8
```

```
1 while (stock != EOF )
2 {
3     stock = fgetc(fichier);
4     if (stock == 'W')
5     {
6         stock = fgetc(fichier);
7         if (stock == 'i')
8         {
9             stock = fgetc(fichier);
10            if (stock == 't')
11            {
12                stock = fgetc(fichier);
13                if (stock == 'h')
```

```

14         {
15             stock = fgetc(fichier);
16             stock = fgetc(fichier);
17             if (stock == ':')
18             {
19                 break;
20             }
21         }
22     }
23 }
24 }
25 }
26

```

La récupération des tags (Comedy, Action ..etc) a posé quelques soucis, notamment pour récupérer le premier tag de chaque film. Nous nous sommes donc permis une seule modification du document fourni sur Arche *BDCSD.txt*, à savoir l'ajout d'un " | " au début de la ligne contenant les tags et cela pour chaque film (Figure 3), afin de faciliter l'accès à cette ligne.

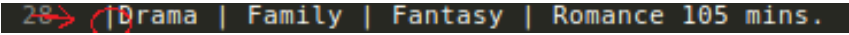


Figure 3. Modification sur le fichier BDCSD.txt

Il faut également noter que nous avons travaillé sur 99 films et non pas 100, pour la bonne et simple raison que nous avons traité caractère par caractère, donc un nombre entre 10 et 99 représente le parcours de 2 caractères, un autre entre 100 et 999 représente le parcours de 3 caractères. Ajouter un traitement pour les id à 3 caractères pour un seul film ne nous semblait peu utile en début de projet lorsque d'autres tâches s'avéraient plus importantes. Cela ne rend pas pour autant notre code inadapté à une base de donnée plus large (à condition qu'elle respecte la même forme que celle qui nous a été fournis), il suffira de copier et ajouter une ligne pour chaque 0 qui s'ajoute à l'id (100, 1000 , 10000,... etc).

```

1     else if ((id < 100) && (id >= 10))
2     {
3         stock = fgetc(fichier);
4         while (stock != EOF) {
5             stock = fgetc(fichier);
6             if (stock == 48+(id/10)) {
7                 stock = fgetc(fichier);
8                 if (stock == 48 + (id%10)) {
9                     stock = fgetc(fichier);
10                    if (stock == '.') {
11                        break;
12                    }
13                }
14            }
15        }
16

```

3.2 Les calculs

3.2.1 Les structures de données utilisés

Avant de s'attaquer tête baissée dans les calculs, il fallait stocker toutes les données récupérées grâce au parsing. Nous avons dû allouer des espaces mémoire, qui dépendent de la nature du rendu de la fonction de parsing en question. Par exemple nous avons utilisé des tableaux de *char** pour la fonction de récupération des tags ainsi que celle des acteurs.

Ensuite nous avons procédé à l'implémentations des dictionnaires (Hashmap). Ce choix de structure de données était motivé par plusieurs facteurs :

- Pouvoir associer à chaque id de film (*int*) une note (*float*), et les stocker ensemble.
- Pouvoir trier les résultats de recommandation en ordre décroissant suivant la note, mais que l'id du film reste liée au film (chose impossible à faire avec un tableau usuel).
- faciliter l'accès et la visualisation des données.

Les structures *Hashmap* et *hashmap_element* sont toutes les deux implémentées avec les prototypes des fonctions associés (nous avons défini le strict nécessaire de fonctions pour réaliser les tâches d'affichage, de stockage et de tri avec cette structure) dans le fichier *fonction_calcul.h* comme toutes les autres structures du projet.

```

1  typedef struct _hashmap_element{
2      int key;
3      float value;
4  } hashmap_element;
5  typedef struct _hashmap_map{
6      int table_size;
7      int* actual_size;
8      hashmap_element *data;
9  } hashmap_map;
10 // CONSTRUCTEURS //
11 hashmap_element* cst_hashmapelement(int key, int value);
12 hashmap_map* cst_hashmap (int size);
13 // FONCTIONS DE MANIPULATIONS DES HASHMAP //
14 void hashmap_add(hashmap_map m, hashmap_element e);
15 void hashmap_tostring(hashmap_map m);
16 void trihashmapvalue (hashmap_map m);
17

```

Ensuite nous avons implémenté une structure *Matrice* afin de coder une approche collaborative, mais finalement le projet a pris une autre direction : nous avons fait le choix de présenter cette approche de manière théorique dans la rapport (explication : cf "points d'amélioration et suggestions").

```

1  typedef struct Matrice{
2      int nbLigne;
3      int nbColonne;
4      float cases[nbLigne][nbColonne];
5  } Matrice;
6

```

3.2.2 calcul de la note

Une fois que nous avons défini toutes les structures de données, nous nous sommes attaqué aux structures *film* et *utilisateur* :

```

1  typedef struct Film{
2      int id;
3      char titre[400];
4      char realisateur[100];
5      char type[50];
6      int annee;
7      int duree;
8      char ** acteur;
9      char ** tags;
10 } Film;
11
12 typedef struct Utilisateur{
13     int id;
14     Film likedmovies[3];
15 } Utilisateur;
16

```

Nous avons défini un modèle où l'utilisateur fait 3 choix parmi les films proposés, et, en fonction de ces choix, nous calculons les préférences de cet utilisateur. Nous comparons tous les films dans la base de données avec ces préférences, et les plus similaires auront les notes les plus élevées. On peut schématiser notre modèle ainsi (Figure 4) :

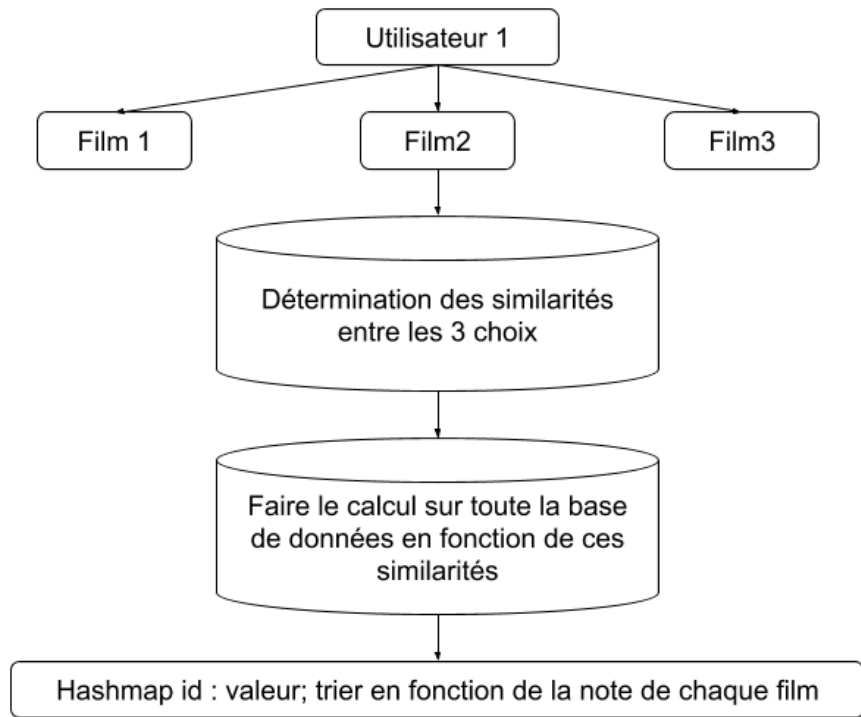


Figure 4. Modèle de notre recommandation Objet

Le choix des films se fait au niveau de l'interface graphique. Il est envoyé aux fonctions de calcul, grâce à la fonction *Selection()*. Une fois les 3 id stockés, nous avons utilisé le principe de diviser pour régner¹. Nous avons codé une fonction pour calculer la similarité entre les 3 films et cela pour chaque donnée à noter :

