

Université de Montréal

Mures : Un système de recommandation de musique

par

Octavian Rolland Arnautu

Département d'Informatique et de Recherche Opérationnelle
Faculté des arts et des sciences

Mémoire présenté à la Faculté des arts et des sciences
en vue de l'obtention du grade de Maitrise ès Sciences
en Informatique

Juillet, 2012

© Octavian Rolland Arnautu, 2012

Université de Montréal
Faculté des arts et des sciences

Ce mémoire est intitulé :

Mures : Un système de recommandation de musique

Présenté par :
Octavian Rolland Arnautu

a été évalué par un jury composé des personnes suivantes :

El Mostapha Aboulhamid, président-rapporteur
Esmâ Aïmeur, directrice de recherche
Jian-Yun Nie, membre du jury

Résumé

Pendant la dernière décennie nous avons vu une transformation incroyable du monde de la musique qui est passé des cassettes et disques compacts à la musique numérique en ligne. Avec l'explosion de la musique numérique, nous avons besoin de systèmes de recommandation de musique pour choisir les chansons susceptibles d'être appréciés à partir de ces énormes bases de données en ligne ou personnelles. Actuellement, la plupart des systèmes de recommandation de musique utilisent l'algorithme de filtrage collaboratif ou celui du filtrage à base de contenu. Dans ce mémoire, nous proposons un algorithme hybride et original qui combine le filtrage collaboratif avec le filtrage basé sur étiquetage, amélioré par la technique de filtrage basée sur le contexte d'utilisation afin de produire de meilleures recommandations. Notre approche suppose que les préférences de l'utilisateur changent selon le contexte d'utilisation. Par exemple, un utilisateur écoute un genre de musique en conduisant vers son travail, un autre type en voyageant avec la famille en vacances, un autre pendant une soirée romantique ou aux fêtes. De plus, si la sélection a été générée pour plus d'un utilisateur (voyage en famille, fête) le système proposera des chansons en fonction des préférences de tous ces utilisateurs. L'objectif principal de notre système est de recommander à l'utilisateur de la musique à partir de sa collection personnelle ou à partir de la collection du système, les nouveautés et les prochains concerts. Un autre objectif de notre système sera de collecter des données provenant de sources extérieures, en s'appuyant sur des techniques de *crawling* et sur les flux RSS pour offrir des informations reliées à la musique tels que: les nouveautés, les prochains concerts, les paroles et les artistes similaires. Nous essayerons d'unifier des ensembles de données disponibles gratuitement sur le Web tels que les habitudes d'écoute de *Last.fm*, la base de données de la musique de *MusicBrainz* et les étiquettes des *MusicStrands* afin d'obtenir des identificateurs uniques pour les chansons, les albums et les artistes.

Mots-clés : Systèmes de Recommandation de Musique, Filtrage Collaboratif, Filtrage a Base de Contenu, Flux RSS, Systèmes de Recommandation Hybrides, Filtrage Contextuel

Abstract

In the last decade we have seen an incredible transformation of the world of music, from cassette tapes and compact discs to online digital music. With the explosion of the digital music we need music recommender systems to select those probably preferred songs from these huge online or personal databases. Currently, most music recommender systems use either collaborative (social) filtering or content-based algorithms. In this work we propose an original hybrid collaborative and tag-based algorithm improved by the context-of-use filtering technique in order to generate better playlists. Our approach assumes that user preferences change depending on the context of use. For example, a user listen one kind of music while driving to work, another type while traveling with the family on vacation, another one in a romantic evening or at parties. Additionally, if the playlist was generated for more than one user (family trip, party) the system will propose songs depending on the preferences of all these users. The main goal of our system is to recommend music to the user from the personal music collection or new music from system music collection, new releases and incoming concerts. Another goal of our system it will be to gather data from external sources based on crawling techniques and RSS Feeds to provide music related content like: new releases, incoming concerts, lyrics, similar artists. We'll try to interlink some free available datasets on the web like listening habits from *Last.fm*, music database from *MusicBrainz* and tags from *MusicStrands* in order to obtain unique identifiers for songs, albums and artists.

Keywords: Music Recommender Systems, Collaborative Filtering, Content-Based Filtering, RSS Feeds, Hybrid Recommendation, Context-Filtering

Table des matières

Résumé.....	i
Abstract.....	ii
Liste des tableaux.....	v
Liste des figures.....	vi
Remerciements	vii
Chapitre 1 Introduction.....	1
Chapitre 2 Les systèmes de recommandation	8
2.1 Introduction.....	8
2.2 Représentation du profil d'utilisateur	10
2.3 Exploitation du profil d'utilisateur : la recommandation.....	12
2.3.1 Filtrage collaboratif.....	15
2.3.2 Filtrage démographique.....	21
2.3.3 Filtrage à base de contenu.....	21
2.3.4 Filtrage contextuel.....	23
2.3.5 Les méthodes hybrides	28
2.4 Conclusion	30
Chapitre 3 La recommandation de musique.....	32
3.1 Introduction.....	32
3.2 Les cas d'utilisation	33
3.2.1 La recommandation des artistes	34
3.2.2 La recommandation des voisins.....	34
3.2.3 La recommandation des listes de lecture.....	35
3.3 Le profil d'utilisateur	36
3.3.1 Les types d'auditeurs	36
3.3.2 Les types de représentation proposées.....	37
3.4 Le profil de la musique.....	43
3.4.1 Les métadonnées des éditeurs	43

3.4.2	Les métadonnées culturelles.....	44
3.4.3	Les métadonnées acoustiques.....	44
3.5	Les méthodes de recommandation.....	46
3.5.1	Le filtrage collaboratif : Ringo, Racofi.....	46
3.5.2	Le filtrage à base de contenu : manuel, automatique.....	48
3.5.3	Le filtrage contextuel : PAPA	50
3.5.4	Les hybrides : Genius, LastFm	52
3.6	Conclusion	56
Chapitre 4 Conception de Mures		57
4.1	Présentation générale	57
4.2	L'architecture du système.....	60
4.3	La gestion de ressources musicales	63
4.4	La gestion de profils d'utilisateurs	67
4.5	Les options de recommandation	70
4.5.1	Le système de recommandation.....	70
4.5.2	L'échelle d'évaluation	78
4.5.3	Approches pour choisir le type d'algorithme de filtrage collaboratif	78
4.6	Conclusion	85
Chapitre 5 Implémentation et validation		87
5.1	Implémentation de Mures	87
5.2	Environnement d'utilisation	90
5.3	Comparaison avec les applications similaires	96
5.4	Stratégies d'évaluation	98
5.4.1	L'évaluation des mesures centrées sur le système.....	98
5.4.2	L'évaluation des mesures centrées sur l'utilisateur	104
Conclusion		113
Bibliographie.....		116

Liste des tableaux

Table 2.1 Forces et faiblesse des méthodes traditionnelles, adaptées de (Burke, 2002)	24
Table 2.2 Les méthodes hybrides, adaptées de (Burke, 2002)	30
Table 2.3 Résumé des éléments impliqués dans le problème de recommandation.....	31
Table 4.1 Exemple d'une matrice d'évaluations classique Usagers x Items	79
Table 4.2 Le voisinage d'utilisateur Alice	81
Table 4.3 Le voisinage d'artiste Cher	83
Table 5.1 Comparaison avec les applications similaires	96
Table 5.2 Exemple de calcul de MAE et RMSE pour un usager.....	102
Table 5.3 Le vote moyen pour les recommandations reçues (contexte).....	107
Table 5.4 Le vote moyen pour les recommandations reçues (jeux de données).....	110

Liste des figures

Figure 2.1 Exemple d'un ensemble d'apprentissage sur le site Ilike.com	11
Figure 3.1 Exemple d'un profil d'utilisateur MPEG 7	39
Figure 3.2 Exemple d'un profil d'utilisateur UMIRL (Chai <i>et al.</i> , 2000)	40
Figure 3.3 FOAF exemple (Bauer <i>et al.</i> , 2012)	41
Figure 3.4 FOAF exemple (Giasson <i>et al.</i> , 2007)	42
Figure 3.5 La quantité d'informations gérées par Last.fm (Celma <i>et al.</i> , 2007)	55
Figure 4.1 Représentation des interactions de Mures dans le Web	59
Figure 4.2 Architectures du système de Mures	60
Figure 4.3 Lecteur mp3 de Mures	62
Figure 4.4 Représentation graphique des tags ID3	64
Figure 4.5 Exemple simplifié d'une réponse au format XML	66
Figure 4.6 Feedback implicite et explicite dans Mures	69
Figure 4.7 Le module de recommandation dans Mures	71
Figure 4.8 La table activity_tag dans Mures	72
Figure 4.9 La liste de tags pour un contexte d'utilisation	73
Figure 4.10 La liste de genres musicaux disponibles dans Mures	73
Figure 4.11 La table user_chanson_votes dans Mures	74
Figure 4.12 Les artistes préférés de l'utilisateur testfoaf	75
Figure 4.13 Les meilleures chansons de l'artiste Cher	75
Figure 5.1 Architecture 3-tiers de Mures	88
Figure 5.2 Les fonctionnalités de Mures	90
Figure 5.3 Enregistrement du nouvel usager	91
Figure 5.4 Gestion de la collectionne personnelle	92
Figure 5.5 Liste de chansons pour écouter en voiture	93
Figure 5.6 Liste de chansons des artistes similaires	94
Figure 5.7 Comparaison de RMSE	101

Figure 5.8 Comparaison de MAE	101
Figure 5.9 L'analyse de données avec SPSS (pour RMSE).....	103
Figure 5.10 Formulaire pour la validation globale de Mures	106
Figure 5.11 L'analyse de données avec SPSS (pour le contexte).....	108
Figure 5.12 L'analyse de données avec SPSS (pour les jeux de données).....	111

À toute ma famille.

Remerciements

C'est avec un grand plaisir que j'apporte ce témoignage écrit de ma reconnaissance à tous ceux qui m'ont gratifié de leur soutien et de leur confiance pendant mes études.

Je tiens remercier tout d'abord à ma directrice de recherche, Professeure Esma Aïmeur, qui m'a encadré tout au long de la réalisation de ce mémoire. Je vous remercie pour votre grande disponibilité, votre écoute et votre patience. Vos commentaires et vos critiques constructives sur mes présentations orales et mes écrits m'ont permis de m'améliorer continuellement. Merci pour tout!

Je remercie également mes collègues de notre laboratoire Héron qui ont su créer une belle et chaleureuse ambiance et une expérience académique des plus riches, surtout pendant les nombreuses réunions d'équipe. Je voudrais faire une mention spéciale à Fodé Touré qui a toujours su me rendre service avec le plus grand des plaisirs. Sa générosité a été fortement appréciée.

J'adresse tous mes remerciements à Monsieur El Mostapha Aboulhamid pour l'honneur qu'il m'a fait en acceptant d'être président-rapporteur de mon mémoire et à Monsieur Jian-Yun Nie. Je vous remercie d'avoir accepté de donner de votre précieux temps afin d'évaluer et critiquer mon travail.

Je tiens remercier mon ami Mihai Florea de m'avoir incité à poursuivre mes études au cycle supérieur. Merci pour ton aide et tes judicieux conseils académiques. Je remercie également mon patron Patrice Béliveau pour la bonne collaboration afin de bien jumeler le travail avec les études.

Enfin, je remercie ma famille, et plus particulièrement ma femme et mes filles de m'avoir encouragé à faire ma maîtrise, malgré les sacrifices à faire. Leur encouragements et leur support mon permis de ne jamais abandonner.

Que toutes ces personnes trouvent ici l'expression de ma grande gratitude et sympathie !

Rolland Octavian Arnautu

Chapitre 1 Introduction

La musique est un art, celui de la muse Euterpe¹ dans la mythologie grecque. Elle est donc à la fois une création (une œuvre d'art), une représentation et aussi un mode de communication (URL, 1).

En ce moment, dans les maisons, les bureaux, les voitures, les restaurants et les clubs dans le monde, les gens écoutent de la musique. La musique a existé dans toutes les sociétés humaines, depuis la préhistoire. Elle est à la fois une forme d'expression humaine individuelle (notamment l'expression des sentiments), une source de rassemblement collectif et de plaisir (fête, danse) et un symbole d'une communauté ou d'une nation (hymne national, style musical officiel, musique religieuse, musique militaire) (URL, 1).

Les manières dont nous entendons, écoutons, apprécions, et utilisons la musique ont changé radicalement pendant les derniers deux siècles. Au 19^{ème} siècle, la musique en direct pourrait être entendue seulement dans les propriétés privées ou dans les salles de concert publiques (Hargreaves *et al.*, 2004), comme les théâtres, les tavernes et les autres emplacements des réunions sociales. À ce moment là, la musique était considérée comme un trésor de grande valeur, et sa valeur était intrinsèquement liée à des contextes clairement définis.

Le développement des mass-médias au XX^e a changé le statut « *trésor* » de la musique pour le rendre facilement disponible. La musique est ainsi devenue une *marchandise* qui est produite, distribuée et consommée comme n'importe quelle autre. Un autre changement important était que les contextes dans lesquels la musique était écoutée par les gens sont devenus bien plus variés et diversifiés. Pendant la dernière décennie nous avons vu une transformation incroyable du monde de la musique qui est passé des cassettes et disques compacts à la musique numérique en ligne. Puisque la musique de différents

¹ Dans la mythologie grecque, Euterpe était la muse qui présidait à la Musique. C'est une jeune fille couronnée de fleurs et jouant de la flûte. Des papiers de musique, des hautbois et autres instruments sont auprès d'elle.

styles et genres est maintenant largement disponible par l'intermédiaire du baladeur numérique, de l'Internet et d'autres médias, on peut soutenir que les gens utilisent la musique de façon active dans une mesure beaucoup plus grande que dans le passé. La musique peut maintenant être considérée comme une *ressource* plutôt que simplement comme une marchandise. Les gens peuvent consciemment et activement l'utiliser dans des situations différentes à différents niveaux d'engagement.

La migration du monde traditionnel vers l'Internet, a rendu l'accès à l'information plus facile et plus rapide que jamais. Les fans d'aujourd'hui peuvent avoir accès et payer pour la musique par différents moyens - achat de chansons ou d'albums dans les magasins de téléchargement en ligne ou en utilisant les services d'abonnement, achat d'applications mobiles pour la musique, et écoute de la musique de façon gratuite à partir de plusieurs sites (URL, 2). Le choix des consommateurs a été transformé : les maisons de disques ont mis en ligne plus de 11 millions de chansons disponibles à partir d'environ 400 services légaux de musique dans le monde entier.

En 2009, pour la première fois, plus d'un quart du chiffre d'affaires global de l'industrie de la musique enregistrée (27 %) provenait de ventes en ligne, un marché d'une valeur estimée à 4,2 milliards de dollars, en hausse de 12 pour cent par rapport à 2008. Aux États-Unis, le plus grand marché de musique du monde, les revenus en ligne comptent pour environ 40 pour cent de ventes de musique (URL, 3).

De la même façon, les collections personnelles de musique se sont développées, favorisées par des améliorations technologiques des réseaux, du stockage, des dispositifs et des services d'Internet. La quantité croissante de musique disponible rend très difficile, pour l'utilisateur de trouver la musique qu'il souhaite écouter.

Pour trouver les pièces musicales préférées en utilisant les systèmes de recherche, il faut qu'on exécute des requêtes à plusieurs reprises. Par conséquent, nous sommes souvent en difficulté de trouver les requêtes appropriées. Pour résoudre ce problème, il est souhaitable d'utiliser les Systèmes de Recommandation (SR) de musique pour choisir les chansons probablement préférées.

Les systèmes de recommandation ont été introduits comme une technique intelligente pour faire le filtrage de l'information afin de présenter les éléments d'information qui sont susceptibles d'intéresser l'utilisateur. Ils peuvent être utilisés pour fournir efficacement des services personnalisés dans la plupart des domaines de commerce électronique, en tant que client ou commerçant. Les systèmes de recommandation sont bénéfiques pour le client en lui faisant des suggestions sur les produits susceptibles d'être appréciés. En même temps, l'entreprise va bénéficier de l'augmentation des ventes qui se produit normalement quand on présente au client plus d'articles susceptibles d'être aimés (Margaritis *et al.*, 2003).

Les deux entités de base qui apparaissent dans n'importe quel système de recommandation sont *l'utilisateur* et *l'article* (*la chanson* pour la musique). Les données d'entrée pour un système de recommandation dépendent du type de l'algorithme de filtrage employé. Généralement, elles appartiennent à l'une des catégories suivantes :

- Les *estimations* (également appelées les *votes*), qui expriment l'opinion des utilisateurs sur les articles. Elles sont normalement fournies de façon explicite par l'utilisateur et suivent une échelle numérique spécifique (exemple : 1 mauvais à 5 excellent). Les estimations peuvent également être collectées implicitement à partir de l'histoire d'achat de l'utilisateur, des logs Web, des habitudes de lecture et d'écoute;
- Les *données démographiques*, qui se réfèrent à des informations telles que l'âge, le sexe et l'éducation des utilisateurs. Ce type de données est généralement difficile à obtenir et est normalement collecté explicitement;
- Les *données de contenu*, qui sont fondées sur une analyse textuelle des documents liés aux éléments évalués par l'utilisateur. Les caractéristiques extraites de cette analyse sont utilisées comme entrées dans l'algorithme de filtrage afin d'en déduire un profil d'utilisateur (Margaritis *et al.*, 2003).

Actuellement, la plupart des systèmes de recommandation de musique utilisent des algorithmes de Filtrage Collaboratif (FC) ou de Filtrage à Base de Contenu (FBC). En outre, la plupart des systèmes de recommandation de musique actuels sont fondés sur

l'approche de filtrage collaboratif : *Last.fm*, *MyStrands*, *MusicMobs*, *iRate*, basé sur le système *Racofi*². L'idée fondamentale de ces systèmes est de garder la trace des artistes (et des chansons) qu'un utilisateur écoute à travers des applications telles que *iTunes*, *Winamp*, de rechercher d'autres utilisateurs avec des goûts semblables et de recommander finalement des artistes (ou des chansons) à l'utilisateur, selon le goût de ces utilisateurs semblables. La méthode consiste à faire usage du feed-back d'utilisateur afin d'améliorer la qualité du matériel recommandé : feed-back explicite, les votes d'utilisateurs et feed-back implicite : à partir des habitudes d'écoute de l'utilisateur.

En s'appuyant sur les chansons favoris des utilisateurs, les méthodes à base de contenu recommandent des chansons semblables en termes de contenu musical (tels que les humeurs, les rythmes, les paroles). À l'heure actuelle, la méthode la plus commune d'accès à la musique est par le biais des métadonnées textuelles. Il y a deux approches pour collecter les métadonnées : manuellement ou automatiquement. Pour l'approche automatique il y a plusieurs systèmes, mais les résultats ne sont pas très bons. L'annotation de l'homme sur la musique prend beaucoup de temps, mais peut être plus précise que les méthodes d'extraction automatique. L'approche de *Pandora* est basée sur la description manuelle du contenu audio.

Afin de fournir les meilleures recommandations musicales, il y a plusieurs tentatives de combiner le filtrage à base de contenu avec le filtrage collaboratif. Les systèmes de recommandation hybrides combinent deux ou plusieurs techniques de recommandation pour obtenir de meilleures performances avec moins d'inconvénients. Généralement, il y a deux méthodes principales pour travailler avec la recommandation hybride : la combinaison des sorties individuelles des deux techniques ou d'intégrer les deux dans un algorithme unique (Cai *et al.*, 2009).

Sur le plan technologique, il existe différents sites Web ou applications qui implémentent certaines des méthodes précédentes. Ces applications, avec des niveaux de

² RACOFI Musique est un site de recommandation multidimensionnelle de musique, avec un accent exclusif sur le contenu canadien

complexité différents, varient des prototypes réalisés par les chercheurs, à des solutions plus complètes, gratuites ou commerciales, réalisées par l'industrie musicale. Ce mémoire présente une étude détaillée de ces applications et une catégorisation selon les méthodes de filtrages employées. Chaque catégorie est illustrée par des études de cas représentatifs (applications vedettes) afin de montrer leurs points forts et leurs faiblesses par rapport à nos besoins.

En se basant sur le goût musical (préférences) des utilisateurs, un Système de Recommandation de Musique (SRM) a comme premier objectif de proposer des artistes d'intérêt, qu'ils soient connus ou inconnus de l'utilisateur, ainsi que leurs chansons (si disponibles).

Un autre objectif d'un système de recommandation de musique devrait être la capacité d'obtenir dynamiquement de nouvelles informations connexes telles que :

- les artistes similaires à ceux que l'utilisateur préfère;
- les nouvelles sorties musicales à partir d'*iTunes*, *Amazon*, *Yahoo*;
- les critiques d'albums de nos artistes favoris et d'artistes recommandés;
- les mp3-blogs pour téléchargement de musique;
- les sessions podcast pour écouter / télécharger;
- les listes de lecture basées sur la similarité;
- les prochains concerts près de chez nous. (Celma *et al.*, 2005).

Originalité et contribution

Dans ce mémoire, nous proposons un **algorithme hybride** et original qui combine le *filtrage collaboratif* avec le *filtrage basé sur l'étiquetage*, amélioré par la technique de *filtrage basée sur le contexte d'utilisation* afin de produire les meilleures recommandations. L'objectif principal de notre système est de recommander à l'utilisateur de la musique à partir de sa collection personnelle ou de la nouvelle musique à partir de la collection du système, des nouveautés ou des concerts suivis. Notre approche suppose que les préférences de l'utilisateur changent selon le contexte d'utilisation. Par exemple, un utilisateur écoute un genre de musique en conduisant vers son travail, un autre type en voyageant avec la famille en vacances, un autre pendant une soirée romantique, aux fêtes, etc. De plus, si la sélection de chansons a été générée pour plus d'un utilisateur (voyage en famille, fête) le système proposera des chansons en fonction des préférences de tous ces utilisateurs.

Un autre objectif de notre système est de collecter des données provenant de sources extérieures, en s'appuyant sur les *techniques de crawling* et sur les *flux RSS* pour offrir des informations reliées à la musique tels que : les nouveautés musicales, les prochains concerts, les paroles et les artistes similaires. Les informations pour ces sources extérieures ne seront pas limitées à quelques sites web bien connues parce que **le système va essayer de trouver toujours de nouvelles sources** (sites avec un contenu musical spécifique à une zone géographique spécifique : musique en provenance d'un pays particulier (la Roumanie), dans une langue spécifique (en espagnol par exemple)).

Nous lions **des ensembles de données** disponibles gratuitement sur le Web comme les *habitudes d'écoute* de *Last.fm*, la *base de données de musique* de *MusicBrainz* et les *étiquettes* des *MusicStrands* afin d'obtenir des identifiants uniques pour les chansons, les albums et les artistes. Jusqu'à présent, les informations que l'une de ces applications gère ne considèrent pas les informations en provenance d'un autre système.

De façon générale, les recommandations de notre système seront générées autant que possible pour *révéler un état d'esprit* comme heureux, triste ou romantique, pour

accompagner une activité telle que fête, travail, conduite en proposant les *chansons favorites* d'utilisateur, mélangées avec des *chansons nouvelles* susceptibles d'être aimées par l'utilisateur.

Organisation de ce mémoire

Ce mémoire est organisé comme suit : le chapitre 2 offre un survol des systèmes de recommandation. Ce survol présente l'histoire des systèmes de recommandation, suivie de la présentation de plusieurs logiques de classification bien connues dans ce domaine. À la fin, nous introduisons la recommandation basée sur le filtrage contextuel pour mettre en relief la classe à laquelle s'apparente notre méthode. Le chapitre 3 représente un état de l'art des systèmes de recommandation de musique, présente et discute des solutions susceptibles de répondre aux besoins de la recherche et préalablement analysées. Le chapitre 4 présente *Mures*, notre solution pour combler les besoins présentés au chapitre 3. Nous décrivons l'architecture du système et les modules implémentées. Le chapitre 5 présente la validation globale de notre système, ainsi que les résultats de tests et comparaisons des différents types d'algorithmes utilisés. Tout cela après avoir présenté les aspects technologiques entourant l'implémentation de notre système ainsi que la description des différentes interfaces de l'application. Le dernier chapitre conclut ce mémoire et présente quelques perspectives futures.

Chapitre 2 Les systèmes de recommandation

Ce mémoire présente un système de recommandation de musique. Pour mieux comprendre notre apport dans le domaine des systèmes de recommandation nous ferons un bref survol du domaine. L'intention de ce chapitre est de présenter les éléments de l'état de l'art organisés dans une classification simple, d'expliquer les méthodes utilisées et de décrire leurs avantages et inconvénients. Ainsi, l'objectif principal est de fournir un point de départ pour notre travail afin de construire notre système de recommandation, de choisir notre classification et la terminologie qui sera utilisée tout au long de ce mémoire.

2.1 Introduction

Avec la croissance de popularité des applications web et l'explosion de l'utilisation de l'Internet pendant les années '90 nous avons senti le besoin de filtrer cette énorme quantité d'informations. La raison pour laquelle les gens pourraient être intéressés à utiliser un système de recommandation est qu'ils ont tant d'éléments à choisir dans une période limitée de temps et ils ne peuvent pas évaluer toutes les options possibles. De manière générale, pour atteindre cet objectif, les utilisateurs doivent fournir leur propre profil et ils vont recevoir une image réduite et personnalisée de ces informations. À première vue, ils ressemblent aux moteurs de recherches d'informations. La différence est que la sémantique des moteurs de recherche est basée sur le "*match*" : renvoie tous les éléments qui correspondent à la requête, classés par degré de *match*. L'objectif de la recommandation est de retourner un contenu personnalisé, intéressant et utile (Burke, 2002).

Aujourd'hui, le domaine d'application de ces systèmes de recommandation est très large. Ils font partie intégrante des sites de commerce électronique. Les articles les plus recommandés sont les films, les livres, les chansons et les photos. Les services les plus recommandés sont les restaurants, les hôtels et les pages Web.

Histoire et définitions

Les racines des systèmes de recommandation remontent aux travaux étendus dans les sciences cognitives, la théorie d'approximation, la recherche d'informations, la théorie de la prévoyance et ont également des liens avec la science de la gestion et le marketing, dans la modélisation des choix du consommateur (Adomavicius & Tuzhilin 2005, Naak, 2009).