```

1  int * favyear (Film f1 , Film f2 , Film f3);
2  int * favduration (Film f1 , Film f2 , Film f3) ;
3  int * favtags (Film f1 , Film f2 , Film f3);
4  char * favrealisateur (Film f1 , Film f2 ,Film f3);
5  char * favacteur (Film f1, Film f2, Film f3);
6

```

- *favyear* : cette fonction prend l'année du film le plus récent et celui du plus ancien, et elle renvoie [ancien -4 , nouveau +4] afin de vérifier si le film à traiter est dans la même période cinématographique.
- *favduration* : même fonctionnement que *favyear*, avec un intervalle de -+15min.
- *favrealisateur* : cette fonction vérifie si un réalisateur est le même dans 2 des 3 films choisis, si oui elle le renvoie, afin de donner une note supplémentaire aux autres films de ce réalisateur.
- *favacteur* : elle fait l'intersection entre les acteurs des 3 choix, mais cette fois deux à deux, si un acteur se répète deux fois, on le garde, pour la même raison que le réalisateur.
- *favtags* : et finalement la fonction la plus importante (car on a fait le choix que la note attribuée aux genres soit la plus importante), cette fonction renvoie tous les tags qui se

1. Principe vu en TOP pendant le premier semestre, qui consiste à diviser : découper un problème initial en sous-problèmes. Puis régner : résoudre les sous-problèmes (récursivement ou directement s'ils sont assez petits). Pour finalement combiner : calculer une solution au problème initial à partir des solutions des sous-problèmes.

répètent au moins deux fois dans les tags des 3 films.

Une fois toutes les fonctions de calcul de similarités codées, il fallait les utiliser dans une grande fonction *calculnoteObjet* qui prend en argument un profil de film et d'utilisateur, et renvoie la note qui représente le degré de similarité entre ce film et le profil de cette personne :

```
1 float calculnoteObjet(Film f, Utilisateur u);
2
```

Tout d'abord on initialise la note à 0, et ensuite, nous l'incrémentons suivant le système de notation suivant :

- si le film est dans l'intervalle de *favyear* → 1.5
- si le film est dans l'intervalle de *favduration* → 0.5
- si le film a le même réalisateur que *favrealisateur* → 1.5
- si le film contient le même acteur que dans *favacteur* → 2.0
- et pour chaque tag similaire à un tag de *favtags* → 3.0/(le nombre de tags des films en question)

Une fois la note attribuée au film, on la stock dans le hashmap avec les autres notes des autres films. Une fois que toute la base de donnée est parcourue, nous trions le hashmap et nous le renvoyons. Finalement pour la fonction *tags*, afin de faciliter les calculs et les intersections des tableaux de tags, nous avons fait en sorte d'associer un tableau de 0 et de 1 à chaque film qui définit ses tags. Donc nous avons compté le nombre de tags totaux dans la base de données (on a créé un tableau de 18 cases, chaque case correspond à un tag, si le tag apparaît dans la description du film dans la case ça sera 1, sinon 0). Or ce choix contraint un peu le code. Lorsque nous incluons plus de films avec d'autres tags en dehors des 18 qu'on a déjà, cela oblige à l'ajouter dans la fonction *tagsbinaire*.

```
1 int* tagsbinaire(Film f1){
2     /* code raccourci pour comprendre le fonctionnement*/
3     int *tagsint = (int*) malloc(sizeof(int)*18);
4     if (strcmp(f1.tags[i], " Action " ) == 0){
5         tagsint[0]=1; }
6         .
7         .
8         .
9
10
11     else if (strcmp(f1.tags[i] , " Sport " ) == 0) {
12         tagsint[16]=1; }
13     }
14     return tagsint;
15 }
16
```

3.3 L'interface graphique

3.3.1 La SDL

Nous avons choisi d'utiliser la bibliothèque SDL afin de réaliser l'interface graphique. Celle-ci est assez complète, elle permet de gérer le son, l'affichage d'une fenêtre et également de faire de la 2D. Elle est de plus libre de droit et gratuite, ce qui est un point non négligeable pour un projet tel que le notre. Nous avons rajouté 2 modules complémentaires pour faciliter certains redimensionnement d'images et d'affichage de texte.

La bibliothèque utilise des *SDL_Surface* pour stocker un tableau de couleur qui correspond aux pixels des images chargées. Ainsi la fonction *load_image* permet de charger les images et de les associer aux surfaces de la bibliothèque SDL. Le programme renvoie un message d'erreur en cas d'échec de chargement, l'image n'existe pas ou ne comporte pas le nom émis par le programme. Une

Avantages	Inconvénients
Rapidité à appréhender	Difficulté d'apprentissage
Facilité pour afficher des images (1 SDL_Surface)	Affichage dynamique
Fonctions complémentaires	Fonctions complémentaires
Rapidité d'exécution	Utilisation d'images non compressées

Table 1. Avantages et Inconvénients des fichiers de la SDL

fonction *apply_surface* permet de simplifier l'affichage des images avec une position précise, au lieu de la fonction déjà implémentée à la bibliothèque SDL : *SDLBlitSurface*. Il suffit de donner en argument les coordonnées (x,y) du premier pixel (situé en haut à gauche)(Figure 5), la SDL_Surface où nous aurons charger l'image au préalable. Il faut également spécifier la Surface "de fond" qui sert de support à toutes les SDL_Surface.

```

1 //Fonction pour bliter la surface //
2 void apply_surface(int x, int y, SDL_Surface* source, SDL_Surface* destination
3 ){
4     SDL_Rect offset;
5
6     offset.x = x;
7     offset.y = y;
8
9     //Blittage de surface//
10    SDL_BlitSurface(source, NULL, destination, &offset );
11 }

```

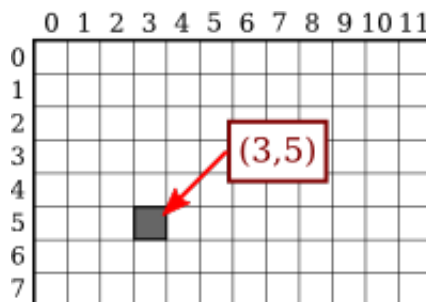


Figure 5. Système de coordonnées en SDL

Au préalable, il faut initialiser SDL puis choisir le mode vidéo. Cela nous permet de spécifier les dimensions et les moteurs graphiques utilisés. Nous utilisons le moteur implémenté à la SDL : *SDL_HWSurface* mais nous aurions pu également utiliser OpenGL (plus connu). Ce choix paraît plus raisonnable, en effet, les problèmes de compatibilité sont moins fréquents car ce sont les mêmes développeurs qui ont élaboré la bibliothèque et le moteur graphique. Les dimensions retenues sont celle de la *High Definition Standard*, ie 1280 par 720. Ainsi tous les écrans peuvent afficher notre programme de recommandation de films.

La transparence est également gérée par la bibliothèque SDL, il faut spécifier la couleur qu'il faut "effacer" et rendre transparente.

```

1 //Rendre une couleur transparente//
2 SDL_SetColorKey(logo, SDL_SRCCOLORKEY, SDL_MapRGB(logo->format, 255,255,255));
3

```

Pour actualiser la surface de "fond" et ainsi afficher les différentes images à l'écran, nous devons utiliser la fonction.

SDL_Flip()

```
1 SDL_Flip( ecran );  
2
```

3.3.2 Fonctionnement des pages

Chaque page est fonction de la précédente, le programme est donc relativement linéaire mais certaines fonctionnalités peuvent être amenées à se répéter et donc sont codées dans des fonctions pour éviter le code trop long. Ainsi la position de la souris, l'affichage d'images en cascade, avec des positions fixes ont été mises dans des fonctions. Également les différentes pages ont été codées dans des fonctions séparées pour faciliter la lecture et également la mémoire. En effet, nous pouvons vider la mémoire où sont stockées les images après avoir changé de page, il suffit de garder seulement les informations importantes (les numéros identifiant les films choisis, recommandés, position du curseur...).

Afin de détecter toutes actions de l'utilisateur, nous avons créé des événements, qui selon leur nature effectuent différentes tâches : renvoient à la page suivante, quittent l'application, affichent d'autres images... Une boucle while permet ainsi d'effectuer constamment une récupération du type d'évènement, qui est ensuite comparé selon l'action faite par l'utilisateur.

Une difficulté a été de régler le passage de la souris (type d'évènement : `SDL_MOUSEMOTION`) sur un bouton cliquable et de changer l'image en conséquence et de gérer la transparence de l'élément. Il faut réafficher alors toutes les images lorsque la souris n'est plus sur le bouton.

La détection de l'action : cliquer sur un bouton, est détectée par l'évènement `SDL_MOUSEBUTTONDOWN`. Ainsi le changement d'affichage s'effectue lorsque le bouton de la souris a été relâché, ce qui permet à l'utilisateur de relâcher le bouton de sa souris à une position différente du bouton, ainsi le changement d'affichage ne sera pas effectué.

Afin de ne pas surcharger les capacités de l'ordinateur, on vide les images affichées sur la `SDL_Surface` de "fond". Ainsi elles restent chargées mais ne surchargent pas l'interface et les graphismes de la machine.

3.3.3 Fonctionnement des polices de texte

La bibliothèque SDL ne permet pas de traiter les affichages de texte, il faut installer un module supplémentaire : *SDL_ttf*. Ce module charge les polices de caractères présentes dans le dossier du programme. Ainsi les polices ont été choisies à l'avis du groupe. Nous pouvons également rajouter des types : souligné, gras, italique...avec le module chargé, une fois que ce dernier a été initialisé. Plusieurs fonctions permettent d'afficher le texte dans le module. La fonction la plus puissante a été choisie. La transparence est totalement respectée avec l'image derrière, mais elle est légèrement plus lente à s'afficher.

Les zones de texte fonctionnent de la même manière que les images : il est nécessaire de créer une `SDL_Surface` pour libérer la mémoire disponible à l'affichage. Puis on "blitte" avec la fonction vue précédemment la surface avec le texte. On spécifie des positions pour fixer la surface. Puis on actualise la surface de "fond" pour que le texte apparaisse.

3.3.4 La création des images

Toutes les images ont été créées à l'aide du logiciel Gimp, qui a l'avantage par rapport à photoshop et illustrator d'être totalement gratuit. Évidemment la principale source d'inspiration des images a été les visuels de Netflix.

4 Résultats obtenus

4.1 Architecture et fonctionnement du projet

Concernant l'architecture adoptée pour ce projet (Figure 6), nous avons réduit au maximum le nombre de fichier .c et .h que nous avons utilisé, un choix qui s'est avéré peu judicieux quand nous sommes arrivés à un stade avancé du projet. Nous avons quand même séparé le code en 3 grands fichiers :

- *fonction_calcul.c* qui contient tous les codes concernant le calcul des recommandations objets, et *fonction_calcul.h* qui contient toutes les structures précédemment illustrées (cf structures de données)
- *fonction_transfers.c* pour les fonctions de parsing.
- Et finalement *interfacetest.c* qui contient le main principal avec tout ce qui concerne l'interface graphique.
- 100 images en .bmp une pour chaque film, ainsi que les images qui ont servi de prototype, que nous n'avons pas réussi à mettre dans un seul fichier, parce que cela empêche l'automatisation de l'appel de ces derniers dans le code (notamment lors de la concaténation de caractères).

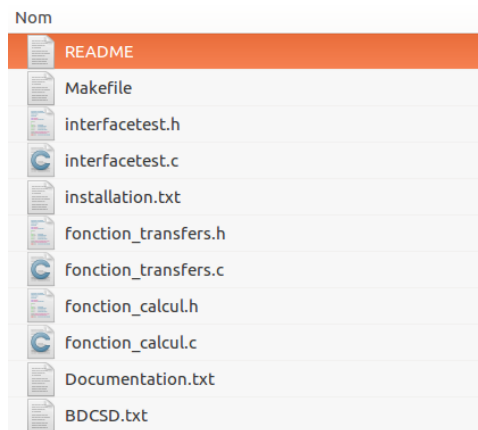


Figure 6. Architecture du projet

4.2 L'impasse sur le stockage des résultats dans un fichier .json

L'approche du stockage des résultats dans un fichier spécifique .json n'a pu être abordée par le groupe en raison de l'avancée du projet. Nous n'avons pas pu enregistrer les données utilisateur et intégrer une interface utilisateur. Ainsi les données relatives à l'application ne sont pas sauvegardées lorsque l'application est quittée. Elles restent cependant stockées dans la mémoire vive lors de l'exécution du programme. L'approche utilisateur n'étant pas réalisée, le stockage de données est inutile dans notre cas.

Cependant des recherches ont été faites sur les fichiers json, pour s'en servir éventuellement dans un futur projet où le stockage des résultats est inévitable. En effet, les json présentent un certain nombre d'avantages et d'inconvénients pour sauvegarder des données.

Avantages	Inconvénients
Format compatible par tous (humain & machine)	Garantir la sécurité des données
Ne dépend d'aucun langage	Syntaxe rudimentaire
Permet de stocker des données de différents types	

Table 2. Avantages et Inconvénients des fichiers .json

4.3 Exactitude des algorithmes