« *Information Lens System* » (Malone *et al.*, 1987) peut être considéré comme le premier système de recommandation. À l'époque, l'approche la plus commune pour le problème de partage d'informations dans l'environnement de messagerie électronique était la liste de distributions basée sur les groupes d'intérêt. La première définition pour le filtrage a été donnée aussi par Malone : « Même si le terme a une connotation littérale de laisser les choses dehors (*filtrage négatif : enlèvement*), nous l'utilisons ici dans un sens plus général qui consiste à sélectionner les choses à partir d'un ensemble plus large de possibilités (*filtrage positif : sélection*) ».

Les systèmes de recommandation sont devenus un important domaine de recherche avec la publication des premiers articles dans le domaine du filtrage collaboratif. La littérature académique a introduit le terme de filtrage collaboratif par le système Tapestry (Goldberg *et al.*, 1992) qui a permis aux utilisateurs de créer des requêtes permanentes, basées sur les annotations des utilisateurs. Quelques années plus tard, un certain nombre de systèmes académiques de recommandation ont été introduits dans les domaines de la *musique* (Shardanand, 1994; Maes et Shardanand, 1995), des *livres* (Resnick *et al.*, 1994), des *vidéos* (Furnas *et al.*, 1995), des *films* (Fisk, 1996), des *pages Web* (Das *et al.*, 1997, Polanco *et al.*, 1997), des *articles de nouvelles Usenet* (Resnick *et al.*, 1994, Konstan *et al.*, 1997) et des *liens Internet* (Terveen *et al.*, 1997).

Intuitivement, un système de recommandation musical peut être comparé avec un « *Disk Jockey* » (*DJ*) qui prépare la musique pour un événement. Le *DJ*, avec sa vaste expérience dans la musique, choisira facilement, des chansons appropriées pour le type d'événement : mariage, baptême, fête. La connaissance du profil des participants à

l'événement sera un atout. Le *DJ* proposera une liste de lecture lente si les participants aux événements sont des personnes âgées, une musique dynamique pour une fête des jeunes, ou de la musique d'un pays particulier si les participants font partie d'une minorité ethnique.

Les deux entités de base qui apparaissent dans tous les systèmes de recommandation sont l'utilisateur et l'item. L'utilisateur est la personne qui utilise un système de recommandation, donne son opinion sur divers items et reçoit les nouvelles recommandations du système (Margaritis *et al.*, 2003). Les utilisateurs doivent être modélisés de telle sorte que le système puisse exploiter leurs profils et préférences. Par ailleurs, une description précise des articles est également cruciale pour obtenir de bons résultats au moment des recommandations d'articles pour les utilisateurs.

2.2 Représentation du profil d'utilisateur

Pour décrire les préférences de l'utilisateur il y a 3 éléments : *la génération, la maintenance et l'exploitation* des profils d'utilisateurs, en utilisant un algorithme de recommandation (Lopez *et al.*, 2003).

Dans sa thèse, « *Music Recommendation and Discovery in the Long Tail* », Celma (2008) a bien expliqué toutes ces étapes. Un des aspects les plus importants d'un profil d'utilisateur est son *initialisation*. Il existe plusieurs méthodes :

- *profil vide* : le plus simple est de créer un profil vide, qui sera mis à jour dès que l'utilisateur interagit avec le système;
- *manuellement* : le système peut demander aux utilisateurs d'enregistrer leurs intérêts (par tags, mots clés), ainsi que des informations démographiques (âge, statut matrimonial, sexe, etc.), des données géographiques (ville, pays, etc.) et des données psychographiques (intérêts, style de vie, etc.);

- *import* : le système peut demander à l'utilisateur des informations disponibles à travers de sources externes (le profil obtenu à partir de l'API³ du site *Last.fm*);
- *ensemble d'apprentissage* : l'utilisateur doit fournir un feedback aux articles bien précis, en les marquant comme pertinents ou non à ses intérêts (Figure 2.1);
- *stéréotype* : un nouvel utilisateur sera attribué à un groupe d'utilisateurs similaires, qui sont représentés par leur stéréotype, selon certaines informations démographiques, géographiques, ou psychographiques;

Tell us which artists you like

We'll recommend you new music and calculate your music compatibility with your friends on iLike. The more artists you rate, the better.

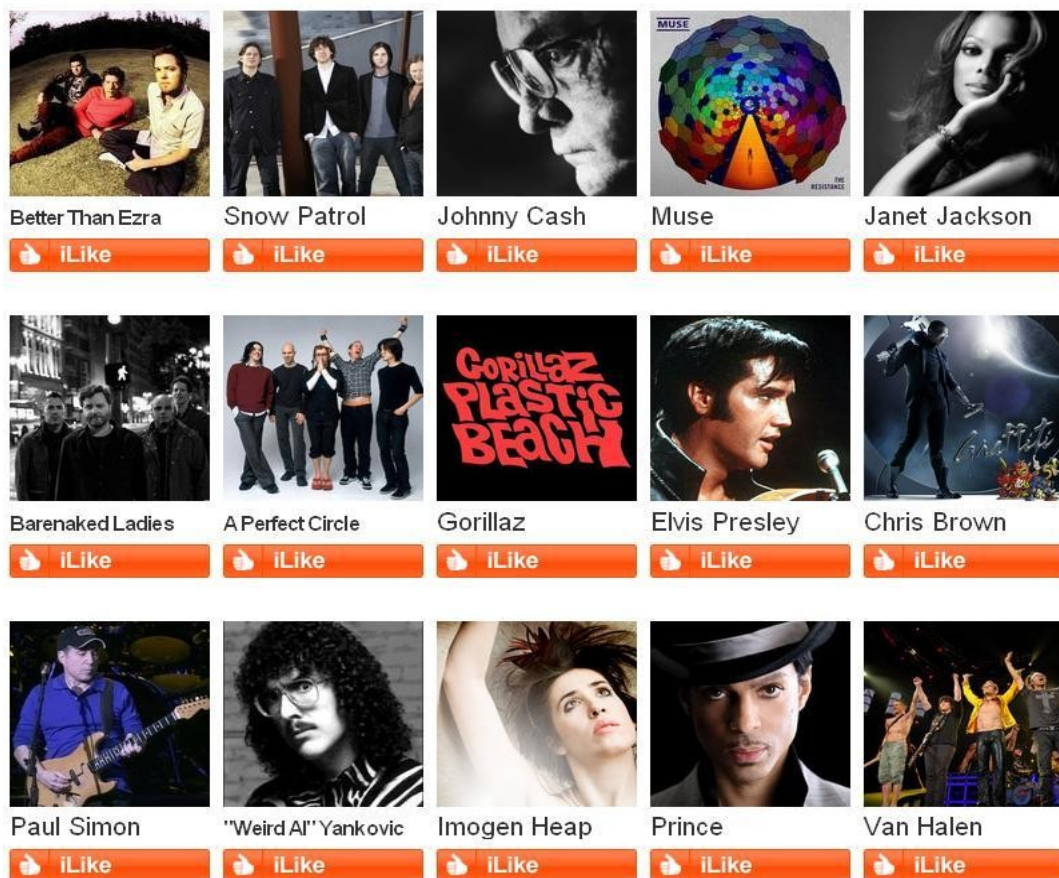


Figure 2.1 Exemple d'un ensemble d'apprentissage sur le site Ilike.com

³ API = Application Programming Interface

La *maintenance* signifie qu'une fois que le profil est initialisé, il sera mis à jour en fonction des actions et préférences de l'utilisateur. Le feedback d'utilisateur peut être explicite ou implicite. Le *feedback explicite* vient généralement sous la forme d'estimations. Ce type de feedback peut être positif ou négatif. Les estimations peuvent être dans une échelle discrète (par exemple de 0 à N), ou d'une valeur binaire (aime/n'aime pas). Une autre manière de recueillir le feedback explicite est de permettre aux utilisateurs d'écrire des commentaires et des avis au sujet des articles.

Un système peut également recueillir le *feedback implicite* en surveillant les actions de l'utilisateur, en analysant : l'histoire d'achats, le temps passé sur une page Web, les liens suivis de l'utilisateur, les habitudes d'écoute (déplastant si on joue la musique, la pause, le saut et l'arrêt d'une chanson).

2.3 Exploitation du profil d'utilisateur : la recommandation

Une fois le profil d'utilisateur créé, la prochaine étape est d'exploiter les préférences des utilisateurs afin de leur proposer des recommandations intéressantes. Depuis l'émergence du domaine des systèmes de recommandation, plusieurs approches et méthodes ont été proposées, certaines ont été étudiées, expérimentées et comparées [Naak 2009]. Certains chercheurs se sont intéressés à la classification de ces méthodes et proposent une taxonomie ou terminologie unifiée. Nous citons les travaux de (Celma 2008; Adomavicius & Tuzhilin 2005; Burke 2002; Pazzani 1999; Resnick & Varian 1997; Konstan *et al.*, 1999, Candillier *et al.*, 2009). Bien que plusieurs terminologies soient utilisées pour désigner une même méthode ou approche, il existe cependant un consensus en ce qui concerne la formalisation du problème de recommandation.

Pour Bourke les plus importants sont les sources de données sur lesquelles la recommandation est basée et comment ces données sont exploitées :

- *les données préalables* : les informations que le système possède avant que le processus de recommandation ne commence;

- *les données d'entrée* : les informations que l'utilisateur donne au système pour recevoir une recommandation;
- *l'algorithme* qui combine les données préalables et les données d'entrée pour fournir la recommandation.

Basé sur ces trois composantes, Burke trouve cinq types différents de systèmes de recommandation : les méthodes basées sur *le filtrage collaboratif*, les méthodes *basées sur le contenu*, le *filtrage démographique*, le *filtrage à base d'utilité* et le *filtrage à base de connaissance*. Parmi ces derniers, les systèmes basés sur *le filtrage collaboratif*, *basés sur le contenu* et le *filtrage à base de connaissance* sont les plus utilisés.

Pour Adomavicius et Tuzhilin le problème de recommandation est réduit au problème d'estimer les votes pour les articles qui n'ont pas été vus par un utilisateur. Intuitivement, cette évaluation est habituellement basée sur les estimations données par cet utilisateur à d'autres articles.

Une des originalités de cette approche réside dans le fait que les auteurs ont proposé une formulation du problème de la recommandation comme suit (Naak 2009) :

$$\forall c \in C, \quad s'_c = \arg \max_{s \in S} u(c, s)$$

$$u : C \times S \rightarrow R$$

C : ensemble représentant tous les usagers

S : ensemble de tous les items qui sont recommandables

u : une fonction d'utilité qui mesure l'utilité de l'item s pour l'utilisateur c

R : ensemble totalement ordonné

Le problème de la recommandation revient à trouver pour chaque usager $c \in C$, un certain item $s' \in S$, qui va maximiser la fonction d'utilité $u : C \times S \rightarrow R$. Dans les systèmes de recommandation, l'utilité d'un item est exprimée par une évaluation sur une échelle de

valeurs définie qui représente l'ensemble R . Dans les systèmes de recommandation l'utilité d'un article est habituellement représentée par une estimation, qui indique comment un utilisateur particulier a aimé un article particulier.

Le problème central des systèmes de recommandation réside dans le fait que la fonction d'utilité u n'est pas habituellement définie sur l'espace entier de $C \times S$, mais seulement sur un certain sous-ensemble de celui. Ceci signifie que u doit être extrapolé à l'espace entier $C \times S$. Les prédictions qui obtiennent les meilleurs scores feront l'objet de la recommandation.

En ce qui concerne la classification des systèmes de recommandation pour Adomavicius et Tuzhilin il existe trois catégories : les méthodes basées sur *le filtrage collaboratif*, les méthodes *basées sur le contenu* et les méthodes *hybrides*.

Pour Celma, le problème de recommandation peut être divisé en deux sous-problèmes. Le premier est un problème de prévision, et concerne de l'évaluation de la probabilité des articles susceptibles à être aimés par un utilisateur donné. Le deuxième problème est de recommander une liste de N articles, en supposant que le système puisse prévoir la probabilité pour les articles non évalués.

En fait, le problème le plus pertinent est l'estimation. Une fois que le système peut estimer les éléments dans un ensemble totalement ordonné, le problème de la recommandation se réduit à énumérer les premiers N articles avec la plus haute valeur estimée.

Pour Celma, l'exploitation du profil d'utilisateur est étroitement liée à la méthode de filtrage d'information. La méthode adoptée pour le filtrage des informations a conduit à la classification standard des systèmes de recommandation, qui est : *filtrage collaboratif*, *filtrage démographique*, *filtrage basé sur le contenu*, filtrage contextuel et les *approches hybrides*.

Toutes ces sections présentent les méthodes de recommandation pour un seul utilisateur. Il est intéressant de mentionner qu'un autre type de recommandeurs (*basés sur groupe*) existe également. Ces méthodes se concentrent à fournir des recommandations à un

groupe d'utilisateurs, en essayant ainsi de maximiser la satisfaction générale du groupe (Coyle *et al.*, 2006; Chen *et al.*, 2008).

Nous nous baserons principalement dans le reste de ce mémoire sur la classification de (Celma, 2008) et nous ferons référence à celles de (Burke, 2002) et (Adomavicius & Tuzhilin, 2005) pour une meilleure clarté.

2.3.1 Filtrage collaboratif

Dans le *filtrage collaboratif*, l'entrée du système est un ensemble de votes d'utilisateurs sur les articles. Les utilisateurs peuvent être comparés en fonction de leur évaluation partagée sur les articles, ce qui introduit la notion de voisinage des utilisateurs. De même, les articles peuvent être comparés en fonction de l'appréciation partagée des utilisateurs, introduisant la notion de voisinage des articles.

Le premier système qui a utilisé la méthode de filtrage collaboratif a été le projet Tapestry au Xerox PARC (Goldberg *et al.*, 1992). Le projet a inventé le terme de filtrage collaboratif. D'autres systèmes pionniers sont : le recommandeur de musique nommé Ringo (Shardanand, 1994; Maes et Shardanand, 1995), et Group Lens⁴, un système pour évaluer les articles USENET⁵ (Resnick *et al.*, 1994).

L'estimation d'articles pour un utilisateur donné peut alors être prédite en se basant sur les estimations données dans son voisinage d'utilisateurs et son voisinage d'articles. D'après (Breese, Heckerman, & Kadie, 1998), sur le plan algorithmique, il y a deux classes de filtrage collaboratif : les algorithmes basés sur la mémoire (*memory-based*) dits aussi basés sur les heuristiques (*heuristic-based*) et ceux basés sur les modèles (*model-based*). Pour Candillier *et al.*, (2009) les méthodes basées sur la mémoire sont de deux types : basés sur les utilisateurs et basés sur les articles. Ces approches sont formalisées et comparées dans cette section.

⁴ www.grouplens.org/

⁵ Usenet est un système en réseau de forums, inventé en 1979 et fondé sur le protocole NNTP. Il a rapidement été rendu utilisable via Internet où il reste en usage au début du XXI^e siècle.

Filtrage collaboratif à base de mémoire ou heuristique (basé sur usager)

Le filtrage collaboratif à *base de mémoire* ou *heuristique* considère la totalité des évaluations par les usagers disponibles au moment du calcul de la recommandation.

Pour les approches à *base d'utilisateur* (Resnick *et al.*, 1994; Maes & Shardanand, 1995), la prédiction d'un utilisateur évaluant un article est fondée sur les estimations, sur cet article, des plus proches voisins. Donc une mesure de similarité entre les utilisateurs doit être définie avant qu'un ensemble de voisins les plus proches ne soit choisi. De plus, une méthode pour combiner les estimations de ces voisins sur l'article prévu doit être choisie.

- Phase du calcul du voisinage

En se basant sur le profil de cet usager u_i , le système recherche les usagers u_j (j diffère de i) qui lui sont les plus similaires. Les deux mesures de similarité qui sont très utilisées sont : *la similarité vectorielle* et *la corrélation de Pearson* (Breese *et al.*, 1998)

- *La similarité vectorielle*

Dans cette méthode les usagers A et B sont considérés comme deux vecteurs de même origine dans un espace de m dimensions, m est égal au nombre d'items évalués par les deux usagers. Plus deux usagers sont similaires, plus l'angle entre leurs vecteurs respectifs est petit. Empiriquement, la similarité entre ces deux usagers est calculée par la formule du *Cosinus* suivante :

$$sim(A, B) = \frac{\sum_{j=1}^n v_{A,j}}{\sqrt{\sum_{j=1}^n v_{A,j}^2}} \times \frac{v_{B,j}}{\sqrt{\sum_{j=1}^n v_{B,j}^2}}$$

n : nombre d'items communs entre A et B votés par v

$v_{A,j}$: vote de A pour l'item j

$v_{B,j}$: vote de B pour l'item j

Formule 2.1 : Le calcul du cosinus, réadapté de (Naak, 2009)

- *La corrélation de Pearson*

La corrélation de Pearson est une méthode issue des statistiques. Elle est aussi très utilisée dans le domaine des systèmes de recommandation pour mesurer la similarité entre deux usagers (Resnick, Iacovou, Suchak, Bergstrom, & Riedl, 1994; Maes & Shardanand, 1995). La formule suivante, nous donne cette valeur pour deux usagers A et B :

$$sim(A, B) = \frac{\sum_j (v_{A,j} - \bar{v}_A)(v_{B,j} - \bar{v}_B)}{\sqrt{\sum_j (v_{A,j} - \bar{v}_A)^2 (v_{B,j} - \bar{v}_B)^2}}$$

j : nombre d'objets ayant été voté à la fois par A et B

$v_{A,j}$: vote de A pour l'item j

\bar{v}_A : moyenne des votes de A

Formule 2.2 : La corrélation de Pearson, réadaptée de (Naak, 2009)

- Phase de prédiction

Une fois que toutes les similarités de l'utilisateur cible A par rapport aux autres usagers sont calculées et que les n usagers les plus similaires qui constituent le voisinage de cet usager cible sont définis, la prédiction de la valeur d'un item j évaluée par l'utilisateur A (P_{Aj}) est calculée à l'aide de la somme pondérée des estimations des voisins les plus proches qui ont déjà estimé l'item j :

$$P_{Aj} = \frac{\sum_{i=1}^n sim(A, i) * v_{i,j}}{\sum_{i=1}^n sim(A, i)}$$

n : nombre d'utilisateurs présents dans le voisinage de A, ayant déjà voté sur l'item j

$v_{i,j}$: vote de l'utilisateur i pour l'objet j

Formule 2.3 : La formule de calcul de la prédiction, adaptée de (Candillier *et al.*, 2009)

Pour tenir compte de la différence dans l'utilisation de l'échelle d'estimation par de différents utilisateurs, les prédictions basées sur les déviations des estimations moyennes ont été proposées. P_{aj} peut être calculé à l'aide de la formule suivante :

$$P_{Aj} = \bar{v}_A + \frac{\sum_{i=1}^n sim(A, i) * (v_{i,j} - \bar{v}_i)}{\sum_{i=1}^n |sim(A, i)|}$$

n : nombre d'utilisateurs présents dans le voisinage de A , ayant déjà voté sur l'item j

$v_{i,j}$: vote de l'utilisateur i pour l'objet j

\bar{v}_i : moyenne des votes de l'utilisateur i

$|sim(A, i)|$: similarité moyenne

Formule 2.4 : La formule de calcul de la prédiction, réadaptée de (Naak, 2009)

Filtrage collaboratif à base de mémoire ou heuristique (basé sur item)

La méthode basée sur les items exploite la similarité parmi les articles. Cette méthode examine l'ensemble d'articles qu'un utilisateur a évalué, et calcule la similarité avec l'article de cible (pour décider s'il est pertinent pour être recommandé à l'utilisateur ou pas).

Étant donné une mesure de similarité entre les articles, de telles approches définissent d'abord des voisinages d'articles. L'estimation prédite pour un utilisateur sur un article est alors obtenue en utilisant les estimations de l'utilisateur sur les voisins de l'article cible. La première étape est d'obtenir la similarité entre les deux items, i et j . Les choix possibles pour calculer la similarité $sim(i, j)$ entre les articles i et j sont aussi la corrélation Pearson et la similarité vectorielle. Cependant, pour la similarité basée sur les items, la similarité vectorielle ne prend pas en considération les différences dans l'échelle

d'évaluation entre différents utilisateurs. La similarité vectorielle ajustée se sert de l'estimation moyenne d'utilisateur de chaque paire évaluée, et fait face à la limitation de la similarité vectorielle. Empiriquement, la similarité entre deux items est calculée par la formule du Cosinus ajusté suivante :

$$sim(i, j) = \sum_{A=1}^m \frac{(v_{A,i} - \bar{v}_A)}{\sqrt{\sum_{A=1}^m (v_{A,i} - \bar{v}_A)^2}} \times \frac{(v_{A,j} - \bar{v}_A)}{\sqrt{\sum_{A=1}^m (v_{A,j} - \bar{v}_A)^2}}$$

m : nombre d'utilisateurs qui ont votés pour les 2 items
 $v_{A,i}$: vote de A pour l'item i
 $v_{A,j}$: vote de A pour l'item j
 \bar{v}_A : moyenne des votes de l'utilisateur A

Formule 2.5 : Le calcul du cosinus ajusté, adapté de (Celma, 2009)

Une fois que la similarité parmi les articles a été calculée, la prochaine étape est de prévoir pour l'utilisateur cible A , une valeur pour l'article actif i . Une manière commune est de capturer comment l'utilisateur a évalué les articles similaires. La valeur prévue est basée sur la somme pondérée des estimations de l'utilisateur ainsi que les déviations des estimations moyennes et peut être calculée à l'aide de la formule suivante :

$$P_{Ai} = \bar{v}_i + \frac{\sum_{j=1}^m sim(i, j) * (v_{A,j} - \bar{v}_j)}{\sum_{j=1}^m |sim(i, j)|}$$

m : nombre d'items présents dans le voisinage de item i , ayant déjà été voté par l'utilisateur A
 $v_{A,j}$: vote de l'utilisateur A pour l'objet j
 \bar{v}_j : moyenne des votes pour l'item j
 $|sim(i, j)|$: similarité moyenne

Formule 2.6 : La formule de calcul de la prédiction, adaptée de (Celma, 2009)

De plus, Sarwar *et al.*, (2001) présente l'évidence empirique que les algorithmes à base d'articles peuvent fournir une meilleure performance de calcul que les méthodes traditionnelles basées sur l'utilisateur, en fournissant en même temps une qualité comparable ou meilleure que les meilleurs algorithmes disponibles, basés sur l'utilisateur.

Filtrage collaboratif basé sur modèle

Pour les *modèles basés sur mémoire* la complexité de calcul est trop grande pour les énormes ensembles de données et beaucoup d'applications réelles ont besoin des prédictions qui peuvent être faites très vite. Ces considérations sont les points de départ d'approches à base de modèle (Breese *et al.*, 1998). L'idée générale est d'obtenir un modèle des données hors ligne pour prédire des estimations en ligne aussi vite que possible.

Pour la construction du modèle plusieurs méthodes sont utilisées, par exemple (Breese, Heckerman, & Kadie, 1998) proposent deux méthodes probabilistes pour construire les modèles.

Les premiers types de modèles qui ont été proposés consistent à regrouper les utilisateurs en utilisant la classification et ensuite prédire le vote d'utilisateur sur un élément en utilisant uniquement les évaluations des utilisateurs qui appartiennent au même cluster.

La deuxième méthode est celle des réseaux Bayésiens. En général, les méthodes basées sur le modèle utilisent les techniques d'apprentissage automatique et les techniques statistiques pour apprendre le modèle à partir des profils des usagers. (Naak, 2009)

2.3.2 Filtrage démographique

Même s'il distingue le *Filtrage Démographique* (FD) comme une méthode à part entière, Burke, fait remarquer qu'il est un cas particulier du filtrage collaboratif. La méthode démographique est similaire à celle collaborative dans son approche (en comparant les utilisateurs), juste en utilisant de différentes données d'entrée (les caractéristiques démographiques contre les estimations) pour produire la recommandation.

Le filtrage démographique peut être utilisé pour identifier le type d'utilisateurs qui aiment un certain article (Rich, 1979). Par exemple, on pourrait s'attendre à apprendre le genre de personne qui aime un certain chanteur (par exemple trouvant l'utilisateur stéréotypé qui écoute le groupe de Bon Jovi). Cette technique classe les profils d'utilisateurs dans les clusters selon les données personnelles (âge, état civil, genre, etc.), données géographiques (ville, pays) et données psychographiques (intérêts, style de vie, attitudes, croyances et personnalités). (Celma, 2009)

Un premier exemple d'un système de filtrage démographique est le système de Grundy (Rich, 1979). Grundy recommande des livres en se basant sur les renseignements personnels recueillis à partir d'un dialogue interactif.

Le problème principal de cette méthode est que le système recommande les mêmes articles aux gens avec des profils démographiques semblables, donc les recommandations sont trop générales (ou, au moins, pas très spécifiques pour un utilisateur donné). L'avantage principal d'une approche démographique est qu'il peut ne pas exiger un historique d'estimations d'utilisateur, comment les techniques en collaboration et à base de contenu ont besoin. En fait, c'est la méthode de recommandation la plus simple.

2.3.3 Filtrage à base de contenu

Dans l'approche de *Filtrage à Base de Contenu* (FBC), le recommandeur collecte des informations décrivant les articles et puis, basé sur les préférences d'utilisateur, il prévoit quels articles l'utilisateur pourrait aimer. Cette approche ne se fonde pas sur d'autres

estimations d'utilisateur, mais sur la description des articles (Celma, 2009). Ou comme (Naak, 2009) l'a défini : « *les méthodes basées sur le contenu, comme leur nom l'indique, se basent sur la compréhension de pourquoi l'utilisateur, à qui la recommandation est destinée, a donné une haute valeur à certains items qu'il a évalués dans le passé ? Une fois cette question résolue, le système cherche parmi les nouveaux items ceux qui maximisent ces caractéristiques pour les lui recommander* ».