La seule exactitude que nous pouvons évaluer dans un projet de recommandation, c'est celle des calculs des préférences. Notre code de calcul (précédemment détaillé), et nous jugeons que les résultats renvoyés par notre algorithme sont potentiellement adaptés aux choix des utilisateurs, et quelques modifications ce sont faites au fur à mesure du projet, pour atteindre une précision plus fine.

Un exemple concret de ces modifications est le changement de la façon d'attribuer les notes en fonction des tags : nous avons remarqué que des films apparaissent régulièrement dans les films recommandés par l'application, cela est dû au grand nombre de tags qui leur sont attribués dans le fichier .txt. naturellement ils vont contenir les tags préférés de l'utilisateur sans que le film soit vraiment totalement dans cette catégorie, donc nous avons dû diviser la note des tags par le nombres de ces derniers par film.

5 Gestion de projet

5.1 Priorisation des tâches

Lorsque le projet a débuté, il était difficile de savoir dans quelle direction nous diriger. Il a donc fallut rapidement catégoriser les tâches principales en fonction de leur urgence et de leur importance pour le projet. C'est en cela que la matrice d'Einsenhower nous a été utile, pouvoir visualiser très rapidement ce qui serait primordial de faire et ce qui pouvait prendre un peu plus de temps à se mettre en route. Il nous a vite paru qu'une une bonne préparation du projet en amont serait inévitable afin de ne pas nous retrouver dépassés par les événements. Nous avons donc cherché à mettre l'accent sur la gestion de projet en début de celui-ci.

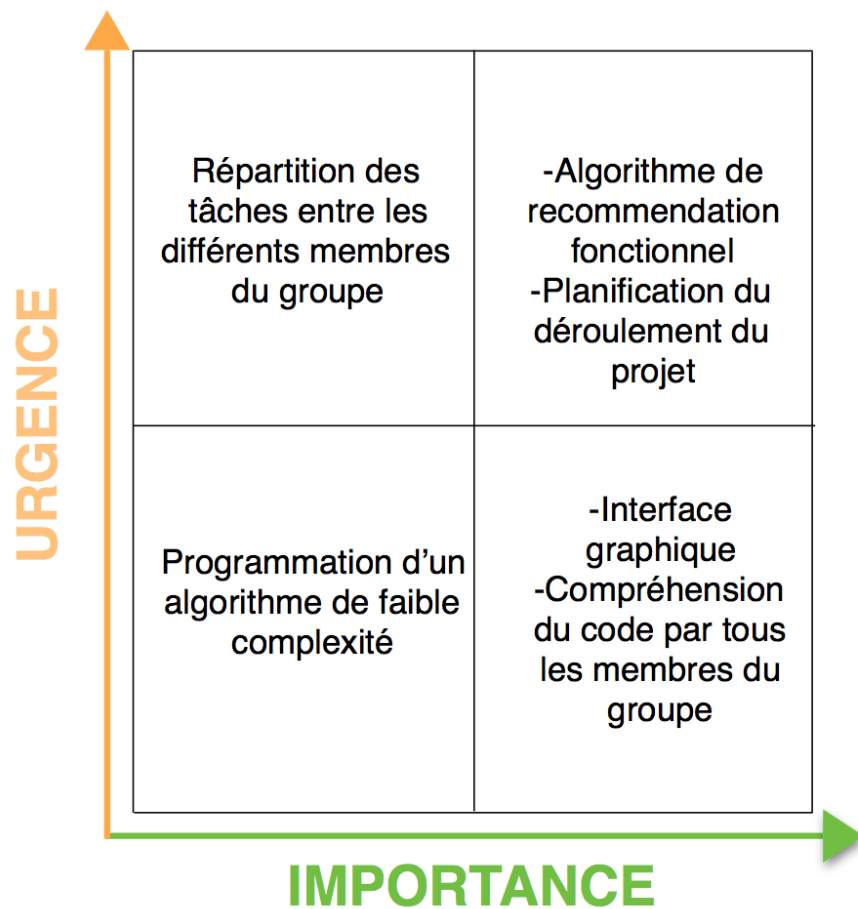


Figure 7. Diagramme importance/urgence de notre projet

5.2 Répartition temporelle des tâches

Une fois la priorisation des tâches effectuée, une tâche importante fût de jalonner le projet afin d'éviter de se retrouver pris par le temps. Dans ce but, le diagramme Gantt était l'outil qui nous semblait le plus adapté. Nous en avons donc réalisé un dès le début du projet, lorsque les principales étapes du projet ont été définies. Nous avons essayé de nous y tenir le plus possible, même si bien évidemment, divers imprévus font que les dates ne sont au final pas toujours respectées rigoureusement.

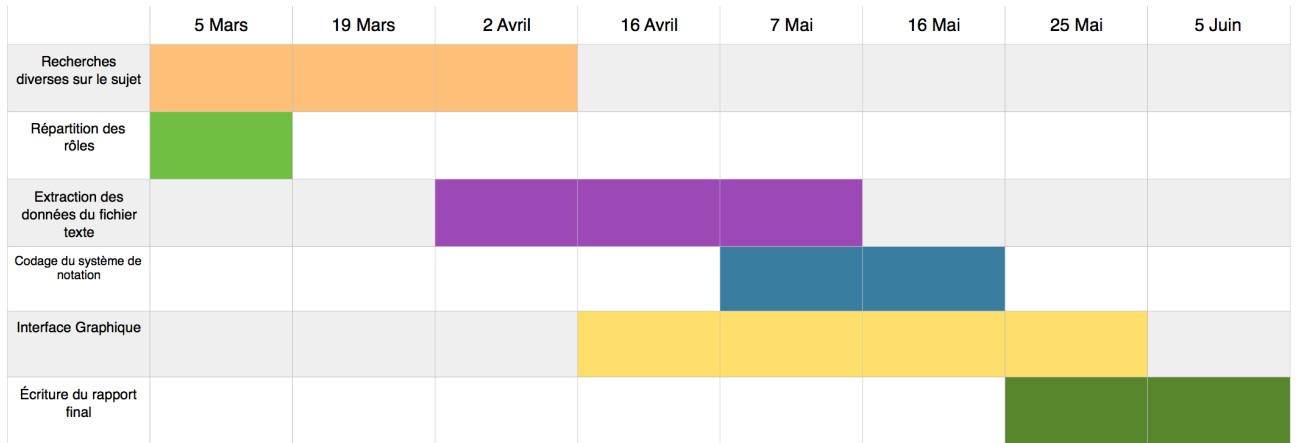


Figure 8. Diagramme prévisionnel des différentes tâches du projet

Nous avons également réalisé, à titre d'indication, un second diagramme, réalisé une fois le projet fini. Celui-ci permet de voir les différences entre le gantt prévisionnel et ce qui a réellement été effectué.

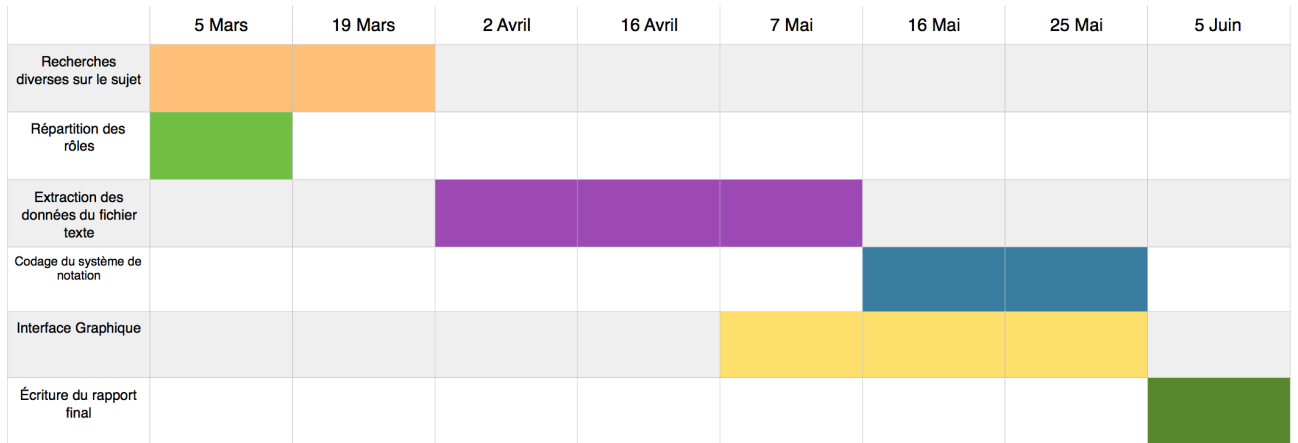


Figure 9. Diagramme final des différentes tâches du projet

On constate que les différences sont plutôt minimales et n'ont pas dérangé outre mesure le déroulement du projet.

5.3 Matrice SWOT

La matrice SWOT est plus un outil de stratégie des entreprises, leur permettant d'établir les menaces et les opportunités d'un nouveau produit. Dans notre cas, elle nous a surtout permis d'établir une liste de nos faiblesses sur lesquelles nous devons nous concentrer et nous améliorer afin de rendre le projet dans les meilleures conditions.

	Positif	Négatif
Origine Interne	Forces : <ul style="list-style-type: none"> ▪ Sujet intéressant car s'appliquant à des cas concrets, lié à l'actualité ▪ Formation préalable par un MOOC gestion de projet ▪ Possibilité de communication informelle et fréquente dans l'équipe par messageries instantanées (Skype et Messenger) ▪ Bonne entente du groupe ce qui facilite la communication ▪ Tous les membres du groupes sont très motivés ▪ Les membres de l'équipe ont déjà un peu d'expérience afin d'allier programmation et gestion de projet 	Faiblesses : <ul style="list-style-type: none"> ▪ Composition peu variée de l'équipe en termes de parcours et de capacités : <ul style="list-style-type: none"> ★ Ali et Rémy : viennent de MP et ont eu plus de contact avec les maths que l'informatique ★ Paul-Louis : vient de PSI et a eu plus de contact avec la physique que l'informatique ▪ Les membres de l'équipe projet n'ont jamais travaillé ensemble ▪ Les membres de l'équipe ont peu d'expérience avec le C ▪ Difficulté à fournir un travail régulier sur une échéance longue (effet tunnel)
Origine externe	Opportunités : <ul style="list-style-type: none"> ▪ Disponibilité des encadrants ▪ Plateforme de gestion des versions agréable à utiliser : gitlab.telecomnancy.univ-lorraine.fr ▪ Pédagogie de la gestion de projet sur un cas concret : connaissances remobilisables 	Menaces : <ul style="list-style-type: none"> ▪ Le langage C n'est pas toujours facile à prendre en main, notamment avec les allocations de mémoire ainsi que les pointeurs ▪ Le fichier ▪ Sujet mal défini qui présente une certaine difficulté afin de s'orienter dans la bonne direction ▪ Emploi du temps chargé des membres de l'équipe (cours, période de partiels, loisirs, ...)

Figure 10. Diagramme prévisionnel des différentes tâches du projet

5.4 Matrice RACI

Bien répartir les différentes tâches d'un projet est un élément essentiel afin de ne pas perdre de temps inutilement. En effet, si deux personnes se mettent à coder la même fonction, la perte de temps engendré sera non négligeable, d'où l'utilité de mettre en place une RACI très tôt afin d'éviter ce type de problème.

	A.LABBAIZE	R.BANEL	PL.FEULVARCH
Récupération des données du fichier texte	RA	I	R
SDL	I	R	RA
Système de recommandation	RA	CR	CI
Création des images du jeu	CI	RA	I
Fonctionnalités supplémentaires	RA	I	R
Rédaction du compte rendu	RA	R	R

5.5 Comptes rendus

Les différents comptes rendus de réunion ont été placés en fin de rapport, dans la partie Annexes.

6 Analyse post-mortem

6.1 Atteinte des objectifs et efforts individuels réalisés

Tout au long du projet, les tâches à effectuer ont été distribuées aux membres de l'équipe au fur et à mesure de l'avancement de celui-ci, selon les compétences de chacun, au cours des différentes réunions (Table 4).