Les approches FBC initiales avaient leurs racines dans le champ de Recherche Documentaire (RD). Les premiers systèmes sont concentrés sur le domaine de type textuel, comme les articles de journaux, les pages Web et ils ont appliqué des techniques de RD pour extraire les informations significatives du texte. Récemment sont apparues quelques solutions qui s'occupent des domaines plus complexes, comme la musique. Cela a été possible, partiellement, parce que la communauté multimédia a mis l'accent et a amélioré l'extraction de caractéristique ainsi que les algorithmes d'apprentissage machine.

Le principal apport de ce type de système de recommandation est la prise en considération des profils qui représentent les goûts et les préférences de l'utilisateur. Le processus de caractériser l'ensemble de données d'article peut être automatique (par exemple en extrayant des caractéristiques en analysant le contenu), basé sur les annotations manuelles des experts du domaine, ou en utilisant même les étiquettes (*tags*) de la communauté d'utilisateurs (Celma, 2009).

L'élément clé de cette approche est la fonction de similarité entre les éléments. La fonction de similarité calcule la distance entre deux articles. La similarité à base de contenu se concentre sur une distance objective parmi les articles, sans introduire n'importe quel facteur subjectif dans la métrique (comme fait le filtrage collaboratif). La plupart de métriques de distance s'occupe des attributs numériques, ou des vecteurs de caractéristiques simples. Quelques distances communes, étant donné deux vecteurs de caractéristique x et y , sont expliquées : la distance euclidienne, la distance Manhattan, la distance Tchebychev, la distance de cosinus pour les vecteurs et la distance Mahalanobis (Celma, 2009).

2.3.4 Filtrage contextuel

Les systèmes de recommandation discutés ci-haut opèrent dans un espace bidimensionnel (*utilisateur, article*) du fait qu'ils tiennent compte que des deux profils : usager et item.

Cependant, les mesures de similarité ne sont pas toujours objectives. Dans quelques domaines, la similarité est très dépendante du contexte. En fait, la partie subjective est un grand facteur de la mesure, et ces mesures ne prennent pas en considération ceci. Il y a plusieurs éléments dépendants du contexte qui devraient être considérés (par exemple à qui?, quand?, où ?, et particulièrement pourquoi ?). (Celma, 2009)

Une définition générale du contexte peut être : l'ensemble des circonstances entourant une situation donnée (Richard, 2010). Le filtrage contextuel utilise les informations contextuelles (circonstances) pour décrire et caractériser les articles. Pour comparer le filtrage à base de contenu et à base de contexte, un exemple clair est la détection de spam de courrier électronique. Le premier est fondé sur l'analyse de texte du courriel (c'est-à-dire à base de contenu), alors que le filtrage contextuel ne s'occupe pas du contenu du courriel. Il utilise plutôt le contexte de la connexion SMTP pour décider si un courriel électronique devrait être marqué comme spam ou non.

Celma explique deux techniques, le forage de données Web et l'étiquetage social (*Social Tagging*), qui peuvent être utilisés pour obtenir la similarité parmi les articles (ou l'utilisateur) et peut également fournir des recommandations efficaces. Le forage de données Web est fondé sur le fait d'analyser le contenu disponible sur le Web, aussi bien que l'utilisation et l'interaction avec le contenu. L'étiquetage social fait un forage de données sur les informations recueillies auprès d'une communauté d'utilisateurs qui annotent (étiquettent, tag) les articles.

Quand les utilisateurs étiquettent des articles, nous obtenons un triplet (*utilisateur, article, étiquette*). Ces triplets représentent une matrice de trois dimensions (également appelée tenseur, un vecteur multidimensionnel).

Force et faiblesses des méthodes traditionnelles :

Dans son mémoire Naak utilise les termes classique et pur, dans le contexte de la recommandation, pour faire référence aux méthodes basées sur le filtrage collaboratif et les méthodes basées sur le contenu. On dit qu'elles sont classiques du fait qu'elles sont les premières utilisées et qu'elles sont largement admises. On dit qu'elles sont pures par comparaison aux hybrides, qui justement sont des combinaisons de méthodes classiques. Nous on peut également ajouter dans la liste de méthodes classiques les méthodes basées sur le filtrage démographique et les méthodes basées sur le Filtrage ConteXtuel (FCX).

Table 2.1 Forces et faiblesse des méthodes traditionnelles, adaptées de (Burke, 2002)

Technique	Forces	Faiblesses
FC	<i>Cross-genre niches</i> ... (a) Connaissance du domaine non requise ... (b) Adaptabilité : la qualité croit avec le temps ... (c) Feed-back implicite suffisant ... (d)	Problème du nouvel usager ... (f) Problème du nouvel item ... (g) Démarrage à froid ... (h) La <i>sparsity</i> ... (i) Le <i>gray sheep</i> ... (j) Le <i>shilling</i> ... (k)
FD	(a), (b), (c)	(f), (i), (j), (k) Recueillir des informations démographiques ... (l)
FBC	(b), (c), (d)	(f), (j) La surspécialisation ... (m) Limite liée au contenu d'un item ... (n)
FCX	Sensible avec le changement des préférences ... (e), pas de (f) Comprendre les utilisateurs (p)	(i), (j) Polysémie et synonymie ... (o)

La table 2.1 résume les forces et faiblesses des méthodes traditionnelles, en l'occurrence le Filtrage Collaboratif (FC), le Filtrage Démographique (FD), le Filtrage à Base de Contenu (FBC), et le Filtrage ConteXtuel (FCX) :

(a) *Crosse-genre niches*

Le filtrage collaboratif se distingue par sa capacité unique de recommander à un usager ce qui est hors du familier, c'est ce que Burke appelle : *cross-genre niches*. En effet, un usager peut apprécier la musique jazz, mais il peut aussi apprécier la musique classique, mais un système de recommandation basé sur le contenu et entraîné sur les préférences de jazz ne serait pas en mesure de suggérer des chansons classiques puisqu'aucune des caractéristiques (artistes, instruments, répertoire) associés aux articles dans les différentes catégories ne serait partagée.

(b) *Connaissance du domaine*

La connaissance du domaine n'est pas requise, le processus de recommandation se base uniquement sur les évaluations des items.

(c) *Adaptabilité*

Au fur et à mesure que la base de données des évaluations augmente, la recommandation devient plus précise.

(d) *Feed-back implicite suffisant*

En effet, un système de recommandation peut utiliser les habitudes d'écoute, les achats sans avoir besoin d'une évaluation explicite.

(e) *Sensible avec le changement des préférences*

Un fan de jazz qui devient un fan de musique classique continuera à obtenir des recommandations de musique jazz pour un certain temps, jusqu'à ce que les plus récentes évaluations ont la chance d'incliner la balance.

(f) *Problème du nouvel usager*

Un problème commun au filtrage collaboratif et au filtrage à base de contenu est qu'un nouvel utilisateur qui n'a pas encore accumulé suffisamment d'évaluations ne peut pas

avoir de recommandations pertinentes. Le filtrage contextuel n'arrive pas d'avoir ce problème.

(g) Problème du nouvel item

C'est un problème qui concerne le filtrage collaboratif, le filtrage contextuel et non pas le filtrage à base de contenu. Comme, dans le cas du filtrage à base de contenu, il suffit d'introduire l'item dans le système (la base de données) pour que celui-ci soit analysé et rentré dans le processus de recommandation. Alors que dans le cas du filtrage collaboratif ou contextuel il doit avoir suffisamment d'évaluations pour que celui-ci soit pris en considération dans le processus de recommandation.

(h) Démarrage à froid

Le problème de démarrage à froid d'un recommandeur arrive quand un nouvel utilisateur (ou un nouvel élément) entre dans le système. Le démarrage à froid est un problème du filtrage collaboratif. D'une part, le démarrage à froid est un problème pour les nouveaux usagers qui commencent à jouer avec le système, parce que le système ne dispose pas d'assez d'informations à leur sujet. Si le profil d'utilisateur est vide, il doit consacrer une somme d'efforts à l'aide du système avant d'obtenir une récompense (les recommandations utiles). D'autre part, quand un nouvel élément est ajouté à la collection, le système doit avoir suffisamment d'informations pour être en mesure de recommander cet article aux utilisateurs.

(i) La Sparsity

La sparsity est une propriété inhérente de l'ensemble des données. Avec un nombre relativement important d'utilisateurs et d'articles, le principal problème est la faible couverture de l'interaction des usagers avec les items. Un facteur connexe est la dimensionnalité de l'ensemble de données, qui se compose de nombreux usagers et items.

(j) Le gray Sheep (Claypool et al., 1999)

Un autre problème, lié avec le précédent, est que les utilisateurs ayant des goûts atypiques (qui varient de la norme) n'auront pas beaucoup d'utilisateurs en tant que voisins. Ainsi, cela mènera à des recommandations pauvres. Ce problème est également connu comme

« *gray sheep* ». La popularité est un problème qui se produit fréquemment dans les FC. Il est analogue au paradigme « riche s'enrichit », en d'autres termes, plus un item est bien évalué, plus il est recommandé.

(k) Le shilling

C'est l'action malveillante d'influencer la recommandation en créant de faux profils pour voter et favoriser/défavoriser certains items.

(l) Recueillir des informations démographiques

Un désavantage est la génération du profil, qui a besoin d'un peu d'effort de la part de l'utilisateur. Quelques approches essaient d'obtenir l'information (non structurée) provenant des pages Web utilisateur, weblogs, etc. Dans ce cas-là, les techniques de classification de texte sont utilisées pour créer les groupes et classifier les utilisateurs.

(m) La surspécialisation

Un autre inconvénient potentiel est le problème de nouveauté. En supposant que la fonction de similarité fonctionne correctement, alors on pourrait supposer qu'un utilisateur recevra toujours des articles trop semblables à ceux dans son profil. Pour faire face à cette lacune, le recommandeur doit juger qu'un item, semblable à ce que l'utilisateur a déjà vu, puisse apporter suffisamment de nouveauté avant de lui proposer sans tomber dans la redondance.

(n) Limite liée au contenu d'un item

Selon la complexité du domaine, un autre inconvénient est la limitation des fonctionnalités qui peuvent être (automatiquement) extraits à partir des objets. Par exemple, dans le domaine du multimédia, il est encore difficile d'extraire les descripteurs de haut niveau avec une signification claire pour l'utilisateur. L'analyse musicale n'est pas encore prête à prédire avec précision l'ambiance, mais, d'autre part, on a de bons résultats lorsqu'il s'agit de descripteurs tels que : l'harmonie et le rythme.

(o) Polysémie et synonymie

Une des principales limites de l'étiquetage social, c'est que, sans être contraint à un vocabulaire contrôlé, les étiquettes ont les problèmes suivants : la polysémie et la synonymie

2.3.5 Les méthodes hybrides

Les faiblesses des méthodes traditionnelles ont fait l'objet de plusieurs études dans la littérature scientifique concernant les systèmes de recommandation. De nombreuses solutions sont proposées. Elles sont, généralement, des méthodes basées sur la combinaison de méthodes traditionnelles présentées auparavant. Ce paragraphe présente un survol de ces méthodes appelées hybrides.

Selon Adomavicius et Tuzhilin on peut distinguer quatre façons de combiner les méthodes traditionnelles :

- Implémenter la méthode collaborative et la méthode basée sur le contenu séparément puis combiner leurs prédictions.
- Incorporer quelques caractéristiques de la méthode basée sur le contenu dans l'approche collaborative.
- Incorporer quelques caractéristiques de la méthode collaborative dans l'approche à base de contenu.
- Construction d'un modèle général unifié qui incorpore les caractéristiques des deux modèles. (Naak, 2009).

La meilleure description des méthodes hybrides a été faite par Burke, en 2002 (voir Table 2.2). Alors, selon Burke on peut distinguer sept façons de combiner les méthodes traditionnelles :

(a) Pondération (Weighted)

Une méthode hybride qui combine la sortie d'approches distinctes, utilisant, par exemple, une combinaison linéaire des scores de chaque technique de recommandation.

(b) Commutation (Switching)

C'est une technique qui permet de faire le choix d'un modèle de recommandation parmi plusieurs, en se basant sur plusieurs critères. La détermination de la technique appropriée dépend de la situation. Le système se doit alors de définir les critères de commutation, ou les cas où l'utilisation d'une autre technique est recommandée. Ceci permet au système de

connaître les points forts et les points faibles des techniques de recommandation qui le constituent.

(c) Technique mixte (Mixed)

Dans cette approche, le recommandeur ne combine pas, mais augmente la description des ensembles de données, en prenant en considération les estimations des utilisateurs et la description des articles. La nouvelle fonction de prédiction doit faire face aux deux types de descriptions et permet d'éviter les problèmes posés par le filtrage collaboratif, à savoir, le démarrage à froid.

(d) Combinaison de caractéristiques (Features combination)

Dans un hybride basé sur la combinaison de caractéristiques, les données provenant de techniques collaboratives sont traitées comme une caractéristique, et une approche basée sur le contenu est utilisée sur ces données.

(e) Cascade

La cascade implique un processus étape par étape. Dans ce cas, une technique de recommandation est appliquée en premier, produisant un ensemble de candidats potentiels. Puis, une deuxième technique raffine les résultats obtenus dans la première étape. Cette méthode a pour avantage que si la première technique génère peu de recommandations, ou si ces recommandations sont ordonnées afin de permettre une sélection rapide, la deuxième technique ne sera plus utilisée.

(f) Augmentation de caractéristiques (Feature augmentation)

L'augmentation de caractéristiques est semblable à la cascade, mais dans ce cas-là les résultats obtenus (le classement ou la classification) de la première technique sont utilisés par le deuxième comme une caractéristique ajoutée.

(g) Méta niveau (Meta-level)

Dans un hybride basé sur méta niveau, une première technique est utilisée, mais différemment que la précédente méthode (augmentation de caractéristiques), non pas pour produire de nouvelles caractéristiques, mais pour produire un modèle. Et dans la deuxième étape, c'est le modèle entier qui servira d'entrée pour la deuxième technique.

Table 2.2 Les méthodes hybrides, adaptées de (Burke, 2002)

Méthode	Description
Pondération (a)	Les résultats (ou votes) des différents techniques de recommandation sont combinés pour produire une seule recommandation
Commutation (b)	Le système commute entre les techniques de recommandation selon la situation actuelle
Technique mixte (c)	Les recommandations de différents recommandeurs sont présentées en même temps
Combinaison des caractéristiques (d)	Les données provenant d'une technique sont traitées comme une caractéristique, et une approche basée sur une autre technique est utilisée sur ces données
Cascade (e)	Un recommandeur raffine les recommandations données par un autre
Augmentation des caractéristiques (f)	La sortie d'une technique est utilisée comme une caractéristique d'entrée à l'autre
Méta niveau (g)	Le modèle appris d'un recommandeur est employé comme entrée à l'autre

2.4 Conclusion

Ce chapitre a présenté le problème de la recommandation. Les composants principaux d'un recommandeur sont les usagers et les articles. Basé sur les préférences d'utilisateur et l'exploitation du profil d'utilisateur, un système de recommandation peut proposer des articles aux utilisateurs. Après avoir présenté les trois principales approches dans la classification des systèmes de recommandation, nous avons pris position en adoptant la classification de Celma, 2008. Sans pour autant négliger les autres approches (Burke, 2002 et Adomavicius, 2005), nous ferons appel à leurs terminologies pour une meilleure clarté.

Pour conclure, la table 2.3 présente les principaux éléments impliqués dans le problème de recommandation : le profil d'utilisateur et les méthodes de recommandation. Ensuite, le chapitre 3 explique toutes ces notions dans le domaine de recommandation de musique.

Table 2.3 Résumé des éléments impliqués dans le problème de recommandation

Le profil d'utilisateur	
Génération	<ul style="list-style-type: none"> • <i>profil vide</i> • <i>manuellement</i> • <i>import</i> • <i>ensemble d'apprentissage</i> • <i>stéréotype</i>
Maintenance	<ul style="list-style-type: none"> • <i>feedback explicite</i> • <i>feedback implicite</i>
La recommandation	
Les méthodes de filtrage	<ul style="list-style-type: none"> • <i>le filtrage collaboratif</i> • <i>le filtrage démographique</i> • <i>le filtrage à base de contenu</i> • <i>le filtrage contextuel</i> • <i>les méthodes hybrides</i>

Chapitre 3 La recommandation de musique

Ce chapitre approfondit l'objet central de notre recherche, qui est la musique avec ses méta-informations, dans l'optique de bien comprendre ses spécificités et ainsi de mieux évaluer les besoins du domaine. Plus précisément, nous passons en revue quatre classes de systèmes de recommandations de musique : les systèmes basés sur le filtrage collaboratif, les systèmes basés sur le filtrage à base de contenu, les systèmes basés sur le filtrage contextuel et finalement les systèmes hybrides.

3.1 Introduction

Les nouvelles technologies créent de nouvelles opportunités et des marchés entiers se développent spontanément. Avec l'introduction d'iTunes⁶, une application qui garde la trace de sa musique numérique, le suivi de la collection numérique d'un usager est devenu plus facile. Immédiatement, les gens sont devenus capables d'acheter la musique numériquement pour \$0.99 une chanson – de télécharger cette musique directement sur leur ordinateur de façon légale, en la conservant sur leurs appareils portatifs et en gravant leur musique sur les CDs.

Mais pendant que le Magasin de Musique iTunes⁷ rend l'acquisition de la musique plus facile que jamais, les amateurs de musique trouvent toujours difficile de découvrir la nouvelle musique. Traditionnellement, la seule façon pour les gens d'écouter la nouvelle musique est à la radio ou en l'entendant chez des amis.

La popularité de la station de radio en ligne *Pandora.com*⁸ (20 millions d'utilisateurs), du site Internet proposant un système de collection de statistiques et de

⁶ iTunes est un logiciel propriétaire de lecture et de gestion de bibliothèque multimédia numérique distribué gratuitement

⁷ L'iTunes Store, d'abord appelé l'iTunes Music Store (iTMS), est un service d'achat de musique et autres contenus en ligne proposé par Apple depuis le 28 avril 2003 aux États-Unis et depuis le 2 décembre 2004 au Canada

⁸ Pandora est un site web d'aide à la découverte de nouvelles musiques proches de ses goûts créé par le Projet du Génome Musical

recommandation de musique *Last.fm*⁹ (le site se targuait d'avoir plus de 40 millions d'utilisateurs actifs dans plus de 200 pays) et de *Genius*¹⁰ *d'iTunes* (apparu en septembre 2008 et disponible pour plus de 10 millions des utilisateurs *iTunes*) a introduit le sujet de recherche de *MIR*¹¹ (la similarité musicale et les systèmes de recommandations) dans le projecteur public.

Barrington *et al.*, (2009) démontrent que le filtrage collaboratif de *Genius* permet de produire de meilleures recommandations de musique que les systèmes basés sur les métadonnées simples (*Last.fm*) - ou l'analyse à base de contenu (*Pandora*). Dans ce chapitre, nous cherchons à comprendre les repères musicaux que *Genius* et les autres systèmes identifie pour capturer la similarité musicale. Nous constatons que ces solutions ne sont pas bien adaptées pour les besoins spécifiques de ce qui est l'objet de notre recherche. Nous pouvons alors élaborer des systèmes de recommandation qui utilisent les mêmes repères, en apportant nos améliorations.

3.2 Les cas d'utilisation

L'objectif principal d'un Système de Recommandation de Musique (SRM) est de proposer, à l'utilisateur, des artistes intéressants et inconnus (et leurs chansons disponibles si possible), basés sur son goût musical (préférences).

La musique se distingue d'autres domaines de divertissement, comme les films, ou les livres. Une différence évidente est qu'une personne peut écouter plusieurs fois la même chanson. Une autre différence est que le suivi des préférences de l'utilisateur est fait de manière implicite, par ses habitudes d'écoute. Le feedback explicite n'est pas recueilli en

⁹ Last.fm est à la fois une station de radio diffusée sur Internet et un site Internet proposant un système de collection de statistiques et de recommandation de musique.

¹⁰ Genius est une fonctionnalité apparue avec iTunes 8, il génère automatiquement des listes de lectures à partir de chansons similaires à la piste sélectionnée dans la bibliothèque.

¹¹ Music information retrieval (MIR).

termes de votes comme pour les films et les livres, et surtout basé sur les actions de jouer, sauter, ou arrêter une chanson recommandée.

La plupart du travail fait dans la recommandation de musique se concentre à présenter à un utilisateur une liste d'artistes, ou de créer une séquence ordonnée de chansons (une liste de lecture personnalisée). Actuellement, la plupart des systèmes de recommandation de musique sont basés sur le filtrage soit collaboratif (social) soit à base de contenu. Récemment, sont apparues d'autres approches contextuelles telles que l'étiquetage social. L'exploration du Web de musique peut être aussi utilisée à cette fin.

3.2.1 La recommandation des artistes

Selon le modèle général présenté dans le chapitre 2, la recommandation des artistes suit la mise en correspondance utilisateur-item (artiste), ou les articles sont recommandés à l'utilisateur en fonction de son profil.

Cependant, la recommandation d'artistes devrait impliquer une plus large expérience avec l'utilisateur, plus que la présentation d'une liste d'artistes. Dans ce sens, il y a beaucoup d'informations relatives à la musique sur Internet : la musique jouée par les artistes « *inconnus* » pour l'utilisateur (peut s'adapter parfaitement comme une nouvelle recommandation), la nouvelle musique d'artistes « connus », les annonces des concerts, les critiques d'albums, les mp3-blogs, les sessions de podcast, etc.

3.2.2 La recommandation des voisins

Le but de la recommandation des voisins est de trouver les personnes semblables, et par eux de découvrir de la musique inconnue et intéressante. La similarité de voisinage peut être calculée en utilisant les pairs des profils usager-usager. Un des principaux avantages de créer des voisinages est qu'un utilisateur peut explorer ses utilisateurs semblables, facilitant ainsi le processus de découverte de la musique. En outre, il permet la création des réseaux sociaux, reliant les personnes qui partagent les mêmes intérêts.

3.2.3 La recommandation des listes de lecture

Une liste de lecture est une collection de chansons groupée ensemble sous un principe particulier. Le principe peut être général, comme « chansons rock des années 70 » ou personnel tel que « chansons qui me rappellent Mélanie ». (Barrington *et al.*, 2009). Cunningham *et al.*, (2006) font la distinction entre les listes de lecture et les « *mix* » ou « *compilations* ». Tandis qu'un mix peut avoir des thèmes abstraits et l'ordre des chansons est important, une liste de lecture incarne simplement une humeur ou un état émotif désiré ou agit en tant que fond à une activité (travail, romance, sports, etc.). L'ordre des chansons dans une liste de lecture n'est pas important et il est souvent joué de façon aléatoire.

La génération des listes de lecture est une application importante dans la recommandation de musique, parce qu'elle permet aux utilisateurs d'écouter la musique aussi bien que de fournir le feed-back immédiat, donc le système peut réagir en conséquence. Il y a deux principaux modes de génération de liste de lecture : en utilisant la *collection personnelle* de l'utilisateur (qui est typique au jeu aléatoire), et en utilisant la musique qui se trouve en *dehors de la collection personnelle*, où le jeu aléatoire n'est pas très utile, mais où une liste de lecture personnalisée a plus de sens.

Les listes de lecture peuvent être générées (soit automatiquement, soit à la main) pour refléter un état d'esprit, pour accompagner une activité ou pour explorer de nouvelles chansons pour la découverte de la musique. Les recommandations peuvent être fondées sur la similarité avec une ou plusieurs chansons.

Puisque la musique est souvent expérimentée dans un contexte social, les facteurs comme la popularité de chanson, la familiarité et la perception du système de recommandation comme un expert humain peuvent jouer un grand rôle dans la qualité perçue de la liste de lecture (Dodds *et al.*, 2006). Même les systèmes qui génèrent des listes de lecture originales ou des listes de lecture avec un heureux hasard pour la découverte de chansons doivent comporter certains éléments familiers et pertinents pour inspirer les utilisateurs à faire confiance au recommandeur. Ceci peut être réalisé en offrant une certaine transparence aux recommandations, par exemple, en montrant les artistes similaires ou en utilisant les tags (Celma *et al.*, 2008).

3.3 Le profil d'utilisateur

La musique est un véhicule important pour relater à d'autres personnes notre personnalité, histoire, etc. Le goût musical et les préférences musicales sont affectés par plusieurs facteurs, en incluant les traits démographiques et de personnalité. Il semble raisonnable de penser qu'en combinant les préférences musicales avec les aspects personnels - telles que : l'âge, le genre, l'origine, le métier, l'éducation musicale, etc. – nous pouvons améliorer la recommandation de musique (Uitdenbogerd *et al.*, 2002).

3.3.1 Les types d'auditeurs

Jennings (2007) résume les quatre degrés d'intérêt pour la musique, ou le type d'auditeurs, identifiés dans le « *UK 2006 Projet Phoenix 2* ». L'étude est fondée sur l'analyse de différents types d'auditeurs, avec une tranche d'âge allant de 16 à 45. La classification inclut :

- *les savants* : pour eux tout dans la vie semble être attaché à la musique. Leur connaissance musicale est très étendue. Comme prévu, ils représentent seulement 7 % de la catégorie d'âge 16-45. Ils sont les auditeurs les plus difficiles à qui fournir des recommandations, parce qu'ils sont très exigeants. Les recommandations ne doivent pas être populaires, mais surtout intelligentes;
- *les passionnés* : représentant 21 % du groupe 16-45 ans, pour eux la musique est un élément clé de la vie, mais est également équilibrée par d'autres intérêts. Ils apprécient une balance entre les recommandations intéressantes, inconnues et familières ;
- *les ordinaires* : la musique joue un rôle occasionnel, mais d'autres choses sont bien plus importantes. Elles représentent 32 % de la catégorie recherchée;

- *les indifférents* : pour eux ce n'est pas grave si la musique cesse d'exister. Avec une proportion de 40 % du groupe 16-45 ans, ils représentent le type prédominant d'auditeurs.

3.3.2 Les types de représentation proposées

Comme nous avons vu dans la section précédente, la recommandation de musique est très dépendante du type d'utilisateur. De plus, la musique est un facteur important pour relater à d'autres personnes notre personnalité, histoire, etc. La modélisation de l'utilisateur est alors une étape cruciale dans la compréhension des préférences des utilisateurs.