Description	Responsable	Livrable
Etat d'art	A.LABBAIZE	Fichier LaTeX
Se renseigner sur les interfaces graphiques	R. BANEL	-
Réalisation du GANTT	AL/PLF	Diagramme
Mise en forme de la matrice SWOT	R. BANEL	Une matrice SWOT
Algorithme de parsing des titres/années	A. LABBAIZE	Code C
Algorithme de parsing des tags(genres)	A. LABBAIZE	Code C
Algorithme de parsing des acteurs/réalisateurs	A. LABBAIZE	Code C
Algorithme de parsing des durées	A. LABBAIZE	Code C
Transformation des tags en données numériques afin de faciliter les calculs	A. LABBAIZE	Code C
Création des structures film et utilisateur	A.LABBAIZE	Code C

Table 3. Analyse des efforts individuels

Rassembler les fonctions (fav) dans une seule fonction de calcul de recommandations et pondérer les résultats	A. LABBAIZE	Code C
Implémentation de la structure Hashmap pour faciliter le tri et la visualisation des résultats	A. LABBAIZE	Code C
Construire plusieurs fonctions (fav) qui calcule les préférences d'utilisateur, en se basant sur le choix de 3 films	A. LABBAIZE	Code C
Modélisation théorique de l'approche sociale	A. LABBAIZE	-
Continuer l'état de l'art	R. BANEL	Fichier LaTeX
Coder une première interface graphique avec SDL	PLF	Code C
Coder un produit matriciel	R. BANEL	Code C
Coder un prototype de l'interface graphique	R. BANEL	Code C
Visuels de l'interface graphique	R. BANEL	Image .png
Récupérer les affiches des 99 films et séries de la liste	R. BANEL	image
Création du logo du programme	R. Banel	Image .png
Implémentation de l'interface graphique	PLF	Code C
Associer les fonctions de calcul et de parsing avec l'interface graphique	A.LABBAIZE/PLF	Code C
Rédaction du rapport	Toute l'équipe	Un Rapport LaTeX

Table 4. Analyse des efforts individuels

6.2 Temps passé

	A.LABBAIZE	R.BANEL	PL.FEULVARCH
La gestion de projet initiale	5h	3h	2h
Recherches préliminaires	6h	4h	3h
Extraction des données du fichier texte	48h	2h	-
Système de notation	40h	2h	1h
Conception des images	-	10h	-
Interface graphique	5h	7h	55h
Fonctions supplémentaires	10h	3h	-
Tests divers	1h	2h	1h
Rédaction du compte rendu	2h	10h	2h
Total	117h	43h	64h

6.3 Points positifs

Les opportunités et forces mises en valeur dans la matrice SWOT réalisée au début du projet ont effectivement été saisies et exploitées :

- La bonne entente entre les différents membres a permis une bonne communication au sein du groupe ce qui a été essentiel dans la répartition des tâches
- Le sujet intéressant a motivé l'équipe projet à s'investir pour voir aboutir les résultats
- La nécessité de réaliser une interface graphique a permis d'avoir une vision plus concrète du bon avancement du projet
- La formation préalable par un MOOC gestion de projet a permis de réaliser des documents ayant permis une meilleure organisation au cours du projet
- La communication par messagerie instantanée a été utile pour connaître l'avancement du projet quasiment en temps réel, tâche par tâche
- Les encadrants ont pu éclaircir certaines interrogations sur le projet, cela nous a aidé à partir dans la bonne direction

6.4 Difficultés rencontrées

Même si nous avons tenté au maximum de réduire de façon pro-active les faiblesses et menaces mises en lumière dans la matrice SWOT, nous avons tout de même fait face à quelques une d'entre elles :

- À cause de l'effet tunnel, il a été difficile de fournir un travail régulier sur une échéance longue, certaines périodes étant creuses en terme de travail fournis en comparaison avec d'autres périodes un peu plus "fasts"
- Le sujet qui n'était pas guidé (contrairement au sujet du premier semestre) a été un ralentisseur notable en début de projet. En effet, il était difficile au début de savoir dans quelle direction partir
- La recommandation sociale nécessitant la création d'une base de donnée d'utilisateurs n'a pu être mise en place
- Le stockage des données utilisateurs ainsi que l'implémentation d'une page d'inscription nous ont posé problème, ce qui nous a empêché de réaliser l'approche collaborative
- La réalisation de l'interface graphique s'est avérée déroutante au début car n'était pas vu en cours. Il a fallu de former de notre côté afin de pouvoir la réaliser

Au niveau du planning durant ce projet, la mise en place du travail a pris un certain temps. Les premières réunions étant là notamment pour essayer de voir quelle direction prendre pour bien début

le projet. Ainsi les premières lignes de code ont mis du temps à voir le jour et nous avons dû attendre les vacances d'avril pour voir une avancée significative du projet. Cependant une fois le projet lancé, le travail a été assez régulier ce qui est un point plutôt positif.

Si on compare le temps prévu sur les tâches avec le temps passé, il est évident que de manière générale, les tâches ont mis plus de temps à être réalisées que prévu. Cela est dû à plusieurs facteurs. Certaines tâches semblaient abordables de prime à bord, mais se sont révélées plus ardues que prévu. On pense notamment à la fonction d'extraction des données du fichier texte qui nous été fourni. Un autre facteur de cette différence de temps était le soucis d'auto formation. Évidemment, nous n'avions pas toutes les connaissances requises en C pour un tel projet, notamment au niveau de l'interface graphique, il a donc été primordial de suivre plusieurs tutoriels afin de palier à ce problème, ce qui représente au final un temps assez important.

6.5 Points d'amélioration et suggestions

Au début du projet, le groupe envisageait un approche hybride, c'est-à-dire mélanger l'approche objet (présenter dans le projet), avec l'approche sociale qui se base sur le recyclage des données enregistrées de chaque utilisateur. Ainsi la recommandation est plus personnalisée (cf état de l'art). Une modélisation théorique est réalisée ainsi qu'une implémentation des structures nécessaires pour réaliser cette approche. Malheureusement une fois que le groupe a affronté la difficulté de l'implémentation de l'interface graphique en C, on a fait le choix de se restreindre à une seule approche fonctionnelle¹ (recommandation objet).