Le fait de construire des profils exacts est une tâche clé puisque le succès du système dépendra dans une large mesure, de la capacité de représenter les intérêts actuels de l'utilisateur.

Un modèle utilisateur est nécessaire pour préciser certaines caractéristiques perceptuelles. Les caractéristiques musicales peuvent être classifiées comme *quantitatives* et *qualitatives* (Chai *et al.*, 2000). Les caractéristiques *quantitatives* (certains d'entre eux sont perceptuels) sont faciles de décrire parce qu'ils dépendent presque entièrement de la musique elle-même et ne diffèrent pas beaucoup d'une personne à une autre. Ainsi, ces caractéristiques peuvent être décrites comme un triple *<musique, caractéristique, valeur>*. Les caractéristiques *quantitatives* sont les informations générales (compositeur, artiste, âge, genre, etc.), la tonalité, le tempo, les instruments, la structure musicale, la hauteur, l'intensité, la mélodie, le rythme et tous les autres paramètres qui pourraient être dérivés de la musique.

Les caractéristiques *qualitatives* (toutes sont perpétuelles) sont plus difficiles à décrire parce qu'elles dépendent des utilisateurs dans une large mesure : les votes, les fonctionnalités, les caractéristiques émotionnelles. Si un utilisateur dit au moteur de recherche qu'il veut une musique "heureuse" ou "bonne" musique, c'est dur pour le moteur de recherche de choisir de façon exacte, sans avoir des informations supplémentaires sur cet utilisateur. Ainsi, ces caractéristiques devraient au moins être décrites comme un *4-tuple* : *<musique, utilisateur, caractéristique, valeur>*. Encore plus compliqué, la musique peut

sembler différente à la même personne dans différents contextes, par exemple, l'utilisateur veut une « musique convenable pour une soirée romantique », dans ce cas, ces caractéristiques devraient être décrites comme un *5-tuple* : *<musique, utilisateur, contexte, caractéristique, valeur>*.

Toutefois, dans le domaine de recommandation musicale, il n'y a eu que quelques tentatives pour augmenter explicitement les profils d'utilisateurs en ajoutant des informations liées à la musique.

Les propositions de représentation de profil d'utilisateur (concernant la musique) les plus pertinentes sont : la norme MPEG-7 qui décrit les préférences de l'utilisateur, le langage de modélisation de l'utilisateur pour la recherche documentaire (*User Modelling for Information Retrieval Language* : UMIRL), et l'initiative l'Ami d'un Ami (*Friend Of A Friend* : FOAF) (accueilli par la communauté de Web Sémantique).

MPEG-7 est une norme de description dont le but est de faciliter l'indexation et la recherche de documents multimédia (URL, 7), connue sur le nom « *Multimedia Content Description Interface* ».

MPEG-7 c'est un standard ISO/IEC développé par MPEG¹² (URL, 6). Le format MPEG-7 comprend trois éléments principaux :

- Un ensemble de descripteurs permettant la description des contenus multimédia :
 - Descripteurs visuels, encodant des informations de couleur, texture, forme ou mouvement.
 - Descripteurs audio, encodant le timbre, le spectre, ou des éléments de plus haut niveau comme la mélodie ou la parole.
- Un langage de description des contenus multimédia : DDL (*Description Definition Language*). DDL est un dérivé du format de balisage XML Schema¹³.

¹² Moving Picture Experts Group (MPEG) est le comité qui a aussi développé les normes réussies, connues comme MPEG-1 (1992) et MPEG-2 (1994) et la norme MPEG-4 (la Version 1 en 1998 et la version 2 en 1999).

- Des éléments, appelés DS (*Description Schemes*), définissant la sémantique et les relations entre descripteurs et entre DS :
 - Les segments permettent de définir la décomposition spatio-temporelle des médias vidéo (image, plan, séquences, etc.)
 - Les informations auteurs : ce qui permet de fournir des informations « bibliographiques » sur les médias (titre, auteur, description libre, etc.)
 - Les informations physiques (principe de codage, taille, résolution, etc.).

L'exemple suivant (voir Figure 3.1) montre une définition d'un profil d'utilisateur hypothétique, indiquant qu'elle aime l'album « *Pour vous apporter mon amour* » de PJ Harvey :

```

<UserPreferences>
  <UserIdentifier protected="true">
    <Name xml:lang="ca">Joan Blanc</Name>
  </UserIdentifier>
  <FilteringAndSearchPreferences>
    <CreationPreferences>
      <Title preferencValue="8">To bring you my love</Title>
      <Creator>
        <Role>
          <Name>Singer</Name>
        </Role>
        <Agent xsi:type="PersonType">
          <Name>
            <GivenName>Polly Jean</GivenName>
            <FamilyName>Harvey</FamilyName>
          </Name>
        </Agent>
      </Creator>
      <Keyword>dramatic</Keyword>
      <Keyword>fiery</Keyword>
      <DatePeriod>
        <TimePoint>1995-01-01</TimePoint>
        <Duration>P1825D</Duration>
      </DatePeriod>
    </CreationPreferences>
  </FilteringAndSearchPreferences>
</UserPreferences>

```

Figure 3.1 Exemple d'un profil d'utilisateur MPEG 7 (Lamere et Celma, 2007)

¹³ XML Schema est un langage de description de format de document XML permettant de définir la structure et le type de contenu d'un document XML

Chai *et al.*, (2000) proposent un langage de type XML, UMIRL (« *User Modeling for Information Retrieval Language* »), dans lequel de différents systèmes peuvent décrire l'utilisateur dans ce format standard pour rendre le modèle d'utilisateur partageable et réutilisable (voir Figure 3.2). UMIRL représente une version simplifiée de MPEG 7, conçue pour la recherche d'informations musicales :

```

<user>
  <generalbackground>
    <name>John White </name>
    <education>MS</education>
    <citizenship>US</citizenship>
    <birthdate>9/7/1974</birthdate>
    <sex>male</sex>
    <occupation>student</occupation>
  </generalbackground>
  <musicbackground>
    <education>none</education>
    <instrument>piano</instrument>
    <instrument>vocal</instrument>
  </musicbackground>
  <generalpreferences>
    <color>blue</color>
    <animal>dog</animal>
  </generalpreferences>
  <musicpreferences>
    <genre>classical</genre>
    <genre>blues</genre>
    <genre>rock/pop</genre>
    <composer>Wolfgang Amadeus Mozart
  </composer>
  <artist>Beatles</artist>
  <sample>
    <title>Yesterday</title>
    <artist>Beatles</artist>
  </sample>
  </musicpreferences>
  <habit>
    <context>I'm happy
    <tempo>very fast</tempo>
    <genre>pop</genre>
  </context>
  <pfeature>romantic
  <tempo>very slow</tempo>
  <softness>very soft</softness>
  <title>*love*</title>
  </pfeature>
  <context>bedtime
  <pfeature>romantic</pfeature>
  </context>
</habit>
</user>

```

Figure 3.2 Exemple d'un profil d'utilisateur UMIRL (Chai *et al.*, 2000)

Le projet FOAF est un effort communautaire pour définir un vocabulaire RDF pour exprimer des métadonnées sur les gens, et leurs intérêts, relations et activités. Fondée par Dan Brickley et Libby Miller, FOAF est une initiative communautaire ouverte, qui aborde directement le but le plus large du Web sémantique : définir un ensemble de technologies visant à rendre le contenu des ressources du World Wide Web accessible et utilisable par les programmes et agents logiciels, grâce à un système de métadonnées formelles, utilisant notamment la famille de langages développés par le W3C¹⁴ (URL, 9).

- Friend of a Friend (FOAF)

- ❖ Linking Open Data

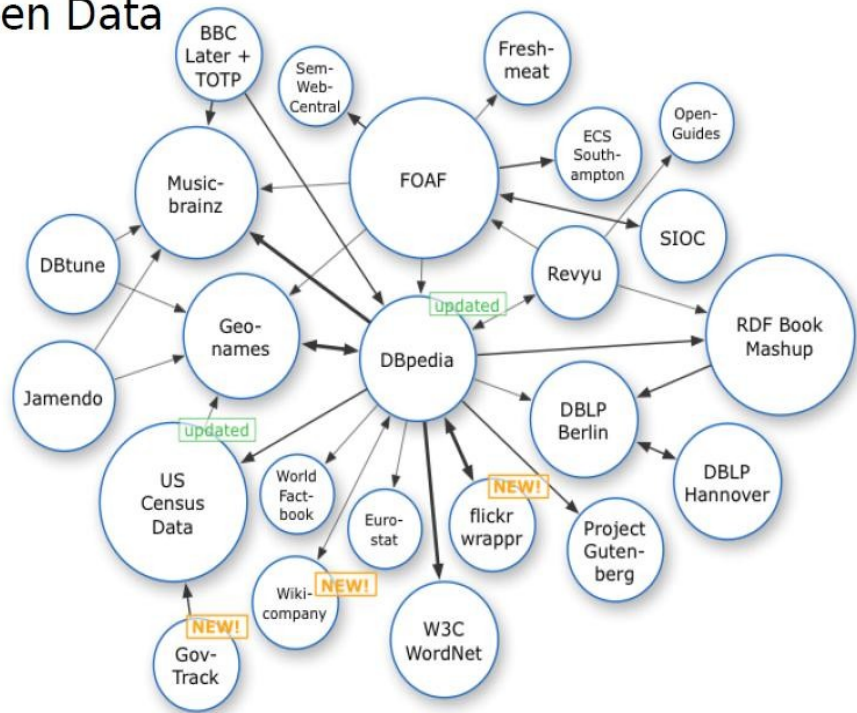


Figure 3.3 FOAF exemple (Bauer *et al.*, 2012)

¹⁴ Le **World Wide Web Consortium**, abrégé par le sigle **W3C**, est un organisme de standardisation à but non-lucratif, fondé en octobre 1994 comme un consortium chargé de promouvoir la compatibilité des technologies du World Wide Web telles que HTML, XHTML, XML, RDF, CSS, PNG, SVG et SOAP.

L'idée d'inter-corrélation entre les informations disponibles dans plusieurs bases de données ouvertes sur le web, est celle utilisée dans notre système, appelé *Mures*, présenté dans le chapitre 4.

Les profils FOAF comprennent des informations démographiques (nom, sexe, âge, pseudonyme, page d'accueil, les comptes web, etc.), géographiques (ville et pays, latitude et longitude géographiques), les informations sociales (relation avec d'autres personnes), psychographiques (i.e. les intérêts d'utilisateur) et comportementales (habitudes d'utilisation). Il y a quelques approches qui permettent la modélisation des goûts musicaux dans un profil FOAF.

L'image suivante (voir Figure 3.4) montre comment exprimer le fait qu'un utilisateur aime un artiste, en utilisant l'ontologie générale de musique proposée par Giasson et Raimond en 2007.

```
<foaf:interest>
  <mo:MusicArtist rdf:about="http://zitgist.com/music/artist/
    ca37-...fc">
    <mo:discogs rdf:resource="http://www.discogs.com/artist/PJ+
      Harvey"/>
    <foaf:img rdf:resource="http://ec2.images-amazon.com/images/
      P/B00852Q....jpg"/>
    <foaf:homepage rdf:resource="http://pjharvey.net/" />
    <foaf:name>P.J. Harvey</foaf:name>
    <mo:wikipedia rdf:resource="http://en.wikipedia.org/wiki/
      PJ_Harvey"/>
  </mo:MusicArtist>
</foaf:interest>
```

Figure 3.4 FOAF exemple (Giasson *et al.*, 2007)

Comme on peut le voir, la complexité, du point de vue de la sémantique, augmente avec chaque proposition, d'UMIRL à MPEG7 à FOAF. Lorsque nous travaillons avec des profils d'utilisateurs et des informations personnelles sensibles, la vie privée est un aspect important. Eggen *et al.*, (2004) présentent quelques recherches sur l'acquisition, le stockage et l'application d'informations personnelles sensibles. Il y a une équilibre entre les

avantages d’avoir des recommandations musicales personnalisées et la perte de la vie privée.

3.4 Le profil de la musique

Nous décrivons dans cette sous-section, la représentation et la modélisation des éléments de musique. Autrement dit, les principaux éléments qui décrivent les artistes et les chansons.

Cependant, notre société fortement numérisée produit continuellement et exploite un nombre croissant de connaissances sur la musique. Ces connaissances, aussi appelées métadonnées, ont pris une importante croissance dans l’industrie de la musique et méritent un traitement spécial. Pour rendre toute cette musique facilement accessible aux auditeurs, il est important de décrire la musique de manière à ce que les machines puissent comprendre. La gestion des connaissances de musique doit répondre à ces 2 questions : 1) comment construire des descriptions expressives de musique qui sont faciles à maintenir? et 2) comment exploiter ces descriptions pour construire des systèmes de recommandations de musique efficaces, qui aident les utilisateurs à trouver la musique, dans ces grandes collections ?

La première étape vers la gestion des connaissances musicales est probablement l’identification de la musique. Pachet (2005) classe la gestion des connaissances de musique dans trois catégories : métadonnées *éditoriales*, *culturelles* et *acoustiques*. Cette classification est basée sur la nature du processus qui a conduit à l’élaboration des métadonnées.

3.4.1 Les métadonnées des éditeurs

Les métadonnées éditoriales font allusion aux métadonnées obtenues, littéralement, par l’éditeur. Pratiquement cela signifie que les informations sont fournies manuellement, par les experts autorisés. Les exemples de métadonnées éditoriales dans la musique varient des informations d’album (par ex. la chanson « *“Yellow Submarine”* » par le Beatles apparaît

sur l'Album « *Revolver* » publié dans le Royaume-Uni) aux informations administratives comme les dates d'enregistrement, les compositeurs ou artistes.

Il est courant que de différents experts ne peuvent pas se mettre d'accord dans le fait d'allouer un genre concret à une chanson ou à un artiste. Encore plus difficile est un consensus pour une taxonomie de genres musicaux.

Du point de vue de la recommandation de musique, les métadonnées éditoriales représentent le cœur des méthodes de filtrage collaboratif.

3.4.2 Les métadonnées culturelles

Les métadonnées culturelles sont définies comme les informations qui sont implicitement présentes dans les quantités énormes de données disponibles. Ces données sont généralement collectées à partir d'Internet; via les blogs, forums, programmes de radio de musique, etc. Les métadonnées culturelles ont une composante subjective évidente car elles sont basées sur l'agrégation d'opinions personnelles.

Barrington *et al.*, (2008) présentent cinq façons différentes de recueillir les annotations au niveau de l'artiste (ou chanson). Les approches sont les suivantes :

- le forage des données ;
- les étiquettes sociales,
- le marquage automatique des contenus audio,
- le déploiement de jeux d'annotation,
- la réalisation d'une enquête

Du point de vue de la recommandation de musique, les métadonnées culturelles représentent la base pour les méthodes contextuelles.

3.4.3 Les métadonnées acoustiques

La dernière catégorie d'informations sur la musique, sont les métadonnées acoustiques. L'acoustique se réfère ici au fait que cette information est obtenue par une analyse du fichier audio, sans aucune référence à une information textuelle ou écrite. Les

métadonnées acoustiques peuvent être des informations purement objectives, en se rapportant "au contenu" de la musique.

Quelques exemples typiques d'informations acoustiques musicales sont le tempo, le rythme, l'harmonie, la mélodie, le timbre et la structure.

Du point de vue de la recommandation de musique, les métadonnées acoustiques représentent la base pour les méthodes basées sur le contenu.

D'après Lamere et Celma (2007), les métadonnées qui décrivent la musique sont de deux types : métadonnées *textuelles* (les métadonnées éditoriales et culturelles de Pachet) et métadonnées *audio* (acoustiques).

3.5 Les méthodes de recommandation

Une façon naturelle d'acquérir des connaissances sur le domaine du problème que nous voulons résoudre est d'étudier les systèmes déjà construits, en essayant de comprendre comment les problèmes pratiques ont été abordés dans la littérature scientifique.

Cependant, revoir les systèmes de recommandation de musique existants n'est pas une tâche facile : les systèmes commerciaux sont complètement fermés, de sorte que seules des suppositions peuvent être faites au sujet de la technologie sous-jacente, en observant attentivement leurs interfaces utilisateur, l'architecture générale et les excentricités. D'autre part, les systèmes non commerciaux sont habituellement des projets ouverts- les sources sont disponibles. Cela permet de surveiller de près quelles sont les principales tendances dans le domaine du développement de recommandation de musique.

Actuellement, nous assistons à quatre approches différentes pour donner des recommandations musicales. Dans ce qui suit, nous allons seulement regarder un petit échantillon de recommandations de musique que nous pensons être représentatifs pour chaque approche spécifique.

3.5.1 Le filtrage collaboratif : Ringo, Racofi

Les techniques de Filtrage Collaboratif (FC) ont été largement appliquées dans le domaine de la musique. FC utilise les métadonnées éditoriales et culturelles. Les premières recherches ont été basées sur le feedback explicite, les votes sur les chansons ou les artistes. Cependant, le suivi des habitudes d'écoute de l'utilisateur est devenu le moyen le plus commun dans la recommandation de musique. Dans ce sens, FC doit s'occuper du feedback implicite (au lieu de votes explicites).

Parmi les exemples les plus représentatifs pour la technique basée sur le feedback explicite sont *RINGO* et *RACOFI*.

RINGO, décrit dans (Shardanand, 1994), est le premier système de recommandation de musique basé sur le filtrage collaboratif et le feedback explicite. L'auteur applique l'approche FC basée sur l'utilisateur (voir section 2.3.1). La similarité entre les utilisateurs

est calculée avec la corrélation de Pearson normalisée (voir équation 2.2). *RINGO* compare les profils d'utilisateurs pour déterminer quels utilisateurs ont un goût similaire (ils aiment les mêmes albums et n'aiment pas les mêmes albums). Une fois que les utilisateurs similaires identifiés, le système peut prévoir comment l'utilisateur actif aimerait un album/artiste qui n'a pas encore été estimé en calculant une moyenne pondérée de toutes les estimations données à cet album par les autres utilisateurs qui ont des goûts similaires (URL 10).

RACOFI est un projet de recherche dirigé par l'Université du Nouveau-Brunswick (Canada). *RACOFI* signifie « *Rule-Applying Collaborative Filtering Systems* ». *RACOFI* combine le filtrage collaboratif basé sur les votes et un ensemble de règles de logique basées sur les clauses de Horn (Anderson *et al.*, 2003). Les règles, basées sur les technologies du Web sémantique : *OWL* et *RDF*, sont appliquées après que les estimations aient été recueillies. Les cinq dimensions d'évaluation qu'ils définissent sont : l'impression, les paroles, la musique, l'originalité et la production. L'objectif des règles est de mesurer le rendement du filtrage collaboratif et de favoriser les articles avec lesquels l'utilisateur sera le plus familiarisé. Ceci laisse ajuster les estimations prévues avec des règles comme :

« *Si un utilisateur évalue à 9 l'originalité d'un album par l'artiste X, puis l'estimation pour l'originalité, prédite pour cet utilisateur, pour tous les autres albums de l'artiste X, est augmentée par une valeur de 0.5.* »

Le feedback implicite dans le domaine de la musique est habituellement recueilli auprès des habitudes d'écoute. Le principal inconvénient est que la valeur que l'utilisateur attribue à un article n'est pas dans une plage prédéfinie (par exemple de 1 à 5 ou *aime / déteste*). Au lieu de cela, l'interaction entre des utilisateurs et des objets est décrite par le nombre total de lectures.

Les recommandations sont généralement effectuées au niveau d'artiste, mais les habitudes d'écoute sont au niveau des chansons. Dans ce cas, un processus d'agrégation, de la chanson jouée, au nombre total de lectures d'artiste, est nécessaire pour avoir la note globale d'un artiste.

3.5.2 Le filtrage à base de contenu : manuel, automatique

Les systèmes de recommandation à l'aide de filtrage par contenu sont basés sur la similarité entre chaque paire d'articles. Les méthodes à base de contenu audio sont utilisées pour classer des titres de musique, basés sur la similarité audio. Ainsi, un système de recommandation doit calculer la similarité entre les chansons, et utiliser cette information pour recommander la musique.

La similarité des artistes peut également être calculée, en regroupant les résultats de similarité de chansons. Il y a deux façons d'annoter les chansons : automatiquement ou manuellement. Les paragraphes suivants présentent chaque approche.

La tâche d'annoter automatiquement la musique avec les étiquettes de texte (appelée « *autotagging* ») est indispensable pour la création à grande échelle d'un moteur sémantique de découverte de musique. L'étiquetage automatique implique la génération automatique d'étiquettes, basée sur l'analyse du signal audio.

Les systèmes d'annotation ont deux composantes : *l'extraction* de caractéristiques musicales et *l'apprentissage*. Utilisant des chansons manuellement annotées, Barrington *et al.*, (2010) apprennent un classificateur qui peut prévoir des étiquettes pour une chanson non annotée basée sur les caractéristiques audio qui sont extraites de la chanson. (URL 11)

La qualité et la quantité des données d'entraînement affectent beaucoup la performance du système de marquage automatique. Un problème pour la communauté de recherche documentaire de musique est le manque d'un grand ensemble de données, correctement étiqueté. Ce problème persiste partiellement à cause de l'incapacité de librement distribuer un grand corpus de musique de haute qualité sans violer le droit d'auteur. Un autre obstacle est l'absence d'un vocabulaire standard d'étiquettes de musique.

Dès que l'audio a été sémantiquement annoté et la similarité parmi les chansons a été calculée, le filtrage à base de contenu pour un utilisateur donné est assez simple. Il est fondé sur la présentation des chansons (ou d'artistes) qui sont semblables au profil d'utilisateur.

La tâche d'annoter manuellement la musique est très fastidieuse, mais peut être plus précise que les méthodes d'extraction automatique. L'approche de Pandora est basée sur la description manuelle du contenu audio.

Un remarquable système de recommandation de musique est *Pandora* (www.pandora.com), fondé par Tim Westergen en 2000 basée sur le projet de *Music Genome*¹⁵. La façon dont cela fonctionne est que chaque chanson est représentée par un vecteur d'environ 400 caractéristiques, appelées gènes, chacune reçoit un numéro entre 1 et 5 par incréments de 0.5.

Par exemple, il y a des gènes pour le type d'instrument, pour le style de musique et pour le type de paroles. Les vecteurs de chanson sont construits par des experts et des musiciens qualifiés, chaque chanson prenant environ 20-30 minutes pour être évaluée. À partir du 2007, la bibliothèque de génome de musique contient plus de 500,000 chansons et de 35,000 artistes contemporains (Lavene, 2010). De plus, selon le FAQ sur le site de *Pandora*, environ 15,000 nouveaux vecteurs de chanson sont ajoutés à la bibliothèque chaque mois.

Quand un utilisateur écoute une chanson, une liste de chansons semblables peut être construite en utilisant une mesure de similarité comme la similarité vectorielle standard. Un avantage de l'approche de *Pandora* consiste en ce que ses auditeurs aient accès à la musique dans la longue file d'attente, parce que les experts peuvent construire des vecteurs pour les chansons moins populaires, par exemple, les nouvelles chansons de musiciens qui ne sont pas très connus ou les vieilles chansons qui ne sont pas encore à la mode.

Afin de raffiner ses recommandations, *Pandora* recueille également les votes des utilisateurs pour permettre à ses algorithmes d'ajuster les poids des caractéristiques et personnaliser les suggestions futures. Le but ultime de *Pandora* est d'offrir un mélange de familiarité, de diversité et de découverte.

¹⁵ Music Genome: Le 6 Janvier 2000, un groupe de musiciens et de mélomanes se sont réunis avec l'idée de créer l'analyse la plus complète de la musique.

3.5.3 Le filtrage contextuel : PAPA¹⁶

Même si les nombreux systèmes de recommandation musicale ont résolu la plupart des questions, il y a des problèmes qui ont été ignorés, tel que le changement au fil du temps des intérêts des usagers; ces facteurs doivent également être étudiés. Pour permettre à la recommandation de suivre les habitudes d'utilisateurs, quelques travaux proposent d'ajouter une séquence de temps dans les listes de lecture, pour fournir une meilleure recommandation, en appliquant le feed-back des utilisateurs et d'apprendre leurs comportements d'écoute de musique. (Liu *et al.*, 2009)

Eggen *et al.*, (2002) définissent le contexte d'utilisation comme le monde réel dans lequel la musique se fait entendre, que ce soit une fête, soirée romantique ou un voyage en voiture ou en train. Ils pensent que l'utilisation de ce concept est un point de départ puissant pour créer une liste de lecture ou comme un principe pour organiser une collection de musique.

Dans le langage quotidien, la préférence de musique et le goût musical sont utilisés pour faire référence au même concept. Nous faisons une distinction entre les deux, après les définitions données par Abeles (1980). Le goût musical est défini comme l'engagement à long terme d'une personne pour un idiome particulier de musique : « *Musical taste is defined as a person's slowly evolving long-term commitment to a particular music idiom* ». D'autre part, la préférence de musique est définie comme la préférence temporaire d'une personne pour un contenu musical particulier, dans un contexte particulier d'utilisation.

Selon la définition de Dey (2001), n'importe quelle information qui peut être employée pour caractériser la situation d'une personne, d'un endroit ou d'un objet de considération compose son contexte. L'auteur différencie quatre types de contexte primaire : *l'endroit, l'identité, le temps et l'activité*. Dans le domaine de recherche documentaire de musique, il existe déjà une série de systèmes qui capturent le temps, l'identité d'utilisateur et l'endroit, comme par exemple la connexion d'*Audioscrobbler*¹⁷ de

¹⁶ Papa : « *Physiology and Purpose-Aware Automatic Playlist Generation* »

¹⁷ <http://www.audioscrobbler.net/>

Last.fm. Cependant, cette information est rarement employée pour décrire un contexte d'utilisation et à notre connaissance elle n'a pas été jusqu'ici employée pour l'accès personnalisé aux collections de musique.

Dans le domaine de filtrage contextuel quelques systèmes ont essayé de différencier les contextes d'écoute : le système de recommandation de musique *M3* décrit par Lee *et al.*, (2006), emploie une approche de raisonnement par cas en deux étapes, pour la recommandation contextuelle. D'abord, les informations sur la saison, le mois, le jour de la semaine, le temps et la température sont utilisées pour déduire si l'utilisateur veut écouter de la musique. Cette décision est prise par le raisonnement par cas à partir de l'histoire d'écoute de l'utilisateur. Si la musique est susceptible d'être désirée, une deuxième étape de raisonnement par cas déduit si la musique devrait être lente, rapide ou peut avoir n'importe quel tempo. Ils estiment le tempo à partir du tag de genre, en supposant que les chansons appartenant à des genres « *Ballade* » et « *R & B* » sont lentes alors que les chansons appartenant à « *Rock / Metal* » et « *Danse* » sont rapides.

Dans (Cho *et al.*, 2006), un système de recommandation de musique sensible au contexte est basé sur les conditions météorologiques (température, humidité, les conditions météorologiques actuelles et prévues) et le temps (saison et l'heure de la journée). De plus, les données intègrent également le niveau de bruit ambiant enregistré par un microphone et l'éclairage mesuré par un capteur. Les données obtenues sont traitées par un réseau bayésien qui infère le contexte actuel. Explicitement, les profils d'utilisateurs créés sont ensuite utilisés pour recommander des chansons à l'égard du contexte.

Indépendamment de rassembler les informations sur l'environnement, l'information de contexte d'écoute peut également comporter des informations directes sur l'état courant d'utilisateur. Par exemple, le système adaptatif pour la génération de listes de lecture, appelé PAPA et le lecteur de musique « *BODiBEAT*¹⁸ » (déjà disponibles dans le commerce) mesurent certains *biosignaux* (tels que le pouls) d'utilisateur et fournissent un feedback immédiat pour la musique actuellement jouée. (Oliver *et al.*, 2006)

¹⁸ <http://www.yamaha.com/bodibeat/>

3.5.4 Les hybrides : Genius, LastFm

Afin d'essayer de corriger les problèmes avec la recommandation de musique, il y a eu des tentatives pour combiner les sorties des différents systèmes de recommandation. Un exemple de cela est la recommandation hybride de musique, qui est une option extrêmement intéressante, car elle mélange ensemble les données de plusieurs types de filtrage. Il y a deux types de fonctionnement de la recommandation hybride : la combinaison des sorties individuelles ou leurs intégration dans un seul algorithme.

Le système de recommandation *Genius d'iTunes* utilise le service *Gracenote MusicID*¹⁹ pour relever les empreintes digitales des chansons dans la bibliothèque de musique d'un utilisateur et pour identifier le nom de la chanson, l'artiste, l'album, etc. Ces métadonnées sont ensuite utilisées pour identifier les chansons dans la base de *Genius*. Les détails exacts de l'algorithme sont un secret commercial d'*Apple Inc*²⁰. *Genius* semble qu'il utilise le filtrage collaboratif pour comparer les métadonnées de la chanson de test avec la gigantesque base de données des ventes de musique d'*iTunes* (plus de 50 millions de clients qui ont acheté plus de 5 milliards de chansons), ainsi que l'histoire d'écoute et les votes recueillis auprès d'utilisateurs d'*iTunes*.

Quand il est d'abord initialisé, *Genius* analyse la bibliothèque de musique d'un utilisateur et compile toutes les données de filtrage collaboratif nécessaires pour construire des listes de lecture de la bibliothèque, basée sur n'importe quelle chanson de test donnée. Les expériences informelles avec *Genius* (Barrington *et al.*, 2009) donnent quelques indices sur son fonctionnement et vérifient qu'il n'utilise pas l'analyse du contenu directement. Par exemple, si nous effaçons les informations de métadonnées *ID3*²¹

¹⁹ Gracenote, Inc, anciennement appelée CDDB (Compact Disc Data Base), est une société qui fournit les logiciels et les métadonnées pour les entreprises qui permettent à leurs clients de gérer et de rechercher dans les médias numériques

²⁰ <http://www.apple.com/ca>

²¹ ID3 est le nom des métadonnées pouvant être insérées dans un fichier audio comme MP3. Ces métadonnées permettent d'avoir des informations sur le contenu du fichier comme le titre, le nom de l'interprète, les commentaires et encore beaucoup plus.

associées à une chanson en format *MP3*²² donnée, ou ajoutons une chanson à la bibliothèque qui est inconnue à *Gracenote* (par exemple, un nouvel enregistrement par une bande inconnue), *Genius* ne parvient pas à recommander de la musique. De plus, si nous choisissons une chanson de test qui est très atypique pour le style de l'artiste ou de l'album qui comporte la chanson, *Genius* recommande de la musique qui représente les aspects les plus communs de l'artiste.

Les listes de lectures alors générées contiennent 25, 50, 75 ou 100 pistes et peuvent être rafraîchies ou sauvegardées. La barre latérale de *Genius* propose des pistes similaires non présentes dans la bibliothèque mais achetable sur *iTunes Store*. Les versions actuelles des *iPods*²³, *iPhones*²⁴ et *iPad*²⁵ disposent aussi de la fonction *Genius*. (URL 12).

Propriété de *CBS Interactive*²⁶, avec les bureaux dont le siège social est à Londres, et un site Web enregistré dans les États fédérés de Micronésie pour atteindre le premier niveau de code de domaine de pays « *.fm* », *Last.fm* est à la fois un site de réseautage social, un système de recommandation de musique et une radio sur Internet.

Le 20 mai 2007, *CBS Interactive*, conglomérat dont le principal actionnaire est le milliardaire *Sumner Redstone*, a acquis *Last.fm* pour 280 millions de dollars américain (URL 14).

Comme *Last.fm*, la plupart des systèmes de découverte de musique ont utilisé le filtrage collaboratif. Bien qu'une grande partie du travail théorique dans le domaine se soit concentrée à améliorer les algorithmes, l'innovation de *Last.fm* a été dans l'amélioration des données avec lesquelles travaillent les algorithmes.

Afin d'approvisionner le site en statistiques d'écoutes, il est nécessaire d'installer un plugin nommé *Audioscrobbler*, et qui retransmet à *Last.fm* la liste des morceaux que

²² Le MPEG-1/2 Audio Layer 3, plus connu sous son abréviation de MP3, est la spécification sonore du standard MPEG-1/MPEG-2, du *Moving Picture Experts Group* (MPEG).

²³ L'iPod est un baladeur numérique d'Apple, lancé le 23 octobre 2001.

²⁴ iPhone est une famille de Smartphones conçue et commercialisée par Apple Inc. depuis 2007

²⁵ L'iPad est une tablette électronique conçue et développée par Apple

²⁶ <http://www.cbsinteractive.com>

l'utilisateur a écouté avec son lecteur multimédia ou son baladeur numérique. Il est également possible d'utiliser un lecteur multimédia qui intègre ce plugin. Ceci permet au système de fournir une analyse détaillée de la musique écoutée par chaque utilisateur, montrant par exemple ses artistes ou ses morceaux favoris sur sa page personnalisée (mais accessible à tous).

Audioscrobbler a été à l'origine un projet informatique de *Richard Jones* lorsqu'il était à l'Université de Southampton au Royaume-Uni. Il s'agissait alors d'un site web communautaire récupérant et permettant d'afficher des statistiques sur la musique qu'écoutaient ses membres. Il a développé le premier plugin et créé une *API* pour la communauté, afin de pouvoir appuyer toutes sortes de systèmes d'exploitation et de plateformes (URL 14).

Comparé au fait de compter sur les utilisateurs pour manuellement fournir leurs préférences, cette capture de données automatique est meilleure. *Last.fm* génère automatiquement des listes de lecture personnalisées en utilisant les algorithmes recommandation basés essentiellement sur le filtrage collaboratif. Ces algorithmes considèrent les tags des utilisateurs ou le comportement d'écoute, donc on parle d'une recommandation hybride de type commutation. (Voir section 2.3.5.b)

En accord avec l'aspect de réseau social, le système affiche également les informations sur d'autres utilisateurs qui ont des goûts similaires et permet aux utilisateurs de communiquer entre eux indépendamment du fait qu'ils se sont proposés comme des amis.

Utilisant l'ensemble de données de *Last.fm* pour calculer la similarité entre artistes, Celma (2008) applique d'abord le filtrage collaboratif et puis réorganise et combine les résultats selon la distance sémantique de l'étiquetage social. Dans ce cas-là, la méthode de chaîne en cascade a du sens. Les premiers résultats sont obtenus en tenant compte de la logique du type : « *les gens qui écoutent 'X' écoutent aussi 'Y'* ». Alors, le deuxième filtrage ne promeut pas des artistes 'Y' qui sont les plus proches, dans l'espace d'annotation sémantique, à l'artiste 'X'.

Last.fm a les capacités que les nombreux systèmes de filtrage collaboratif ne peuvent que souhaiter (voir Figure 3.5). La quantité d'informations gérées par *Last.fm* est énorme, composée de plus de 58 milliards de morceaux « *scrobbled* » à ce jour, un nombre qui augmente à un taux de plus de 400 millions de titres par semaine. (URL 15)



Figure 3.5 La quantité d'informations gérées par Last.fm (Ellis *et al.*, 2011)

3.6 Conclusion

Ce chapitre a présenté tous les éléments de recommandation de musique : le profil d'utilisateur et la représentation d'article et les méthodes de recommandation existantes. Les préférences d'utilisateur dépendent du type d'auditeur et de son niveau d'engagement avec la musique. Dans la recommandation musicale, la similarité basée sur l'artiste / la chanson est la manière la plus commune de prévoir les recommandations. La représentation du profil d'artiste est la première étape pour calculer la similarité et pour fournir des recommandations aux utilisateurs.

Plus précisément, dans le Chapitre 3, nous avons passé en revue quatre classes de systèmes de recommandations de musique : les systèmes basés sur le filtrage collaboratif, les systèmes basés sur le filtrage à base de contenu, les systèmes basés sur le filtrage contextuel et finalement les systèmes hybrides. Nous avons constaté que ces solutions ne sont pas bien adaptées pour les besoins spécifiques du problème de la recommandation musicale (tel que présentés dans le Chapitre 3).

Chapitre 4 Conception de Mures

Nous avons vu dans le chapitre précédent que chaque type de méthodes de recommandation, exemplifiés par les applications les plus représentatives, pris indépendamment, ne répondent pas aux besoins des usagers en matière de recommandation de musique. Parce qu'il n'existe pas de solutions qui répondent à ces besoins et après avoir étudié et analysé les systèmes dans l'état de l'art (Chapitre 2 et 3), nous nous en sommes inspirés pour concevoir *MURES* (*MUSIC REcommender System*), notre système de recommandation de musique. Dans ce chapitre nous étudierons plus loin le problème de recommandation et plus précisément la recommandation de musique. Nous commençons par une description générale de notre système et ensuite nous détaillons la structure de ses différents modules et ses fonctionnalités.

4.1 Présentation générale

De nos jours, la musique est utilisée (consommée) à diverses occasions et peut accompagner diverses activités. *Mures* essaie de se mettre à l'écoute de l'utilisateur, en lui proposant une liste de chansons en vue d'obtenir un état d'esprit souhaité, tel que romantique, heureux, triste, etc. De la même manière, un usager écoute un type de musique durant une fête, un autre type de musique le matin lorsqu'il se rend au travail, un autre type de musique lorsqu'il travaille ou il étudie et certainement un autre type de musique lors d'une soirée romantique. Étant donné ces aspects, le contexte d'utilisation de la musique représente un des plus importants éléments que *Mures* prend en compte lors de recommandations musicales.

En ce moment il existe une multitude des systèmes de recommandation de musique que l'utilisateur peut utiliser. De plus, la plupart des systèmes mettent à la disposition de l'utilisateur une série d'APIs (sous la forme de services web) qui offrent des informations en format XML. Cependant, aucun des systèmes n'a pas essayé de profiter des informations que les autres systèmes génèrent (en mode coopératif). *Last.fm* met à la disposition les tags qui caractérisent chaque artiste ou chanson, le nombre de fois qu'un artiste ou une chanson

a été écoutée, ainsi que les coefficients de similarité entre les différents artistes. D'un autre côté, *MusicBrainz* offre aussi une similarité entre les différents artistes, les évaluations pour la majorité des chansons, ainsi qu'une série d'informations d'intérêt général (des métadonnées). Puisque chacun des systèmes énumérés a ses limites, *Mures* essaie de corriger ces problèmes, en combinant les informations pertinentes que ces deux systèmes offrent (les tags, les évaluations, les similarités entre les artistes, etc.). Ainsi, *Mures* propose à l'utilisateur une liste de chansons diversifiée, qui va être composée par des chansons choisies parmi les préférences de l'utilisateur (selon les informations contenues dans le système *Mures*) et des nouvelles chansons (à partir des corrélations des données en provenance de *Last.fm* et *MusicBrainz*) qui sont susceptibles d'être appréciées par l'utilisateur.

Un autre objectif du système *Mures* est de collecter des données provenant de sources externes, en se basant sur les techniques de *crawling* et sur les flux RSS²⁷, pour offrir plus d'informations liées à la musique comme les nouveautés musicales, les prochains concerts, les paroles et les artistes similaires.

Mures, opère sur deux plans : *personnel* et *public*. Sur le plan personnel, l'utilisateur dispose d'un espace personnel où il a la possibilité de gérer sa collection musicale, d'avoir des recommandations personnalisées à partir de sa collection personnelle et de localiser ses chansons. En revanche, sur le plan public, l'utilisateur a la possibilité d'avoir de recommandations à partir de la collection musicale du *Mures* et de partager les connaissances liées aux chansons avec les autres utilisateurs.

Mures offre plusieurs fonctionnalités Web 2.0²⁸ comme les votes, les tags, le flux RSS tout en utilisant des techniques d'application Internet riches telles qu'*AJAX* pour améliorer l'expérience d'utilisateur. L'utilisation, ainsi que la combinaison de plusieurs jeux de données et des services web disponibles gratuitement sur le Web (voir Figure 4.1), tels que la base de données de musique de *MusicBrainz*, les tags et les habitudes d'écoute

²⁷ RSS (sigle venant de l'anglais « *Rich Site Summary* ») désigne une famille de formats XML utilisés pour la syndication de contenu Web.

²⁸ L'expression « Web 2.0 » utilisée par Dale Dougherty en 2003, diffusée par Tim O'Reilly en 2004 et consolidée en 2005 avec la *position paper* « What Is Web 2.0 » s'est imposée à partir de 2007.

de *Last.fm* et les informations liées à la musique comme les nouveautés musicales, les prochains concerts de *Amazon* et les paroles des chansons de *LyricWiki* constitue une des originalités de notre application.

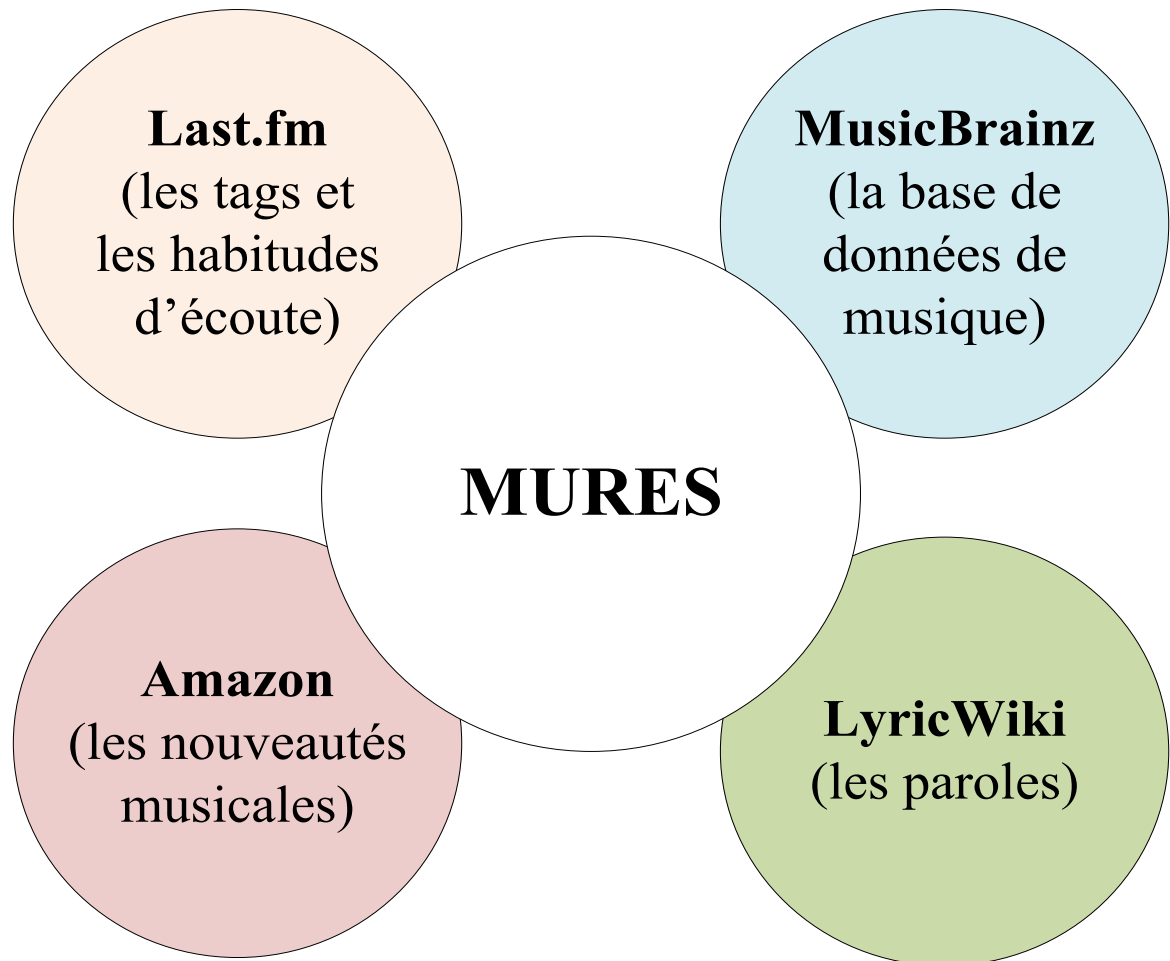


Figure 4.1 Représentation des interactions de Mures dans le Web

4.2 L'architecture du système

La figure 4.2 montre l'architecture de notre système, avec ses modules et les interactions entre eux.

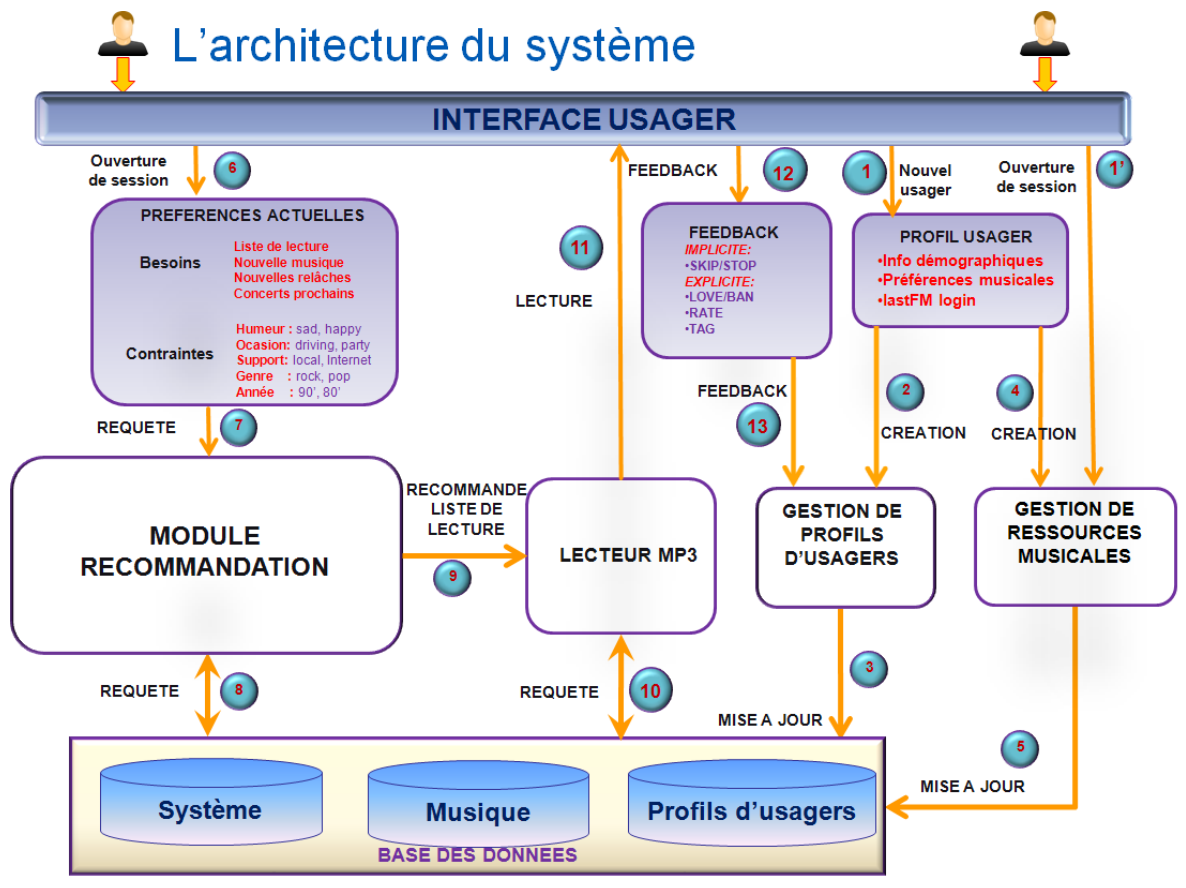


Figure 4.2 Architecture du système de Mures

Cette section décrit notre système *Mures* étape par étape. Nous montrerons son fonctionnement global.

Pour utiliser notre système, la première étape consiste à la création d'un compte utilisateur (1). Une fois qu'il a choisi un nom d'utilisateur et un mot de passe, l'utilisateur est invité à saisir quelques informations personnelles comme *l'âge, la localisation, la langue* ainsi que *le pseudonyme de Last.fm* et quelques préférences musicales (2). Toutes ces informations vont générer son profil d'utilisateur (3).

Une fois son profil créé, la deuxième étape pour l'utilisateur est de rendre disponible sa collection musicale personnelle. Pour être capable de proposer des listes de lecture à partir de la collection personnelle d'utilisateurs, notre système doit connaître le chemin où se trouvent ses fichiers *mp3* (4).

Cette étape est suivie de l'extraction des métadonnées (*les tags ID3*²⁹) de chaque fichier *mp3*. Ces métadonnées peuvent être *le titre, le nom d'artiste, le genre musical, l'album, la date de sortie, etc.* Ces informations seront ensuite acheminées vers sa base de données de l'espace personnel en utilisant un module conçu spécialement pour cette tâche (5). Ce module fait la lecture récursive de tous les fichiers dans le chemin spécifié et extrait *les tags ID3* (voir section 3.5.4). Chaque chanson dans la base de données (BD) a un identifiant unique qui est le chemin, suivi du nom de la chanson, de l'artiste et de l'album. Dans la BD, seules les métadonnées ID3 sont enregistrées. On ne s'intéresse pas aux fichiers *mp3* en entier.

Pour manipuler cette collection personnelle, plusieurs fonctionnalités de base sont disponibles : ajouter des fichiers dans la bibliothèque, modifier les fichiers existants ou bien effacer les fichiers.

Une fois cette étape terminée, à partir de la collection personnelle d'utilisateurs, *Mures* peut recommander plusieurs types de listes de lecture selon les préférences spécifiques (6). Ces listes de lecture peuvent être classifiées et réutilisées ou partagées avec les autres utilisateurs du système.

Une fois connecté dans notre système, l'utilisateur peut recevoir comme recommandations (7), des chansons générées à partir de la collection générale de *Mures*, en se basant sur son profil d'utilisateur et ses besoins spécifiques (courants) (8).

Indépendamment de son choix de *collection personnelle* ou de *collection générale* de *Mures*, l'utilisateur a encore plusieurs choix disponibles : il peut choisir la musique de ses artistes favoris, des artistes similaires avec eux, ou il peut choisir la musique appropriée

²⁹ ID3 est le nom des métadonnées pouvant être insérées dans un fichier audio comme MP3. Ces métadonnées permettent d'avoir des informations sur le contenu du fichier comme le titre, le nom de l'interprète, les commentaires et encore beaucoup plus.

pour le contexte d'utilisation : en se rendant au travail, aux fêtes ou pendant une soirée romantique. La seule différence c'est que dans le cas de la collection générale de *Mures*, seules les 30 premières secondes sont jouées pour respecter les droits d'auteurs (9). *Mures* cherche les chansons recommandées dans sa base de données (10).

Pour écouter les fichiers *mp3* nous avons aussi développé un petit *lecteur Mp3* (11) (voir Figure 4.3). Pour l'enrichissement du contenu musical et pour avoir meilleures recommandations nous avons ajouté à ce module plusieurs fonctionnalités de web 2.0 comme les tags, les votes, les commentaires, aimer/bannir une chanson/artiste (12). Avec chacune de ces options le profil d'utilisateur sera mis à jour (13).

The screenshot shows a web browser window with the URL `localhost/player3/im_doctor.php`. The page features a navigation menu with links for Home, User collections, Recommendations, User playlists, Similar users, Events, and Logout. The main content area displays the following information:

- Song:** Lay Your Hands On Me **Album:** Cross Road **Artist:** Bon Jovi
- Playlist number:** 626 **Personal collections - favorites song:** both
- Media player controls (play, pause, stop, next, previous) and a 'BUY' button.
- A star rating of 4.0/5 (1 vote cast).
- Bon Jovi Rating: 4.0/5 (12 vote cast).
- You rated Bon Jovi with 4.4/5 (7 vote cast).
- A 'TAG' button.
- Tagged as:** [rock](#) [hard rock](#) [classic rock](#) [80s](#) [hair](#) [metal](#) [bon jovi](#) [pop](#)
- Similar Artists:** Jon Bon Jovi, Richie Sambora, Def Leppard, Europe, Poison, Aerosmith, Skid Row, Bryan Adams, Cinderella, Gotthard, Firehouse.
- A playlist of 23 songs, including:
 19. Phil Collins - Phil Collins...Hits - Sussudio
 20. Phil Collins - Phil Collins...Hits - One More Night
 21. Phil Collins - Phil Collins...Hits - A Groovy Kind Of Love
 22. Elton John - Greatest Hits 1970-2002 CD1 - Bennie And The Jets
 23. Elton John - Greatest Hits 1970-2002 CD1 - Daniel

Figure 4.3 Lecteur mp3 de Mures

4.3 La gestion de ressources musicales

Toutes les chansons introduites dans la base de données personnelle, sont caractérisés par des métadonnées (voir Figure 4.4) de type *ID3 TAG* et elles peuvent appartenir à l'une des trois catégories suivantes : *ID3v1*, *ID3v1.1* et *ID3v2*.