Notre modélisation de cette approche se base essentiellement sur les matrices, et le stockage des films aimé par l'utilisateur dans un fichier txt. Par exemple pour un utilisateur non inscrit à notre logiciel, nous lui créons un profil (pseudo, mot de passe, id ...etc), et nous enregistrons toutes ses données dans le fichier *id.txt*. Une fois tout cela est fait, pour la prochaine connexion d'un utilisateur (id = 1 par exemple), on récupère les 3 films les mieux noté par ce dernier, depuis ce fichier *1.txt* dans ce cas, on applique la fonction *calculnoteObjet*; entre ces 3 films et tous les autres films pour stocker chaque note résultante dans une matrice $(U)_{0 < i < 100, 0 < j < 100}$, liée à chaque élément $u_{i,j}$ soit à 1 ou à 0, en fonction de si la fonction renvoie une note de plus de 6/10 au film de id $i*j$ on aura $u_{i,j} = 1$ sinon $u_{i,j} = 0$. Une fois deux matrices de deux utilisateurs sont calculées nous avons décidé d'utiliser la norme 1

$$||M||_1 = \sum_{k=0}^n |m_{ik}|$$

pour calculer la distance $d(U1, U2) = ||U1 - U2||$ entre deux utilisateurs U1 et U2.

Illustration :

Soit $U1$ et $U2$ deux matrices de $(\mathcal{M})_{100}(R)$ définies ainsi :

$$u_{i,j} = \begin{cases} 0 & \text{si calculnoteObjet(film(i*j),u) < 6} \\ 1 & \text{sinon} \end{cases}$$

Donc $U1$ ou $U2$ ressemblerons à :

$$\begin{pmatrix} 0 & 1 & \dots & 0 \\ \vdots & \ddots & \ddots & \vdots \\ 1 & 0 & \dots & 1 \end{pmatrix}$$

1. «Commencez simple, et puis on verra ..» - Sébastien Da-Silva

Une fois que la soustraction $U1 - U2$ est faite, plus les matrices sont similaires, plus de 0 seront présents dans la matrice résultante, donc la norme de cette soustraction (distance) sera petite, et vice versa.

L'idée était de calculer la distance entre les utilisateurs déjà inscrits chez nous, et si l'un d'eux désire une recommandation, on lui proposera un film aimé par l'utilisateur le plus proche de ce dernier, dans le sens de la distance précédemment définie. La structure matrice étant déjà définie, ainsi que la modélisation des calculs, le problème restait dans l'implantation d'une page d'inscription avec SDL, qui semblait très long et compliqué, donc nous avons privilégié de fournir une seule approche avec une interface graphique fonctionnelle, que deux approches sans interface graphique.

6.6 Remobilisation des compétences acquises

De nombreuses connaissances acquises lors des différents TP de C ont été nécessaires. Notamment tout ce qui concerne l'extraction des données du fichier texte. Certains exercices de TP ressemblant à ce qui a pu être réalisé sur cette partie, la difficulté étant évidemment bien moins importante lors des TP que face au cas concret auquel nous avons fait face ici. De plus la plus part des connaissances obtenues lors du MOOC de gestion de projet nous ont été utiles. Ceci nous a permis de mettre en application toutes les méthodes vues et de nous rendre compte de l'utilité de ce MOOC qui s'est avéré bien moins trivial que ce que l'on pouvait penser de prime à bord.

7 Conclusion

Ce projet a été très enrichissant et professionnalisant de part la partie importante de gestion de projet y étant été menée. De plus, la rigueur et la diversité des rendus demandés ont été perçus comme très formateurs. Les membres de l'équipe projet ont trouvé le domaine scientifique étudié passionnant.

Le projet basé sur un cas concret nous a permis de constater chaque amélioration que nous réalisons pour celui-ci, ce qui s'est avéré particulièrement motivant. Le travail sur l'interface graphique, qui était une première, nous a permis de voir au mieux l'avancée du projet et nous a permis de découvrir de nouvelles fonctionnalités du C que nous n'avons pu étudier en cours. En cela le projet est un prolongement naturel des cours enseignés à Telecom Nancy.

8 Annexes

Bibliographie

- [1] Jonathan Lou  dec. Algorithmes de bandits pour les syst  mes de recommandation. *Institut de Recherche en Informatique de Toulouse (IRIT) Institut de Math  matiques de Toulouse (IMT) UMR5505, UMR5219, CNRS*, 2005.
- [2] Prem Melville, Raymond J. Mooney, and Ramadass Nagarajan. Content-boosted collaborative filtering for improved recommendations. *Department of Computer Sciences University of Texas Austin, TX 78712*, 2002.
- [3] Christophe BLAESS.   viter les failles de s  curit   d  s le d  veloppement d'une application 3   me partie. *Ing  nierie et formations sur les syst  mes libres*, 2011.

Projet C/SD 2018 : Compte-rendu de réunion 1

Minutes for Mars 15, 2018

Present: Rémy BANEL, Ali LABBAIZE Paul-Louis FEULVARCH

Absent: Aucun

Type de réunion : Réunion de lancement de projet et première répartition des tâches

Durée : 1 heure et 10 minutes

Lieu : Médiathèque de l'école

Ordre du jour :

1. Partage des points de vue et des connaissances récoltées sur le sujet.
2. Répartition des tâches en fonction des compétences de chacun.
3. Se mettre d'accord sur l'environnement de travail (Code block, éditeur de texte ..etc)
4. La mise en évidence des risques temporels. Définir les étapes du projet, afin de mettre en place un GANTT.
5. Présentation du débbugueur GDB par A. LABBAIZE au groupe.
6. Election des responsables de projet.

Informations échangées :

1. Présentation de deux approches possibles des systèmes de recommandation, le "control-based learning" et le "collaborative filtering" (cf état de l'art).
2. Échange des informations récoltées sur le sujet:
 - **A. LABBAIZE** : propose de mettre en place un système de notation qui se base sur 2 approches, une notation par critère (genre du film, acteur, réalisateur,...) et une notation basée sur le choix des utilisateurs. L'idée serait de créer un système se basant sur ces 2 notations.