ID3 est un format audio très populaire d'étiquetage des fichiers musicaux souvent utilisé par les réalisateurs de logiciels et de matériels autour du monde. Une étiquette *ID3* est un conteneur de données dans un fichier *mp3* audio stocké dans un format prédéfini. Ces données contiennent généralement le nom d'artiste, le titre de chanson, l'année et le genre du fichier audio courant.

- ***ID3v1*** : la couche audio I, II et III (*MP3*) du format *MPEG* n'a aucune manière propre de sauvegarder les informations sur le contenu, excepté certains paramètres simples booléens comme ceux précisant que le fichier est sous copyright ou encore qu'il est privé. En ajoutant un petit morceau de données supplémentaires à la fin du fichier on a pu permettre au fichier *MP3* de stocker des informations à propos de *l'origine du fichier* et pas seulement des informations audio. Le placement des tags du nom de ces données a été probablement choisi car il y avait peu de chance qu'elle puisse déranger les décodeurs. Afin de les rendre plus facile à détecter, une taille fixe de 128 octets a été choisie;
- ***ID3v1.1*** : les tags *ID3v1* sont plus faciles à mettre en application pour les programmeurs, mais leurs utilisation est difficile et frustrante pour les usagers avec leurs propres idées créatrices. Puisque le tag *ID3v1* a une taille fixe et aucun espace « *réservé pour un usage futur* », il n'y a pas vraiment de possibilité d'amélioration. Celui qui trouva une solution fut Michael Mutschler. Il a apporté une amélioration tout à fait intelligente aux tags *ID3v1* : les tags *ID3v1.1*;
- ***ID3v2*** : est un nouveau système de taggage qui permet d'intégrer des informations enrichissantes et appropriées dans un fichier audio. En termes plus clairs, *ID3v2* est un morceau des données ajouté au début des données audio binaires. Chaque tag

ID3v2 contient un ou plusieurs plus petits morceaux d'information, appelés les *cadres*. Ces cadres peuvent contenir n'importe quel genre d'informations ou données auxquelles nous pourrions penser comme le titre, l'album, l'interprète, le site Web, les paroles, les pré-réglages d'égaliseur, les images, etc. (URL 16)

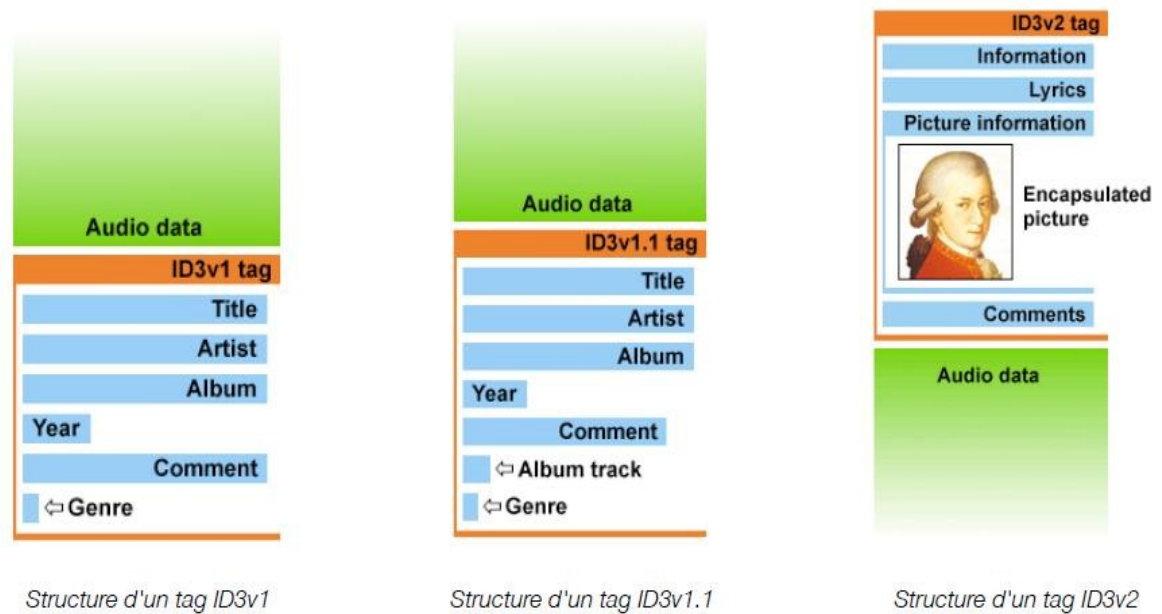


Figure 4.4 Représentation graphique des tags *ID3* (URL, 16)

Pour le *module de gestion de ressources musicales*, une première étape a été de construire la base de données de *Mures*. Pour débiter, nous avons téléchargé l'image de la base de données de *MusicBrainz*. Pour chaque entité (artiste, album et chanson) nous avons établi un identifiant unique, qui est le *GID*³⁰ utilisé par *MusicBrainz*. Pour l'organisation, chaque artiste peut avoir un ou plusieurs albums et chaque album peut avoir une ou plusieurs chansons.

Dans un deuxième temps, nous avons essayé de remplir la base de données ainsi transformée, avec des informations d'intérêt que les autres systèmes de recommandation de

³⁰ GID - sorte d'identifiant unique de *MusicBrainz* pour chaque article

musique partagent à travers des services web. Nous avons d'abord essayé d'intégrer les tags, le nombre de fois qu'un artiste ou une chanson a été écoutée et la similarité entre les artistes que *Last.fm* partage. Un premier problème a été la compatibilité entre les informations que les deux systèmes partagent (*Last.fm* et *MusicBrainz*). Il faut mentionner que pour presque 80 % des cas, l'identifiant unique (le *GID* de *MusicBrainz*) se retrouve dans les résultats obtenus à partir de *Last.fm*. Pour le reste de 20 % des cas, nous avons créé une fonction de type *soundex*³¹, pour calculer le coefficient de similarité entre la série (artiste, album, chanson) de *Last.fm* et son correspondant de *MusicBrainz*. Nous avons considéré que lorsque le coefficient *soundex* (*CS*) ≥ 0.95 (avec *CS* prenant des valeurs dans l'intervalle [0..1]), nous pouvons parler de la même série dans les deux sources.

Dans le *module de gestion de ressources musicales*, nous avons développé un sous module spécial pour l'extraction de ces métadonnées (les tags *ID3*) de chaque fichier mp3. Ce module fait la lecture récursive de tous les fichiers dans le chemin spécifié et extrait les tags *ID3*. Chaque chanson de la base de données *Mures* possède l'identifiant unique « *GID MusicBrainz* » et en plus, un autre identifiant composé du nom de l'artiste + le nom de l'album + le nom de la chanson. Les autres informations enregistrées sont le genre musical, la date de sortie de la chanson, la taille, la durée, la version *ID3*, les commentaires, la piste du *CD*, etc.

Dans la BD, seules les métadonnées sont enregistrées. Ces informations seront ensuite acheminées vers la base de données pour mettre à jour le profil d'utilisateur. Au moment de l'insertion un *déclencheur* va remplir chaque chanson avec son identifiant unique de *MusicBrainz*. Le *déclencheur* compare le *soundex* de chaque nom de chanson et artiste avec les valeurs similaires provenant de la base de données de *MusicBrainz*. Si le coefficient *soundex* est < 0.95 , *Mures* propose une liste d'options pour que l'utilisateur puisse

³¹ Soundex est un algorithme phonétique d'indexation de noms par leur prononciation en anglais britannique. L'objectif basique est que les noms ayant la même prononciation soient codés avec la même chaîne de manière à pouvoir trouver une correspondance entre eux malgré des différences mineures d'écriture.

choisir manuellement la chanson qui est sur le point d'être introduite dans la base de données.

Ensuite, la mise à jour est faite avec de requêtes *XML* (voir Figure 4.5) sur les services web de *MusicBrainz* et *Last.fm*. En ayant l'identifiant de chaque chanson/album/artiste notre système peut à présent profiter de toutes les informations disponibles sur les deux bases de données comme les « *playcounts* », les étiquettes de *Last.fm* et les votes de *MusicBrainz*.

```
ws.audioscrobbler.com/2.0/?method=track.getInfo&api_key=b25b959554ed76058ac220b7b2e0a0268&artist=cher&track=believe
- <lfm status="ok">
  - <track>
    <id>1019817</id>
    <name>Believe</name>
    <mbid/>
    <url>http://www.last.fm/music/Cher/_/Believe</url>
    <duration>239000</duration>
    <streamable fulltrack="0">0</streamable>
    <listeners>242321</listeners>
    <playcount>985241</playcount>
  - <artist>
    <name>Cher</name>
    <mbid>bfcc6d75-a6a5-4bc6-8282-47aec8531818</mbid>
    <url>http://www.last.fm/music/Cher</url>
  </artist>
  - <album position="1">
    <artist>Cher</artist>
    <title>The Very Best of Cher</title>
    <mbid>a7e2dad7-e733-4bee-9db1-b31e3183eaf5</mbid>
  - <url>
    http://www.last.fm/music/Cher/The+Very+Best+of+Cher
  </url>
```

Figure 4.5 Exemple simplifié d'une réponse au format XML (URL, 18)

Les résultats sont simples, on retrouve par exemple les albums dont l'artiste contient le mot « *Cher* » et le titre « *Believe* ». Les informations sont limitées à l'artiste et au titre, mais grâce au *GID* (sorte d'identifiant *MusicBrainz* pour chaque article), on peut récupérer

toutes les informations d'un article (couvertures *CD*, artiste, pistes du *CD*, année...) en renvoyant une requête *XML*.

Par la suite, il nous suffit de connaître la structure du fichier *XML* pour récupérer nos informations. Le principe est le même pour *Last.fm* ou *Amazon*, on compose des requêtes pour chercher des informations sur un artiste, un album ou une chanson. Le système des *GID* est présent par l'utilisation des identifiants : « *artist_id* », « *release_id* », « *track_id* ». Par ailleurs, puisqu'aucune information sur le genre musical n'est disponible sur *Last.fm*, *Amazon* ou *MusicBrainz*, on utilise les informations retrouvées dans les étiquettes *ID3*. ***La combinaison de ces trois jeux de données (Last.fm, Amazon ou MusicBrainz) disponibles gratuitement sur le web constituant une des originalités de notre système.***

4.4 La gestion de profils d'utilisateurs

En premier, la décision à prendre lors de la conception d'un système de recommandation de musique est de décider comment représenter les besoins des utilisateurs. La pertinence de représentation aura une responsabilité majeure sur la réussite de notre système de recommandations musicales.

Comme nous avons vu dans le chapitre précédent, la recommandation de musique est très dépendante du type d'utilisateur. La musique est un facteur important de transmission aux autres quelque chose de pertinent sur notre personnalité, notre histoire, etc. La modélisation de l'utilisateur est alors une étape cruciale dans la compréhension des préférences des utilisateurs.

Le fait de construire des profils exacts est une tâche clé puisque le succès de notre système dépendra dans une large mesure, de la capacité de représenter les intérêts actuels de l'utilisateur. Les trois étapes pour construire un profil d'utilisateur sont : *la génération*, *la maintenance* et *l'exploitation* des profils d'utilisateurs, en utilisant un algorithme de recommandation.

Un des aspects les plus importants d'un profil d'utilisateur est son *initialisation*. Dans notre système nous avons décidé de combiner deux méthodes :

- *manuelle* : *Mures* demande aux utilisateurs d'enregistrer leurs intérêts (par tags, mots clés), ainsi que des informations démographiques (âge, état matrimonial, sexe, etc.), des données géographiques (ville, pays, etc.) et des données psychographiques (intérêts, style de vie, etc.);
- *automatique* : *Mures* demande après le pseudonyme de *Last.fm* pour rendre complet le profil d'usager avec les informations *importées* à travers des sources externes (le profil obtenu à partir de l'API du site *Last.fm*).

La *maintenance* signifie qu'une fois le profil initialisé, il sera mis à jour en fonction des actions et préférences de l'utilisateur. Comme nous avons montré dans la section 2.2, une des problématiques fondamentales pour résoudre le problème de recommandation c'est d'enregistrer le feedback des utilisateurs. Le feedback d'usager peut être *explicite* ou *implicite* (voir Figure 4.6). Le feedback explicite vient généralement sous la forme d'estimations. Ce type de feedback peut être *positif* ou *négatif*. Dans notre système les estimations sont dans une échelle discrète de 0 à 5. Une autre possibilité est l'option d'une valeur binaire (aime/n'aime pas). Une autre manière pour recueillir le feedback explicite serait que *Mures* permette aux utilisateurs d'écrire des commentaires et d'ajouter des tags.

Notre système peut également recueillir le feedback implicite en surveillant les actions de l'utilisateur, en analysant les éléments suivants : le temps passé sur une page Web, les liens suivis de l'utilisateur, les habitudes d'écoute (dépistant le jeu, la pause, le saut et l'arrêt).

Une fois le profil d'utilisateur créé, la prochaine étape est d'exploiter les préférences des utilisateurs afin de leur proposer des recommandations intéressantes. L'objectif principal de notre système est de proposer, à l'utilisateur, les artistes intéressants et inconnus et leurs chansons disponibles, en se basant sur son goût musical (préférences).

The screenshot shows the Mures music player interface. At the top, there is a navigation bar with links: Home, User collections, Recommendations, User playlists, Similar users, Events, and Logout. Below this, a header bar displays the song information: "Song: Lay Your Hands On Me Album: Cross Road Artist: Bon Jovi".

The main content area includes a music player with playback controls (play, stop, previous, next) and a "BUY" button. To the right of the player is a star rating system showing 4.0/5 (1 vote cast). Below the rating, it says "Bon Jovi Rating: 4.0/5 (12 vote cast)" and "You rated Bon Jovi with 4.4/5 (7 vote cast)".

On the left side, there is a "104" counter, a heart icon, and a crossed-out icon. Below the player, there is a "Tagged as:" section with a list of tags: rock, hard rock, classic rock, 80s, hair metal, bon jovi, pop. A "TAG" button and a "See more..." link are also visible.

Figure 4.6 Feedback implicite et explicite dans Mures

La musique se distingue d'autres domaines de divertissement, comme les films, ou les livres. Une différence évidente est qu'une personne peut écouter plusieurs fois la même chanson. Donc, une fonctionnalité importante de *Mures* est de créer des listes de lecture à partir de la collection personnelle de l'utilisateur. Avec *Mures*, les listes de lecture peuvent être générées soit *à la main*, soit *automatiquement* pour refléter un état d'esprit, pour accompagner une activité ou pour explorer de nouvelles chansons pour la découverte de la musique.

Une autre fonctionnalité de *Mures* c'est *la recommandation des voisins*. Un des avantages principaux de créer des voisinages est qu'un utilisateur puisse explorer ses utilisateurs semblables, facilitant ainsi le processus de découverte de la musique inconnue et intéressante. En outre, il permet la création des réseaux sociaux, de groupes, reliant les personnes qui partagent les mêmes intérêts.

On retrouve dans le système *Mures* les fonctionnalités de base pour la gestion des listes de lecture, par exemple : ajouter, supprimer une chanson, trier et exporter la liste de lecture, copier le contenu locale sur un support externe : clé USB, cd, iPods et afficher toute sorte d'information liée à chaque chanson jouée.

De nos jours, le nombre de chansons et artistes disponibles sur le web augmente et en conséquence les collections personnelles ne cessent de grandir et la tâche de retrouver les chansons susceptibles d'être aimées devient très difficile. Pour y remédier, nous avons conçu et implémenté le module de recommandation, qui est l'objet de la prochaine section, afin de faciliter cette tâche.

4.5 Les options de recommandation

Cette section explique le processus de génération de listes de lecture dans Mures. Notre module de recommandation est de type *hybride* et nous distinguons deux types de filtrage : *contextuel* et *collaboratif*. Le premier type de recommandation est basé sur l'utilisateur, il spécifie lui-même ses critères : contexte d'utilisation, genres musicaux et par la suite, il analyse les résultats retournés. Dans le deuxième cas, la recommandation de listes de lecture se base sur les profils des usagers.

4.5.1 Le système de recommandation

Notre approche de recommandation appartient à la classe des systèmes de recommandation *hybride*. Selon la classification de Burke (2002), c'est un *hybride* de type *commutation* (voir la section 2.3.5), basculant d'un *filtrage contextuel* à un *filtrage collaboratif* (voir Figure 4.7).

La détermination de la technique appropriée dépend du contexte d'utilisation. Le système se doit alors de définir les critères de commutation, ou les cas où l'utilisation d'une autre technique est recommandée. La *commutation* entre le *filtrage contextuel* et le *filtrage collaboratif* est faite dépendamment des besoins des utilisateurs. Si l'utilisateur demande une liste de lecture pour une soirée romantique (pour une *situation thématique* en général : fête, voyage avec la famille, etc.) Mures va utiliser l'algorithme de *filtrage contextuel*. Si l'utilisateur veut *découvrir* de nouvelles chansons l'algorithme employé sera le *filtrage collaboratif*.

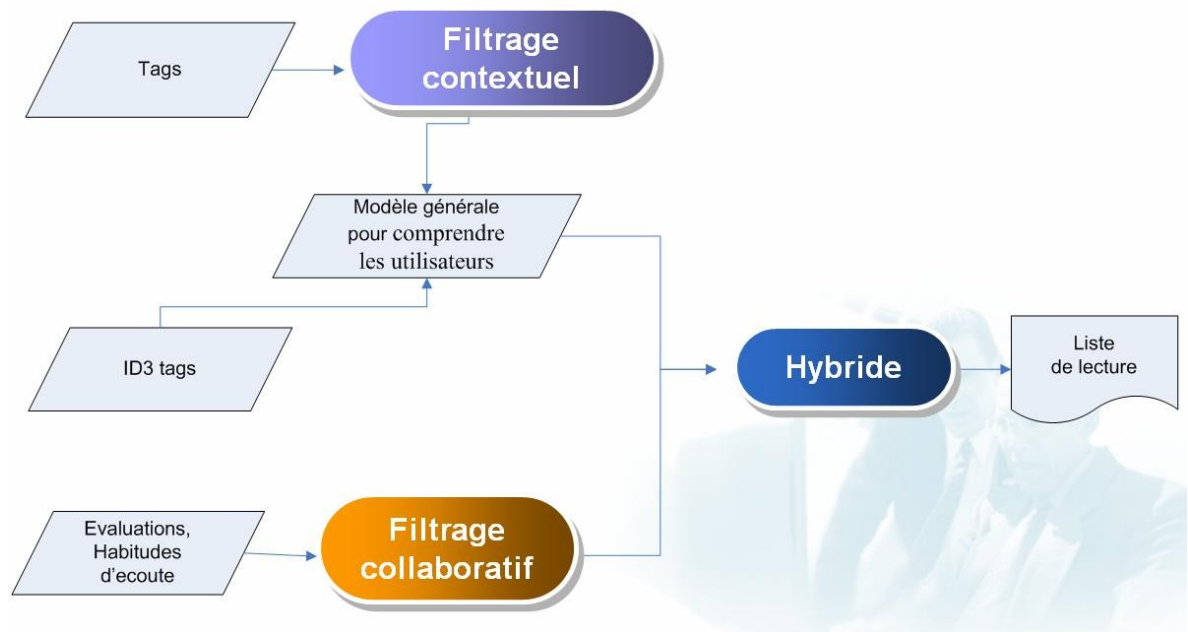


Figure 4.7 Le module de recommandation dans Mures

La première option est *le filtrage contextuel*. En général, un usager qui a besoin d'une liste de lecture peut fournir quelques informations caractéristiques de ce qu'il veut avoir. Ces informations initiales peuvent être classées en deux types : celles qui décrivent le contexte d'utilisation : *soirée romantique*, *en allant vers son travail* et celles qui décrivent les *genres musicaux*. Suivant le type de ces informations, nous distinguons plusieurs types de recommandation : la recommandation basée sur les tags, la recommandation basée sur les genres musicaux et un hybride de type *cascade* (voir la section 2.3.5).

Au moment de la création de la base de données *Mures* (voir la section 4.3), une étape particulière a été la création des tables qui contiennent les tags pour chacun des artistes et chacune des chansons. Ainsi nous avons créé deux tables : *artiste_tag* et *chanson_tag*, ayant les structures suivantes : ***artiste_tag*** (*artist_id*, *tag_id*, *source_id*, *fréquence*), respectivement ***chanson_tag*** (*chanson_id*, *tag_id*, *source_id*, *fréquence*). « *Source_id* » représente d'où vient le tag en question et peut avoir une des valeurs suivantes : *Mures*, *Last.fm* ou *MusicBrainz*. Après que ces tables soient créées, nous avons créé la table ***activity_tag*** (*tag_id*, *activity_id*, *fréquence*). Celle-ci a été créée manuellement, en choisissant les tags les plus fréquents à partir des deux tables et nous avons essayé

d'assigner à chacune des activités disponibles à travers *Mures* (soirée romantique, conduite automobile, etc) le plus de tags possibles (voir Figure 4.8).

			activity_id	tag_id	frequence
<input type="checkbox"/>			driving	electronic	239
<input type="checkbox"/>			driving	electronica	107
<input type="checkbox"/>			driving	Energetic	39
<input type="checkbox"/>			driving	Bruce Springsteen	30
<input type="checkbox"/>			driving	CCR	30
<input type="checkbox"/>			driving	driving	29
<input type="checkbox"/>			driving	energy	17
<input type="checkbox"/>			driving	vangelis	11
<input type="checkbox"/>			driving	feelgood	10
<input type="checkbox"/>			driving	night	8
<input type="checkbox"/>			driving	street at night	7
<input type="checkbox"/>			driving	on the road	6
<input type="checkbox"/>			driving	late night	5
<input type="checkbox"/>			driving	car songs	4
<input type="checkbox"/>			driving	driver picks the music	4
<input type="checkbox"/>			driving	highway	3
<input type="checkbox"/>			driving	drive	3
<input type="checkbox"/>			driving	running	3

Figure 4.8 La table activity_tag dans Mures

Dans le cas de filtrage contextuel, nous parlons d'une simple recherche. L'utilisateur spécifie le contexte d'utilisation et le système retourne une liste de tags prédéfinis (voir Figure 4.9). De plus, l'utilisateur a l'option d'ajouter ses propres tags pour enrichir la recommandation. Le système envoie alors des requêtes vers la base de données de musique pour retrouver les chansons correspondantes.

Pour la classification des genres, de nombreuses approches ont été développées avec succès par la communauté MIR³². Certaines propositions intéressantes ont été comparées à la compétition MIREX³³ faites en 2011 (URL, 17).

³² Music information retrieval (MIR).

³³ Music Information Retrieval Evaluation EXchange (MIREX)

Les genres musicaux ont été organisés hiérarchiquement, en genres, sous-genres et feuilles. Nous avons créé un graph de genres. Le graphe sert à représenter les distances entre les différents genres musicaux afin de pouvoir évaluer les chansons sélectionnés lors de la génération des listes de lecture (voir Figure 4.10).



Figure 4.9 La liste de tags pour un contexte d'utilisation dans Mures

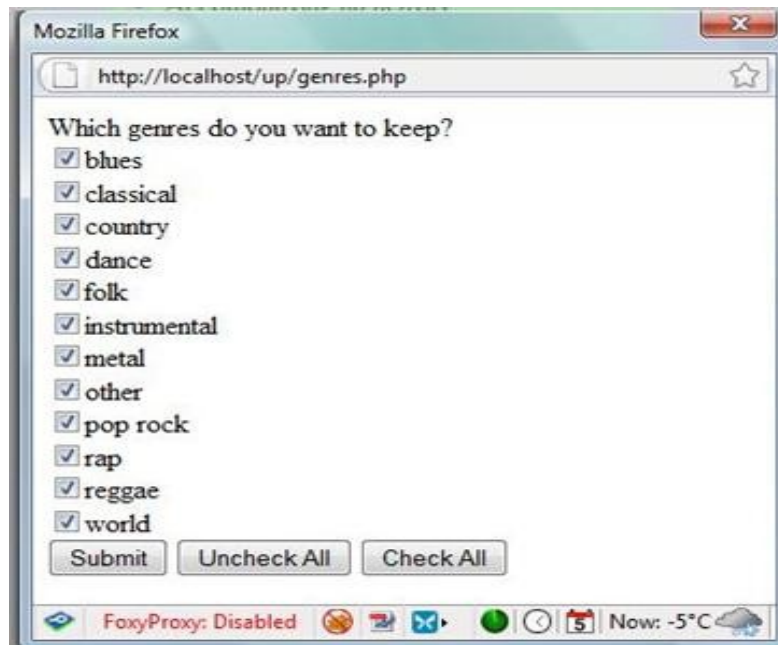


Figure 4.10 La liste de genres musicaux disponibles dans Mures

Lors de la création de la base de données *Mures*, nous avons aussi créé (en utilisant le même principe que celui utilisé pour les tags) deux tables pour les évaluations des artistes et des chansons : *votes_artistes* et respectivement *votes_chanson* :

- ***Votes_artistes*** (artist_id, vote_Mures, no_vote_Mures, vote_MB, no_vote_MB, vote_LF, no_vote_LF)
- ***Votes_chanson*** (chanson_id, vote_Mures, no_vote_Mures, vote_MB, no_vote_MB, vote_LF, no_vote_LF).