 - **R. BANEL** : parle de son expérience sur les projets en tant que R1. Trois grands conseils en sont ressortis :
 - La mise en œuvre des outils de gestion de projet est primordiale
 - Les tests sont obligatoires
 - Il faut tout expliciter dans le rapport final, même ce qu'on va remplacer ou effacer
 - **PL. FEULVARCH** : explique que pour gagner en complexité, il serait possible d'utiliser des systèmes de recommandation prédéfinis.

3. Travail pour la semaine prochaine :

- Définir les lots de travail
- Faire un diagramme de Gantt, puis un diagramme de PERT
- Commencer à écrire l'état d'art
- S'informer sur les interfaces graphiques en C (SDL,...) et les structures de données à utiliser

Décisions :

- A. LABBAIZE a été élu à l'unanimité chef de projet, R. BANEL sera quant à lui secrétaire
- Construire un diagramme GANTT/PERT
- Faire une réunion de chantier chaque semaine afin d'éviter l'effet tunnel

Todo list :

Description	Responsable	Délai	Livrable	Validé par
Etat d'art	A.LABBAIZE.	Pour le 30/03/2018	Fichier LaTeX	Toute l'équipe
Se renseigner sur les interfaces graphiques	R. BANEL	Pour le 30/03/2018	-	Toute l'équipe
Réalisation du GANTT	AL/PLF	Pour le 30/03/2018	Diagramme	Toute l'équipe

Table 1: Distribution des tâches

Questions/Remarques :

- Afin de stocker des ressources à trier ou à stocker entre les réunions de projet, un drive d'équipe a été mis en place.

Date de la prochaine réunion : le 30 Mars 2018

Projet C/SD 2018 : Compte-rendu de réunion 2

Minutes for Mars 30, 2018

Present: Rémy BANEL, Ali LABBAIZE , Paul-Louis FEULVARCH

Absent: Aucun

Type de réunion : Réunion de chantier

Durée : 1 heure 30 minutes

Lieu : Salle de travail de l'école

Ordre du jour :

1. Présentation du début de l'état de l'art par A. LABBAIZE
2. Présentation de la bibliothèque graphique SDL par R. BANEL
3. Mise au point sur les structures de données à utiliser

Informations échangées :

1. **A. LABBAIZE :** a présenté 4 systèmes de recommandation différents (cf. état d'art) se basant plus ou moins sur l'utilisateur et la communauté ainsi que le produit recommandé. Une approche hybride permettant d'allier les avantages de chaque système serait une option envisageable afin d'obtenir le meilleur système possible.
Ensuite il a présenté au groupe une proposition concernant les structures de données à utiliser (cf choix de conception dans le rapport), et le calcul matriciel à effectuer sur ces derniers. Afin d'optimiser les temps de recherche, et proposer des résultats plus précis.
2. La mise en évidence des risques temporels a également été abordée, ce qui a aboutit à la décision de faire un autre diagramme de Gantt en fonction de l'avancement du projet, afin de le comparer à la version prévisionnelle, dans l'optique d'une analyse post-mortem.
3. **R. BANEL** propose d'utiliser la bibliothèque SDL afin de réaliser l'interface graphique. Tous les membres du projet devront donc télécharger celle-ci afin de pouvoir l'utiliser. Il sera également nécessaire de se former sur son utilisation.
4. Brainstorming pour remplir une matrice SWOT (sur un document Google Docs)
5. Revue de l'atteinte des objectifs de la Todo list de la réunion précédente :

Description	Responsable	Validation
Écrire l'état de l'art	A.LABBAIZE	Oui.
GANTT	PL.FEULVARCH	Non.
Choix de l'interface graphique	R. Banel	Oui.

Table 1: Atteinte des objectifs

Décisions :

- Mettre en forme la matrice SWOT pour la prochaine réunion.
- Valider les structures de données mentionnées précédemment.
- Mettre en forme le compte-rendu de cette réunion sur ShareLaTeX afin de ne pas prendre de retard.
- Refaire un diagramme de Gantt à un stade plus avancé du projet.

Todo list :

Description	Responsable	Délai	Livrable	Validé par
Mise en forme de la matrice SWOT	R. BANEL	Pour le 10/04/2018	Une matrice SWOT	Toute l'équipe
Faire un diagramme GANTT	PLF	Pour le 10/04/2018	diagramme GANTT	Toute l'équipe
Coder des fonctions d'extraction de données à partir du fichier .txt donné sur Arche	A. LABBAIZE	Pour le 10/04/2018	Code C	Toute l'équipe

Table 2: Distribution des tâches

Date de la prochaine réunion : le 10 Avril 2018

Projet C/SD 2018 : Compte-rendu de réunion 3

Minutes for Avril 10, 2018

Present: Rémy BANEL, Ali LABBAIZE , Paul-Louis FEULVARCH

Absent: Aucun

Type de réunion : Réunion de chantier

Durée : 1 heure 00 minutes

Lieu : Salle de travail de l'école

Ordre du jour :

1. Présentation du code d'extraction des données contenues dans le fichier .txt par A. LABBAIZE
2. Présentation du GANTT par PL. FEULVARCH
3. Mise au point sur les structures de données à garder et à utiliser dans l'extraction du fichier .txt

Informations échangées :

1. **A. LABBAIZE** : a présenté ses idées afin d'extraire les données du fichier texte. Il met en évidence les difficultés qu'il a rencontrées afin de pouvoir séparer le titre du film, son genre, son réalisateur et toute autre information utile.
2. **PL. FEULVARCH** : a présenté le diagramme GANTT qu'il a réalisé. Les lots de travail semblent bien définis, cependant la mise en page de ce GANTT pose problème afin de pouvoir l'intégrer dans le rapport final. Il sera nécessaire de le mettre en forme afin de le garder clair et lisible dans la version finale du rapport.
3. **R. BANEL** a présenté la matrice SWOT.
4. Revue de l'atteinte des objectifs de la Todo list de la réunion précédente :

Description	Responsable	Validation
Fonction d'extraction du fichier .txt	A.LABBAIZE	Partiellement.
GANTT	PL.FEULVARCH	Oui.
SWOT	R. Banel	Oui.

Table 1: Atteinte des objectifs

Décisions :

- Il est important de décider des délimiteurs dans le fichier texte afin de pouvoir obtenir les informations souhaitées. Il a par exemple été décidé de prendre tout ce qui est compris entre le symbole "." et le symbole "(" comme étant des titres de films.
- Refaire le GANTT sous une autre forme afin de le rendre plus lisible.
- Terminer l'état de l'art.
- Commencer à se familiariser avec les interfaces graphiques en essayant de coder quelques fonctions simples afin d'afficher des images et des boutons cliquables.

Todo list :

Description	Responsable	Délai	Livrable	Validé par
Continuer l'état de l'art	R. BANEL	Pour le 20/04/2018	Fichier LateX	Toute l'équipe
Coder une première interface graphique avec SDL	PLF	Pour le 20/04/2018	Code C	Toute l'équipe
Coder des fonctions d'extraction de données à partir du fichier .txt donné sur Arche	A. LABBAIZE	Pour le 20/04/2018	Code C	Toute l'équipe

Table 2: Distribution des tâches

Date de la prochaine réunion : le 20 Avril 2018

Projet C/SD 2018 : Compte-rendu de réunion 4

Minutes for Avril 20, 2018

Present: Rémy BANEL, Ali LABBAIZE , Paul-Louis FEULVARCH

Absent: Aucun

Type de réunion : Réunion d'avancement

Durée : 1 heure 00 minutes

Lieu : Salle PI de l'école

Ordre du jour :

Voir l'avancement du projet, ce qui a été fait, ce qu'il reste à faire. Définir le travail à effectuer par chaque membre pendant les vacances.

Informations échangées :

1. **A. LABBAIZE** : a présenté son code d'extraction des données du fichier texte tout en expliquant aux autres membres comment celui-ci fonctionne. Pour l'instant, celui-ci permet seulement de récupérer les titres du fichier.
2. **PL. FEULVARCH** : a présenté son travail sur l'interface graphique. Il arrive notamment à ouvrir une fenêtre et peut y placer l'image souhaité dessus.
3. **R. BANEL** a récapituler brièvement les différents types de recommandation.
4. Revue de l'atteinte des objectifs de la Todo list de la réunion précédente :

Description	Responsable	Validation
Fonction d'extraction du fichier .txt	A.LABBAIZE	Oui.
Premier pas avec l'interface graphique	PL.FEULVARCH	Oui.
Terminer l'état d'art	R. Banel	Oui.

Table 1: Atteinte des objectifs

Décisions :

- Au vu des différents type de recommandation, il est évident que nous aurons très vite besoin de pouvoir faire des calculs matriciels et notamment des produits de 2 matrices. Il peut donc s'avérer utile de coder ceci.
- Il peut être utile d'avoir une idée de ce à quoi ressemblera l'interface graphique, des données qui seront présente dessus, etc. Il est donc important de continuer à travailler sur le code de celle-ci ainsi que sur son aspect visuel. R. Banel se propose de travailler sur ce dernier point grâce à son expérience des logiciels de photo-montage.
- Il est nécessaire de commencer à ranger les informations pertinentes dans des matrices afin de pouvoir commencer à effectuer des recommandation.

Todo list :

Description	Responsable	Délai	Livrable	Validé par
Coder un produit matriciel	R. BANEL	Pour le 07/05/2018	Code C	Toute l'équipe
Prototype de l'interface graphique	PLF	Pour le 07/05/2018	Code C	Toute l'équipe
Visuels de l'interface graphique	R. BANEL	Pour le 07/05/2018	Image .png	Toute l'équipe
Ordonner les informations importantes dans des matrices	A. LABBAIZE	Pour le 07/05/2018	Code C	Toute l'équipe

Table 2: Distribution des tâches

Date de la prochaine réunion : le 07 Mai 2018

Projet C/SD 2018 : Compte-rendu de réunion 5

Minutes for Mai 15, 2018

Present: Rémy BANEL, Ali LABBAIZE

Absent: Paul-Louis FEULVARCH

Type de réunion : Réunion d'avancement

Durée : 30 minutes

Lieu : Médiathèque de l'école

Ordre du jour :

Mettre en commun le travail après les vacances.

Informations échangées :

1. **A. LABBAIZE** : présente la finalisation du code d'extraction. Toutes les informations importantes sont disponibles. Des | ont dû être ajouté afin d'obtenir les genres (aventures, comédie,...) des films.
2. **R. BANEL** parle des structures à définir. Pose la question de la pertinence de créer une structure matrice.
3. Revue de l'atteinte des objectifs de la Todo list de la réunion précédente :

Description	Responsable	Validation
Fonction d'extraction du fichier .txt	A.LABBAIZE	Oui.
Avancement de l'interface graphique	PL.FEULVARCH	Oui.
Produit matriciel	R. Banel	Oui.

Table 1: Atteinte des objectifs

Décisions :

- Afin d'avancer plus rapidement dans le projet, il a été décidé d'effectuer de réunion de travail. Ceci permettant de travailler à 3 et de pouvoir s'appuyer sur les autres membres du groupe en direct en cas de blocage ou de question.

Todo list :

Description	Responsable	Délai	Livrable	Validé par
Récupérer les affiches des 99 films et séries de la liste	R. BANEL	Pour le 17/05/2018	image	Toute l'équipe
Réfléchir aux différentes structures utiles	A. LABBAIZE	Pour le 17/05/2018	-	Toute l'équipe

Table 2: Distribution des tâches

Date de la prochaine réunion : le 17 Mai 2018

Projet C/SD 2018 : Compte-rendu de réunion 6

Minutes for Mai 23, 2018

Present: Rémy BANEL, Ali LABBAIZE, Paul-LOuis FEULVARCH

Absent:

Type de réunion : Réunion d'avancement

Durée : 30 minutes

Lieu : Salle PI de l'école

Ordre du jour :

Réunion d'avancement.

Informations échangées :

1. **A. LABBAIZE :** présente le calcul de notation des films en fonction des goûts de l'utilisateur. Le système se base donc pour l'instant sur une recommandation objet.
2. **R. BANEL PL. FEULVARCH :** présentent l'interface graphique.
3. Revue de l'atteinte des objectifs de la Todo list de la réunion précédente :

Description	Responsable	Validation
Création des structures film et utilisateur	A.LABBAIZE	Oui.
Transformation des tags en valeurs numériques	A.LABBAIZE	Oui.
Système d'évaluation des films	A.LABBAIZE	Oui.
Création du logo du programme	R. Banel	Oui.
Création des différents prototypes	R. Banel	Oui.
Implémentation de l'interface graphique	PL.FEULVARCHE	Oui.

Table 1: Atteinte des objectifs

Décisions :

- Le programme doit être terminé pour le 26 mai, afin de laisser quelques jours pour réaliser la phase de test et corriger d'éventuels bugs.