Afin d'évaluer l'artiste ou de la chanson « X », *Mures* calculera l'évaluation moyenne selon la moyenne pondérée suivante :

$$V_m = \frac{vote_{Mures} * no_vote_{Mures} + vote_{MB} * no_vote_{MB} + vote_{LF} * no_vote_{LF}}{no_vote_{Mures} + no_vote_{MB} + no_vote_{LF}}$$

V_m : vote pour l'artiste ou chanson X

$vote_{Mures}$: moyenne des votes dans Mures (MusicBrainz ou Last.fm)

no_vote_{Mures} : numéro d'évaluations dans Mures (MusicBrainz ou Last.fm)

Mures offre aussi la possibilité de sélectionner les artistes préférés (de l'utilisateur), selon les données comprises dans la table ***user_chanson_votes*** (*user_id*, *artist_id*, *chanson_id*, *vote*, *playcount*) (voir Figure 4.11).

user_id	artist_id	chanson_id	vote	playcount
testfoaf	Patricia Kaas	Le Mot De Passe	5	11
mirceaStrat	The Beatles	Yesterday	4	7
iuli76	Madonna	Frozen	4	10
dani	Celine Dion	My Heart Will Go On	5	16
testfoaf	Rob Zombie	More Human Than Human	5	2
iuli76	Patricia Kaas	Elle Voulait Jouer Cabaret	4	17
dani	NIRVANA	Smells Like Teen Spirit	5	1
marin	50 Cent	So Amazing (Feat. Olivia)	3	2
allsystem	Elton John	Blue Eyes	4	18
rolly	Iron Maiden	Fear Is The Key	5	13
allsystem	Nightwish	The Phantom of the Opera	4	1

Figure 4.11 La table *user_chanson_votes* dans *Mures*

Ainsi, les artistes les plus écoutés (les sommes des « *playcount* » les plus élevées - voir Figure 4.12) sont considérés *les préférés* et les chansons ayant les meilleures évaluations vont être considérées *les meilleures chansons* de cet artiste (voir Figure 4.13).

user_id	artist_id	playcount
testfoaf	Cher	187
testfoaf	Iron Maiden	151
testfoaf	Creedence Clearwater Revival	145
testfoaf	Shania Twain	137
testfoaf	Mihai Margineanu	122
testfoaf	Ozzy Osbourne	117
testfoaf	Bruce Dickinson	115
testfoaf	Rob Zombie	111
testfoaf	Judas Priest	102
testfoaf	Deep Purple	101
testfoaf	The Jackson 5	94
testfoaf	Patricia Kaas	85
testfoaf	Dire Straits	72
testfoaf	Michael Jackson	60
testfoaf	Led Zeppelin	13

Figure 4.12 Les artistes préférés de l'utilisateur testfoaf

artist_id	chanson_id	vote
Cher	Believe	4.88369652059273
Cher	After All (Love Theme From Chances Are)	4.81168653425918
Cher	Half-Breed	4.78916940632199
Cher	Love And Understanding	4.76965283767288
Cher	Dark Lady	4.75851856894653
Cher	If I Could Turn Back Time	4.68694798991731
Cher	Love Is The Groove	4.67236164274845
Cher	Strong Enough	4.6551512541763
Cher	All Or Nothing	4.52198631551302
Cher	Gypsys, Tramps And Thieves	4.34591519399166
Cher	Taxi Taxi	4.31951921742365
Cher	I Found Someone	3.73914693457926
Cher	Heart Of Stone	2.35186283323156

Figure 4.13 Les meilleures chansons de l'artiste Cher

La prochaine section explique le fonctionnement du filtrage contextuel dans Mures :

❖ Pseudo-code de l'algorithme de filtrage contextuel

Début

1) *L'utilisateur choisit le contexte d'utilisation : **Mures** propose une liste de tags et laisse la possibilité de valider ces tags et/ou des choisir d'autres tags $t_i \in T$*

2) *L'utilisateur choisit les genres musicaux $g_i \in G$*

3) *Pour chaque tag t_i sélectionné **Mures** va générer une liste de 10 meilleures chansons*

Si *l'utilisateur a spécifié des genres musicaux*

o *Garde seulement les chansons qui appartiennent aux genres spécifiées*

Fin si

4) **Mures** *trouve les artistes pour chaque chanson sélectionnée dans l'étape 3)*

5) **Mures** :

o *Trouve les 15 meilleures chansons des artistes connus de l'utilisateur*

o *Trouve les 15 meilleures chansons des artistes inconnus de l'utilisateur*

o *Recommande les 30 chansons trouvées de façon aléatoire*

Fin

Il a été largement reconnu que la popularité d'un article (et dans notre cas la popularité d'un artiste) peut diminuer la satisfaction des utilisateurs, en fournissant des recommandations évidentes (Herlocker *et al.*, 2004; Konstan *et al.*, 2006.). Pourtant, il n'y a pas de recette pour fournir des recommandations utiles aux utilisateurs. Mais d'après nous et en tenant compte de travaux de Lamere *et al.*, (2007), nous pouvons prévoir au moins

quatre éléments clés qui devraient être pris en compte. Ce sont : *la nouveauté*, *la sérendipité*³⁴, *la familiarité* et *la pertinence*.

La tâche difficile, pour les deux types de filtrage, a été de trouver le bon niveau de familiarité, de la nouveauté et de la pertinence pour chaque utilisateur. Nous avons voulu offrir à l'utilisateur la possibilité d'être familiarisé avec certaines chansons recommandées, pour améliorer sa confiance dans notre système. En tenant compte de la popularité des artistes et des informations de profil utilisateur, nous avons fourni des recommandations personnalisées, pertinentes qui sont également nouvelles pour l'utilisateur. En outre, certaines chansons devraient être inconnues pour l'utilisateur afin de découvrir les chansons cachées dans le catalogue.

La deuxième option de notre système de recommandation hybride est *le filtrage collaboratif*. Le calcul de la similarité est une étape indispensable dans le filtrage collaboratif. Pour notre système il est très important de trouver les usagers similaires ainsi que les artistes similaires.

Une des originalités de notre application c'est l'utilisation et la combinaison, des plusieurs jeux de données et des services web disponibles gratuitement sur le Web, comme la base de données de musique de *MusicBrainz* et les habitudes d'écoute de *Last.fm*. Afin de trouver les artistes similaires, on utilise une table prédéfinie, calculée à partir du jeu de données de *MusicBrainz*. Nous avons sélectionné 13,323 artistes distincts et la table contient 262,859 relations. Pour chaque relation (*artiste, artiste similaire, similarité*) (Ex : (*Madonna, Celine Dion, 0.87*)) nous avons testé la similarité correspondante dans *Last.fm* et nous avons gardé la moyenne pondérée.

Pour trouver les usagers similaires, on utilise au début les données de *Last.fm* comme première étape. Ainsi, notre système n'aura aucun problème avec le démarrage à froid, ni du nouvel usager, ni du nouvel artiste qui est un des plus grands problèmes du filtrage collaboratif (voir la section 2.3).

³⁴ La sérendipité est le fait de réaliser une découverte inattendue grâce au hasard et à l'intelligence, au cours d'une recherche dirigée initialement vers un objet différent de cette découverte

4.5.2 L'échelle d'évaluation

Comme nous avons vu dans le paragraphe précédent, pour le début on utilise les similarités pré-calculées. Mais à partir du moment que notre système va avoir un nombre suffisamment d'évaluations, les similarités seront calculées à partir de données de *Mures*.

Nous avons choisi une échelle variant de 1 à 5 pour l'échelle d'évaluation. Le chiffre 1 étant la plus haute expression du refus (rejet) et le chiffre 5 est la plus haute appréciation (acceptation). Le choix d'un nombre impair permet un choix intermédiaire qui représente l'état neutre ou indécis. Une autre possibilité dans *Mures* est l'option d'une valeur binaire (aime/n'aime pas chanson/artiste). Dans ce cas là, on considère la valeur 5 pour la valeur binaire *aime* et 1 pour la valeur *n'aime pas*.

Si l'utilisateur ne donne pas un feedback explicite *Mures* donne la possibilité du feedback implicite, caractérisé par les habitudes d'écoute. Pour chaque usager, les « *nombre de lectures* » seront normalisés selon l'échelle discrète de 1 à 5.

Cependant, pour la similarité basée sur les items, la similarité vectorielle ne prend pas en considération les différences dans l'échelle d'évaluation entre différents utilisateurs. La similarité vectorielle ajustée se sert de l'estimation moyenne d'utilisateur de chaque paire évaluée, et fait face à la limitation de la similarité vectorielle (voir la section 2.3.1).

4.5.3 Approches pour choisir le type d'algorithme de filtrage collaboratif

Avec le filtrage collaboratif, il est impératif de trouver le voisinage de l'utilisateur cible auquel est destinée cette recommandation ou le voisinage pour chaque chanson. La qualité des recommandations est fortement liée à la qualité de ce voisinage. Comme nous avons vu dans le chapitre deux, concernant les systèmes de recommandation, le filtrage collaboratif heuristique peut être basé sur l'utilisateur ou sur un article.

Nous commencerons avec l'approche « *basé sur usager* » afin de la tester et la comparer avec l'approche « *basé sur item* ». Nous considérons deux phases : *le calcul de voisinage et la prédiction*.

Le filtrage collaboratif à base de mémoire ou heuristique considère la totalité des évaluations des usagers disponibles au moment du calcul de la recommandation. *Pour les*

approches à base d'utilisateur (Resnick *et al.*, 1994; Maes & Shardanand, 1995), la prédiction d'un utilisateur évaluant un article est fondée sur les estimations, sur cet article, des plus proches voisins. Donc une mesure de similarité entre les utilisateurs doit être définie avant qu'un ensemble de voisins les plus proches ne soit pas choisi. Une méthode pour combiner les estimations de ces voisins sur l'article prévu doit être choisie.

La table suivante illustre cinq usagers qui évaluent quatre artistes, le résultat est une matrice de 20 éléments (évaluations) :

Table 4.1 Exemple d'une matrice d'évaluations classique Usagers x Items, où :

\bar{v}_i : la moyenne des votes de l'utilisateur i : la somme des votes de l'utilisateur i divisé par le total des artistes évalués,

* : l'utilisateur i n'a pas voté pour l'artiste j ,

? : la prédiction de vote de l'utilisateur Alice pour l'artiste Cher

Usager \ Artiste	Cher	Madonna	Shania Twain	Britney Spears	\bar{v}_i
Dany	5	4	4	4	4.25
Iuli76	4	5	4	*	4.33
Alice	?	4	5	3	4
Merrick	*	1	*	3	2
Bob	4	*	4	3	3.66

- Phase du calcul du voisinage

En se basant sur le profil d'utilisateur Alice (u_i), le système recherche les usagers u_j (*Dany*, *Iuli76* et *Bob*) qui lui sont les plus similaires. Nous avons choisi comme mesure de similarité, la *similarité vectorielle*. Empiriquement, la similarité entre deux usagers est calculée par la formule du *Cosinus* suivante :

$$sim(A, B) = \sum_{j=1}^n \frac{v_{A,j}}{\sqrt{\sum_{j=1}^n v_{A,j}^2}} \times \frac{v_{B,j}}{\sqrt{\sum_{j=1}^n v_{B,j}^2}}$$

n : nombre d'items communs entre A et B votés par v

$v_{A,j}$: vote de A pour l'item j

$v_{B,j}$: vote de B pour l'item j

Formule 4.1 : Le calcul du cosinus, réadapté de (Naak, 2009)

- Phase de prédiction

Une fois que les n usagers les plus similaires qui constituent le voisinage de *Alice* sont trouvés, la prédiction de la valeur d'un artiste j évaluée par l'utilisateur A (P_{aj}) est calculée à l'aide de la somme pondérée des estimations des voisins les plus proches qui ont déjà estimé pour l'artiste j .

Pour tenir compte de la différence dans l'utilisation de l'échelle d'estimation par les différents utilisateurs, les prédictions basées sur les déviations standard des estimations moyennes ont été proposées. P_{aj} peut être calculé à l'aide de la formule suivante :

$$P_{Aj} = \bar{v}_A + \frac{\sum_{i=1}^n sim(A, i) * (v_{i,j} - \bar{v}_i)}{\sum_{i=1}^n |sim(A, i)|}$$

n : nombre d'utilisateurs présents dans le voisinage de A , ayant déjà voté pour l'artiste j

$v_{i,j}$: vote de l'utilisateur i pour l'artiste j

\bar{v}_i : moyenne des votes de l'utilisateur i

Formule 4.2 : La formule de calcul de la prédiction, réadaptée de (Naak, 2009)

Table 4.2 Le voisinage de l'utilisateur Alice

Usager \ Artiste	Cher	Madonna	Shania Twain	Britney Spears	\bar{v}_i
Dany	5	4	4	4	4.25
Iuli76	4	5	4	*	4.33
Alice	4.26	4	5	3	4
Merrick	*	1	*	3	2
Bob	4	*	4	3	3.66

Si on trouve la similarité entre Alice et Dany 0.9, entre Alice et Iuli76 0.85 et entre Alice et Bob 0.8, alors la prédiction d'Alice pour l'artiste *Cher* sera :

$$P_{Alice,Cher} = 4 + \frac{0.9 * (5 - 4.25) + 0.85 * (4 - 4.33) + 0.8 * (4 - 3.66)}{0.9 + 0.85 + 0.8} = 4 + 0.26 = 4.26$$

La prochaine section explique le fonctionnement du filtrage collaboratif basé sur l'utilisateur, dans Mures :

❖ Pseudo-code de l'algorithme de filtrage collaboratif basé sur l'utilisateur

Début

- 1) **Mures** trouve les **N** usagers présents dans le voisinage de l'utilisateur **U**
- 2) **Mures** génère la liste de tous les artistes trouvés dans ce voisinage
- 3) **Mures** garde dans cette liste seulement les artistes inconnus de l'utilisateur **U**

Si l'utilisateur a spécifié des genres musicaux

- o **Mures** garde seulement les artistes qui appartiennent aux genres spécifiés

Fin si

- o **Mures** calcule la prédiction de chaque artiste en utilisant la formule (4.2)
- o **Mures** classe les artistes par ordre décroissant
- o **Mures** recommande les 30 premières chansons trouvées

Fin

Avec ce type de filtrage, l'utilisateur a la possibilité de découvrir des artistes inconnus, mais aimés par les utilisateurs similaires. Cet algorithme peut être renforcé avec l'option de garder dans la liste des artistes au moins 2-3 artistes faiblement connus par le système. Comme cela, l'utilisateur a toujours la chance de découvrir les jeunes artistes, qui n'ont pas la chance d'être évalués par beaucoup d'utilisateurs.

La méthode *basée sur les items* exploite la similarité parmi les artistes. Cette méthode examine l'ensemble d'artistes qu'un utilisateur a évalué, et calcule la similarité avec l'artiste cible.

- Phase du calcul de voisinage

La première étape est d'obtenir la similarité entre deux artistes, i et j . Nous avons choisi *la similarité vectorielle* comme mesure de similarité. Cependant, pour la similarité basée sur l'item, la similarité vectorielle ne prend pas en considération les différences dans l'échelle d'évaluation entre différents utilisateurs. *La similarité vectorielle ajustée* se sert de l'estimation moyenne d'utilisateur de chaque paire Co-évaluée, en palliant ainsi aux limitations de *la similarité vectorielle*. Empiriquement, la similarité entre deux artistes est calculée par la formule du Cosinus ajusté suivante :

$$sim(i, j) = \sum_{A=1}^m \frac{(v_{A,i} - \bar{v}_A)}{\sqrt{\sum_{A=1}^m (v_{A,i} - \bar{v}_A)^2}} \times \frac{(v_{A,j} - \bar{v}_A)}{\sqrt{\sum_{A=1}^m (v_{A,j} - \bar{v}_A)^2}}$$

m : nombre d'utilisateurs qui ont votés pour les 2 items
 $v_{A,i}$: vote de A pour l'item i
 $v_{A,j}$: vote de A pour l'item j
 \bar{v}_A : moyenne des votes de l'utilisateur A

Formule 4.3 : Le calcul du cosinus ajusté, adapté de (Celma, 2009)

- Phase de prédiction

Une fois que la similarité entre les articles calculée, la prochaine étape est de prévoir à l'utilisateur cible *Alice*, une valeur pour l'artiste actif *i*. Une manière commune est d'identifier comment l'utilisateur a évalué les articles similaires. La valeur prévue est basée sur la somme pondérée des estimations de l'utilisateur *Alice* ainsi que les déviations des estimations moyennes et peut être calculée à l'aide de la formule suivante :

$$P_{Ai} = \bar{v}_i + \frac{\sum_{j=1}^m sim(i,j) * (v_{A,j} - \bar{v}_j)}{\sum_{j=1}^m |sim(i,j)|}$$

m : nombre d'items présents dans le voisinage d'item i , ayant déjà voté par l'utilisateur A

$v_{A,j}$: vote de l'utilisateur A pour l'objet j

\bar{v}_j : moyenne des votes pour l'item j

$|sim(i,j)|$: similarité moyenne

Formule 4.4 : La formule de calcul de la prédiction, adaptée de (Celma, 2009)

Table 4.3 Le voisinage de l'artiste Cher où :

\bar{v}_j : la moyenne des votes pour l'artiste j ,

* : l'utilisateur i n'a pas voté pour l'artiste j

Usager \ Artiste	Cher	Madonna	Shania Twain	Britney Spears	\bar{v}_i
Dany	5	4	4	4	4.25
Iuli76	4	5	4	*	4.33
Alice	4.94	4	5	3	4
Merrick	*	1	*	3	2
Bob	4	*	4	3	3.66
\bar{v}_j	4.33	3.5	4.25	3.25	

Avec les données de table 4.3, dans le voisinage de Cher on trouve seulement Madonna et Shania Twain. Si on trouve la similarité entre Cher et Madonna 0.8 et entre Cher et Shania Twain 0.7, alors la prédiction d'Alice pour artiste Cher sera :

$$P_{Alice,Cher} = 4,33 + \frac{0,8 * (4 - 3,5) + 0,7 * (5 - 4,25)}{0,8 + 0,7} = 4,33 + 0,61 = 4,94$$

La prochaine section explique le fonctionnement du filtrage collaboratif basé sur l'utilisateur, dans Mures. Les artistes les plus écoutés (les sommes des « *playcount* » les plus élevées - voir Figure 4.12) sont considérés *les préférés* :

❖ Pseudo-code de l'algorithme de filtrage collaboratif basé sur les items

Début

- 1) **Mures** trouve les 10 artistes préférés de l'utilisateur **U**
- 2) **Mures** trouve le voisinage pour chaque artiste préféré :
au début on utilise une table prédéfinie, calculée à partir du jeu de données de MusicBrainz; après on calcule la similarité vectorielle ajustée (formule 4.3)

Si l'utilisateur a spécifié des genres musicaux

- o Garder seulement les artistes qui appartiennent aux genres spécifiés

Fin si

- o Calculer la prédiction de chaque artiste en utilisant la formule (4.4)
- o Classer les chansons des artistes par ordre décroissant
- o Recommander les 30 premières chansons trouvées avant

Fin

Avec ce type de filtrage, l'utilisateur a la possibilité de découvrir des artistes inconnus, mais similaires à ses artistes favoris. Pour déterminer les artistes favoris, *Mures* utilise les habitudes d'écoute de *Last.fm* et *Mures* ainsi que les votes de *MusicBrainz* et *Mures*. Cet algorithme peut être renforcé avec l'option de garder dans la liste des artistes au moins 2-3 artistes faiblement connus par le système. Avec l'option de choisir les genres musicaux,

Mures peut faire la distinction entre les artistes similaires par rapport à la popularité et les artistes similaires par rapport aux genres musicaux.

De plus, Sarwar (2001), présente l'évidence empirique que les algorithmes à base d'items peuvent fournir une meilleure performance de calcul que les méthodes traditionnelles basées sur l'utilisateur, en fournissant en même temps une qualité comparable ou meilleure que les meilleurs algorithmes disponibles, basés sur l'utilisateur.

4.6 Conclusion

Mures est une application Web de gestion et de recommandation de musique. *Mures* offre plusieurs fonctionnalités Web 2.0 comme les votes, les tags, le flux RSS tout en utilisant des techniques d'application Internet riches telles qu'AJAX pour améliorer l'expérience d'utilisateur.

Nous avons *relié quelques ensembles de données* disponibles gratuitement sur le Web (tels que la base de données de musique de *MusicBrainz*, les tags et les habitudes d'écoute de *Last.fm*, des informations liées à la musique - les nouveautés musicales, les prochains concerts de *Amazon* et les paroles de *LyricWiki*) afin d'obtenir des identificateurs uniques pour les chansons, les albums et les artistes. À notre connaissance, *ce travail constitue une première* puisque jusqu'au moment auquel nous avons commencé ce travail en 2010, chacune de ces applications ne prenait pas en compte qu'une partie des informations disponibles.

Dans ce chapitre, nous avons proposé un *algorithme hybride* de type *commutation* qui combine le *filtrage collaboratif* avec le *filtrage basé sur le contexte d'utilisation* afin de produire les meilleures recommandations. De plus, les recommandations peuvent être encore filtrées, en gardant seulement les genres musicaux spécifiées par l'utilisateur (*hybride* de type *cascade*).

L'objectif principal de notre système est de recommander à l'utilisateur de la musique à partir de sa collection personnelle ou de la nouvelle musique à partir de la collection du système. Notre approche suppose que les préférences d'utilisateur changent selon le contexte d'utilisation. Par exemple, un utilisateur écoute un genre de musique en conduisant vers son travail, un autre type en voyageant avec la famille en vacances, un

autre pendant une soirée romantique, etc. De plus, si la sélection de chansons a été générée pour plus d'un usager (voyage en famille, fête) le système proposera des chansons en fonction des préférences de tous ces utilisateurs.

À cette étape, toutes les fonctionnalités de *Mures* ont été présentées. Le chapitre 5 présente les résultats de la validation globale de notre système, ainsi que les résultats de tests et comparaisons des différents types d'algorithmes utilisés. Tout cela après avoir présenté les aspects technologiques entourant l'implémentation de notre système.

Chapitre 5 Implémentation et validation

Ce chapitre présente les détails de l'implémentation et de la validation de *Mures*. Nous commençons par le type d'architecture implémentée, l'environnement de développement et les technologies utilisées.

Par la suite, nous présentons les stratégies d'évaluation suivies pour tester et comparer les différents algorithmes de filtrage collaboratif, ainsi que l'interprétation des résultats obtenus. Pour finir, nous présentons la validation de toute l'application grâce au feedback reçu des participants de notre sondage.

5.1 Implémentation de Mures

Mures est une application basée sur le Web (voir Figure 5.1). Elle suit une *architecture client-serveur 3-tiers* (URL, 19), qui vise à séparer très nettement les trois couches logicielles de notre application. Nous avons modélisé notre application comme un empilement de trois couches, dont le rôle est clairement défini :

- La **présentation** des données : correspond à la partie de l'application visible et interactive avec nos usagers. Elle peut être réalisée par une application graphique ou textuelle. Nous avons choisi que *l'interface utilisateur* soit représentée en *HTML* pour être exploitée par un navigateur web;
- Le **traitement métier** des données (*data access tier or middle tier*) : correspond à la partie fonctionnelle de l'application, celle qui implémente la « logique », et qui décrit les opérations que l'application opère sur les données en fonction des requêtes des utilisateurs, effectuées au travers de la couche de présentation. Les différentes règles de gestion et de contrôle du système sont mises en œuvre dans cette couche. Cette couche contient dans notre cas les quatre modules présentés dans le chapitre précédent : *le module de recommandation, le module de gestion de profils d'utilisateurs, le module de gestion de ressources musicales et le lecteur mp3*
- L'**accès** aux données persistantes : correspond aux serveurs de bases de données. Ici les informations sont stockées et récupérées. Ce niveau conserve les données

neutres et indépendantes des serveurs d'application. En donnant aux données leur propre couche on améliore l'extensibilité et la performance du système.

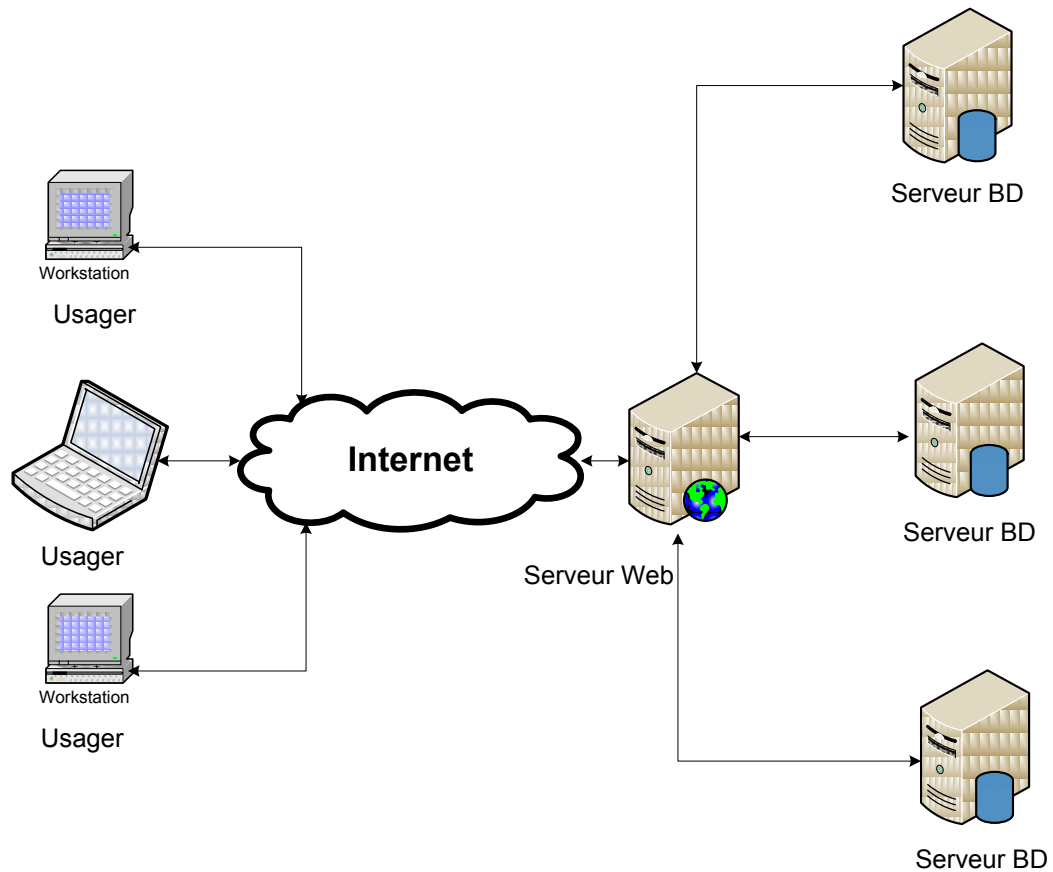


Figure 5.1 Architecture 3-tiers de Mures (URL, 19)

L'avantage principal d'une architecture multi-tiers est la facilité de déploiement. L'application en elle-même n'est déployée que sur la partie serveur (serveur Web). L'utilisateur ne nécessite qu'une installation et une configuration minimale. En effet il suffit d'installer un navigateur web pour que l'utilisateur puisse accéder à l'application. Cette facilité va permettre une évolution régulière du système. Cette évolution ne nécessitera que la mise à jour de l'application sur le serveur applicatif (et si nécessaire sur le serveur de base de données). Le

troisième avantage est l'amélioration de la sécurité. Dans un système client-serveur tous les clients accédaient à la base de données ce qui la rendait vulnérable. Avec une architecture multi-tiers l'accès à la base n'est effectué que par le serveur applicatif. Pour nous, c'est très important parce qu'on utilise un grand nombre de chansons et pour éviter des problèmes avec les droits d'auteur.

Physiquement, les données sont sur un serveur de données qui est géré par un *SGBDR* (Système de Gestion de Base de Données Relationnel) dans notre cas *MySQL* version 5.1.36. Nous avons normalisé les structures de données dans la base de données. Ceci pour les raisons suivantes : une structure normalisée se laisse idéalement implémenter au moyen d'un *SGBD* relationnel et exploiter au moyen d'outils existants, notamment le langage *SQL*. Les structures normalisées sont plus simples à comprendre et le développement de programmes est plus efficace avec les outils actuels.

L'application serveur est sur un serveur Web *Apache* version 2.2.11. L'application côté serveur Web est dynamique. Elle est implémentée avec le langage de script *PHP* version 5.2.11.

Par contre, le côté client est un client riche qui utilise la technologie *JavaScript* et *Ajax* pour divers avantages comme, la réduction du trafic, ce qui réalise des économies sur la bande passante, décharge les serveurs, et donne un meilleur temps de réponse. Nous avons essayé de respecter la séparation entre donnée, traitement et mise en forme en utilisant notamment les recommandations de *W3C*, telles que de valider les scripts *XHTML* et le *CSS*.

La prochaine section décrit les fonctionnalités de *Mures*.

5.2 Environnement d'utilisation

La figure 5.2 offre une vue générale des fonctionnalités de *Mures* :

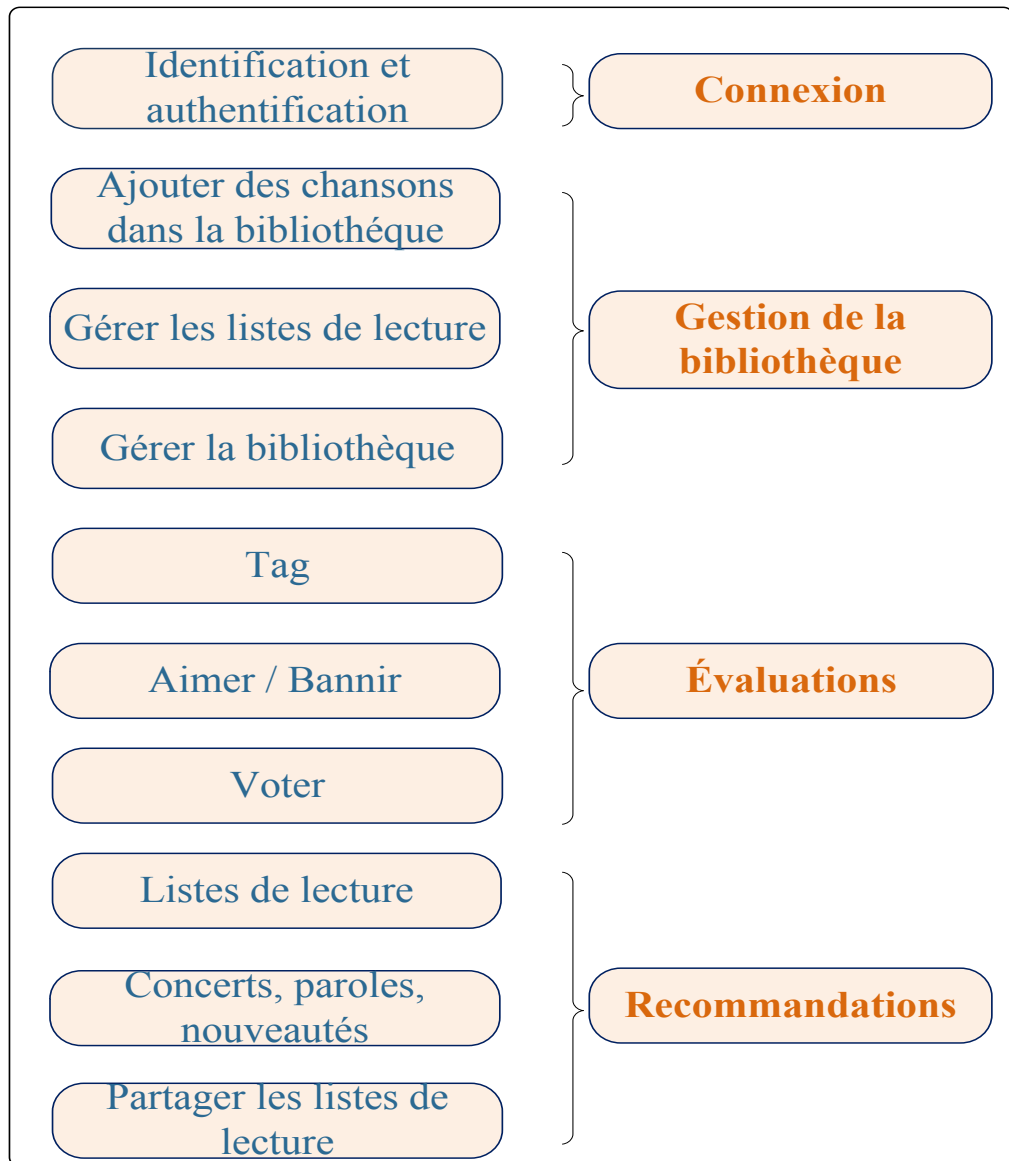


Figure 5.2 Les fonctionnalités de *Mures*

La section suivante décrit notre système *Mures*, étape par étape. Nous montrerons son fonctionnement global à travers quelques scénarii de fonctionnement et quelques captures d'écrans. Nous détaillerons chacun de ses modules dans ce qui suit.

La page d'accueil est la page publique de *Mures*. Cette page offre principalement trois choix pour accéder au système : ouverture de session, lien pour rappeler une session oubliée et un lien pour s'enregistrer. Si l'utilisateur possède déjà un compte, il peut directement s'identifier et s'authentifier avec respectivement son nom d'utilisateur et son mot de passe. En cas d'oubli, il peut demander au système de les lui envoyer vers son courriel utilisé lors de son enregistrement la première fois. Sinon, il doit s'enregistrer pour y accéder au système.

Dans ce qui suit, nous proposons 3 scénarii : un nouvel usager qui vient de se connecter au système pour la première fois, un usager qui demande une liste de chansons pour écouter en voiture, à partir de sa collection personnelle et finalement un usager qui veut découvrir des artistes similaires à ses artistes favoris.

a) Nouvel usager

- **Identification et authentification**

Pour utiliser notre application, la première étape consiste en création d'un compte utilisateur. Une fois fait, l'utilisateur est demandé de saisir quelques informations personnelles comme l'âge, la localisation, la langue ainsi que le pseudonyme de *Last.fm* et quelques préférences musicales. Toutes ces informations vont générer son profil d'utilisateur (voir Figure 5.3).

Bienvenu chez Mures...votre systeme de recommandation de musique



Nom	X
Prenom	Iuliana
Mures id	iuli76
Mot de passe	••••••
Confirm mot de passe	••••••
Location	Montreal
Artiste #1	Cher
Artiste #2	Madonna
Artiste #3	Bon Jovi
LastFm login	iuli76
	<input type="button" value="Enregistrement"/>

Figure 5.3 Enregistrement du nouvel usager dans Mures

- **Ajout et gestion de collectionne personnelle**

Le système offre à l'utilisateur un espace privé où il peut exploiter le système indépendamment des autres usagers. Après que le profil d'utilisateur soit créé, la deuxième étape est de rendre disponible sa collection musicale personnelle. Pour être capable de proposer des listes de lecture à partir de la collection personnelle d'utilisateur, notre système a besoin de savoir le chemin où se trouvent ses fichiers mp3. Cette étape est suivie de l'extraction de métadonnées de chaque fichier *mp3*.

Pour manipuler la collection personnelle plusieurs fonctionnalités de base sont disponibles : ajouter des fichiers mp3 dans la bibliothèque, modifier les fichiers existants ou bien effacer des fichiers (voir Figure 5.4).

Home User collections Recommendations User playlists Similar users Events Logout

Re-scanning a new directory will only add new, previously unscanned files into the list (and not erase the database).

Directory:

- Remove deleted or changed files from database
- List files with identical artist + title (same mix only)
- Tag Type Distribution
- Genre Distribution
- Blank genres
- Track "zero"
- Tags that are not synchronized (autofix)

Currently in the database:for user iuli76

Total Files	1,977
Total Filesize	10,879 MB
Total Playtime	130.6 hours
Average Filesize	5.5 MB
Average Playtime	3:58
Average Bitrate	193.0 kbps
Total Artists	149
Total Albums	149

Figure 5.4 Gestion de la collectionne personnelle

De plus, *Mures* met à la disposition de l'utilisateur plusieurs options pour optimiser la recherche dans sa propre collection musicale comme : offrir une liste de chansons avec le même nom et qui appartient au même artiste, offre la distribution de tags avec la possibilité d'exporter une liste de lecture pour chaque tag sélectionné et la distribution de genres

musicaux avec la possibilité d'exporter une liste de lecture pour chaque genre sélectionné, offrir une liste avec les chansons pour lesquelles il n'y a pas de métadonnées (genres, tags, etc.).

b) Usager qui demande une liste de chansons pour écouter en voiture

L'utilisateur demande une liste de chansons pour écouter en voiture, à partir de sa collection personnelle : le système utilise un algorithme de recommandation basé sur l'étiquetage social (voir Figure 5.5) :

- l'utilisateur choisit les tags (voir section 4.5.1);
- l'utilisateur choisit les genres musicaux (voir section 4.5.1);
- le système va générer une liste avec les meilleures chansons pour chaque tag sélectionné;
- *Mures* choisit de façon aléatoire N (=30) chansons

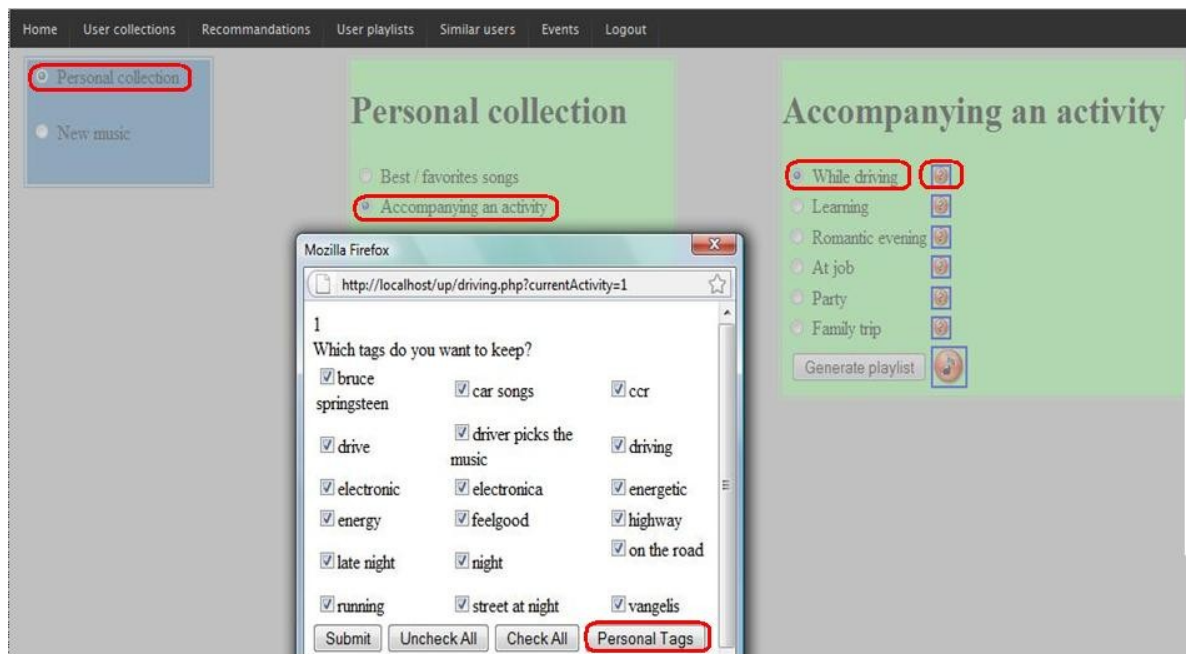


Figure 5.5 Liste de chansons pour écouter en voiture dans Mures

c) Usager qui veut découvrir la musique des artistes similaires aux artistes favoris

L'utilisateur demande une liste de chansons de ses artistes favoris. Le système va proposer 3 options: seulement les chansons des artistes favoris, seulement les chansons des artistes similaires ou une combinaison de deux avant (voir Figure 5.6).

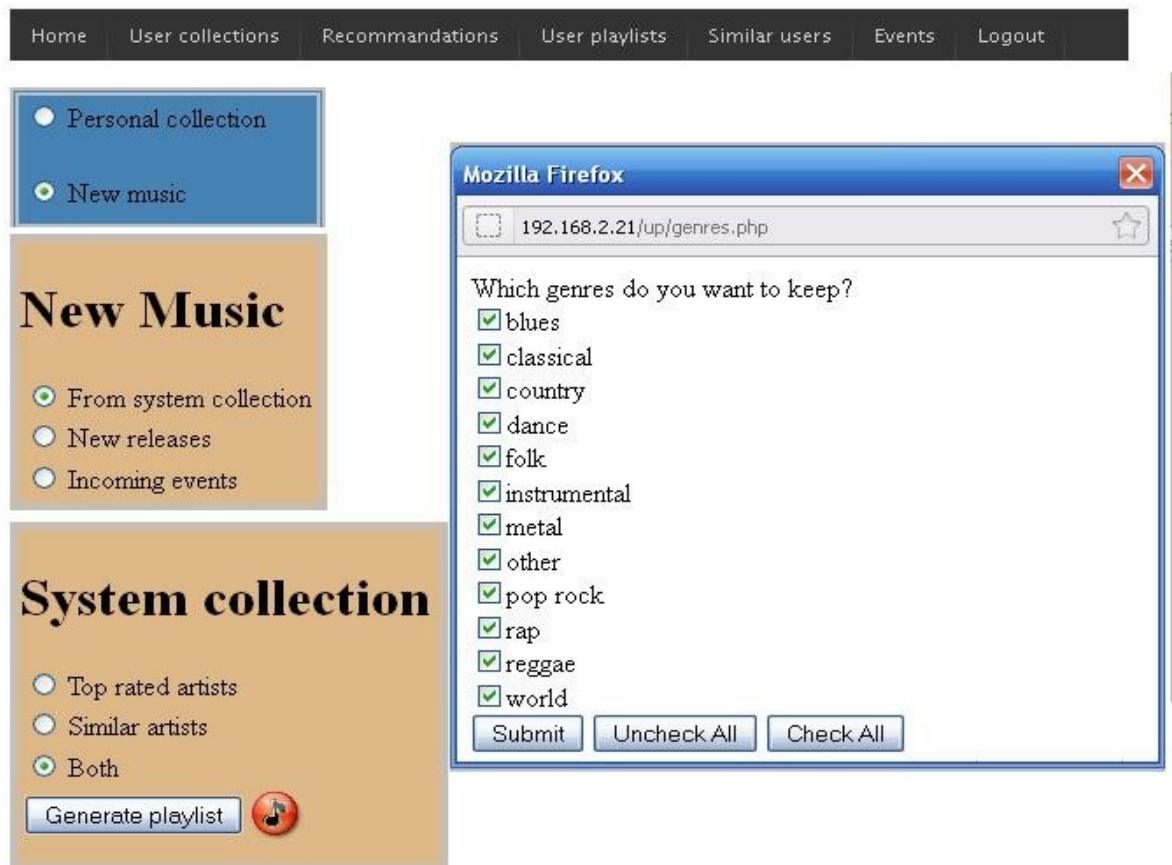


Figure 5.6 Liste de chansons des artistes similaires dans Mures

Dans notre scénario on considère la dernière option. Pour trouver les artistes favoris, *Mures* utilise les habitudes d'écoute de *Last.fm* et *Mures* (voir section 4.5.1). Pour déterminer les artistes similaires *Mures* utilise le filtrage collaboratif basé sur les artistes:

- le système va sélectionner les premiers 10 artistes favoris;

- le système cherche les artistes similaires pour chaque artiste favori : au début on utilise une table prédéfinie, calculée à partir de jeu de données de *MusicBrainz* et après on calcule « la similarité vectorielle ajustée » entre les artistes;
- le système choisit les premiers n artistes parmi les m plus similaires et on garde dans la liste des artistes au moins 2-3 artistes faiblement connus dans le système;
- le système va choisir les meilleures chansons des artistes sélectionnés.

- **Évaluation de chansons et artistes**

Le feedback d'utilisateur peut être explicite ou implicite. Dans notre système les estimations sont dans une échelle discrète de 0 à 5. Une autre possibilité est l'option d'une valeur binaire (aime/n'aime pas). Une autre manière pour recueillir le feedback explicite est que *Mures* permette aux utilisateurs d'écrire des commentaires et d'ajouter des tags. Notre système peut également recueillir le feedback implicite en surveillant les actions d'utilisateur, en analysant: le temps passé sur une page Web, les liens suivis de l'utilisateur, les habitudes d'écoute (dépistant le jeu, la pause, le saut et l'arrêt).

Le feedback implicite, obtenu de l'analyse des habitudes d'écoute sera transformé en feedback explicite selon l'algorithme suivant : lorsqu'une nouvelle chanson commence à être jouée, *Mures* incrémente automatiquement le compteur playcount dans la table *user_chanson_votes* (voir section 4.5.1). Si la chanson se termine sans l'aide des boutons *STOP* ou *NEXT*, le compteur playcount est validé et le champ vote de la table *user_chanson_votes* sera actualisé avec la moyenne des votes de l'artiste, en utilisant un arrondi vers l'entier le plus grand ($3.21 \Rightarrow 4$). Si l'artiste n'a pas encore été évalué par l'utilisateur, nous allons considérer la moyenne des votes de l'utilisateur, en utilisant un arrondi vers l'entier le plus grand.

Si un des boutons *STOP* ou *NEXT* est utilisé et que la chanson n'a pas été jouée au moins 80 % du temps, *Mures* va considérer que la chanson proposée n'a pas été aimée par l'utilisateur. *Mures* va attribuer l'évaluation « 1 » à la chanson si l'artiste n'a jamais été évalué

par l'utilisateur ou si l'évaluation moyenne de l'utilisateur pour l'artiste en question est inférieure à la moyenne générale de l'utilisateur (pour l'ensemble des artistes).

Lorsque l'utilisateur quitte l'application *Mures*, les tables *votes_artist* et *votes_chanson* (voir section 4.5.1) sont actualisées selon la table *user_chanson_votes*.

5.3 Comparaison avec les applications similaires

Pour bien montrer les caractéristiques de *Mures*, nous l'avons comparé avec les systèmes que nous avons mentionnés dans le chapitre 3 : *Genius*, *Last.fm*, *Pandora* et *PAPA*.

Table 5.1 Comparaison avec les applications similaires

	Type de filtrage	Recommande artistes	Recommande listes de lecture	Recommande voisinage d'utilisateurs	Gère la collection personnelle	Exporte .m3u ³⁵	Exporte sur supporte externe	Web 2.0	Gratuit
Genius	Hybride	x	x		x	x	x	x	
Last.fm	Hybride	x	x	x				x	
Pandora	FBC	x	x					x	x
PAPA	FCX		x			x	x		x
Mures	Hybride	x	x	x	x	x	x	x	x

Selon les observations faites à partir du tableau 5.1, *Mures* cadre bien avec le type de filtrage hybride. Les différences par rapport aux méthodes *Genius* et *Last.fm* sont

³⁵ **M3U** (MPEG version 3.0 URL) est un format de fichier qui a pour but de stocker une liste d'adresses, généralement, de fichier audio et/ou fichier vidéo. Créé à l'origine pour les listes de lectures de Winamp, ces fichiers sont de simples fichiers textes éditables à la main (ligne par ligne).

données aussi bien par le type d'algorithmes qui sont utilisés pour former le comportement hybride, que par la méthode de combinaison de ces comportements.

Du point de vue des options de recommandation, *Mures* et *Last.fm* sont les seuls qui offrent toutes les 3 possibilités : listes de lecture, artistes similaires et utilisateurs similaires.

Du point de vue de la portabilité et des possibilités de gestion de la collection personnelle, *Mures* et *Genius* sont les seuls qui offrent une gamme complète de fonctionnalités : export de liste de lecture vers le format standard *m3u*, export sur un support externe et la gestion de la collection personnelle. De ce point de vue, *Pandora* et *Last.fm* n'offrent aucun type de fonctionnalités.

La majorité des systèmes étudiés, à part *PAPA*, offre un GUI³⁶ avec des fonctions de type Web 2.0.

Mures, *PAPA* et *Pandora* sont des applications gratuites, avec la précision que le système *Pandora* n'est disponible qu'aux États Unis, et que *Last.fm* offre seulement 30 chansons gratuites (promotion limitée).

Ces comparaisons montrent la diversité des fonctionnalités de *Mures* afin de couvrir un large spectre d'aspects entourant la recommandation musicale. *Mures* se distingue des autres applications du genre par ses fonctionnalités, par son système de recommandation et par l'idée d'inter-corrélation entre les informations disponibles dans plusieurs bases de données ouvertes sur le Web. Comme mentionné dans la section 4.5.1, le filtrage hybride utilisé dans *Mures* est basé sur des nouvelles approches qui combinent le filtrage collaboratif avec le filtrage contextuel. Avant la validation globale de notre système, nous présenterons d'abord les tests de validation de celles-ci.

³⁶ GUI : une interface graphique (anglais *GUI* pour « *graphical user interface* »)

5.4 Stratégies d'évaluation

Cette section décrit en détail les procédures suivies pour tester les deux algorithmes de filtrage collaboratif préalablement introduits dans le chapitre 4. Nous classons l'évaluation d'algorithmes de recommandation en deux groupes: centrée sur le système et centrée sur l'utilisateur. L'évaluation des mesures centrées sur la précision du système permet de prédire les valeurs réelles que l'utilisateur a préalablement assignées. Cette approche a été largement utilisée dans le filtrage collaboratif, avec feedback explicite (par exemple les évaluations). L'évaluation des mesures centrées sur l'utilisateur se concentre sur la qualité perçue par l'utilisateur et l'utilité des recommandations. Les sections suivantes sont consacrées à expliquer chaque méthode d'évaluation.

5.4.1 L'évaluation des mesures centrées sur le système

En date d'aujourd'hui, l'évaluation des mesures centrées sur la précision du système a été largement étudiée. Les approches les plus courantes sont fondées sur la méthode *leave one out approach* (Girouard, Smith, & Slonim, 2006; Mitchell, 1997), qui ressemble à la validation croisée n fois classique. Étant donné un ensemble de données où l'utilisateur a explicitement ou implicitement interagi avec (via votes, achats, etc.), on divise les données en deux ensembles normalement disjoints: échantillon d'apprentissage et échantillon de test. L'évaluation de la précision est basée uniquement sur les données d'un utilisateur, afin que le reste des éléments soit ignoré. Autrement dit, cette méthode consiste à choisir aléatoirement une paire *Usager/Artiste* puis suppose que cet usager n'a pas encore évalué cet artiste pour essayer de prédire son évaluation.

Ces prédictions seront comparées par la suite avec les valeurs réelles omises en utilisant deux mesures : *MAE* (Mean Absolute Error) et *RMSE* (Root Mean Squared Error) qui sont des mesures de qualité de la prédiction très utilisées dans ce domaine. Les deux techniques de filtrage collaboratif sont testées équitablement sur les mêmes données et nous avons comparé et interprété les résultats obtenus.

MAE est une métrique qui est régulièrement utilisée pour évaluer la précision d'une prédiction. En bref, elle représente la différence moyenne entre la valeur originale et celle prédite.

$$MAE = \frac{1}{n} \sum_{i=1}^n |\hat{R}_i - R_i| \quad (5.1)$$

Où : n est le nombre de prédictions,

\hat{R}_i est la prédiction et

R_i est l'évaluation originale.

MSE (Mean Squared Error) est également utilisé pour comparer la valeur prédite avec la valeur réelle qu'un utilisateur a attribuée à un élément.

$$MSE = \frac{1}{n} \sum_{i=1}^n (\hat{R}_i - R_i)^2 \quad (5.2)$$

La différence entre MAE et MSE est que MSE souligne fortement les grandes erreurs. $RMSE$ est égal à la racine carrée de la valeur MSE .

$$RMSE = \sqrt{MSE} \quad (5.3)$$

$RMSE$ est une des métriques les plus utilisées dans le filtrage collaboratif basé sur les estimations explicites. $RMSE$ est la métrique utilisée lors du concours *Netflix 1000000* §³⁷.

Pour évaluer le système *Mures*, nous avons effectué une expérience impliquant les éléments suivants : l'hypothèse, l'échantillon et la procédure d'évaluation. Nous avons formulé l'hypothèse suivante :

Hypothèse : *Le filtrage collaboratif basé sur l'artiste offre une meilleure performance globale que le filtrage collaboratif basé sur l'utilisateur.*

³⁷ Le Prix Netflix est un concours ouvert en 2006 pour le meilleur algorithme de filtrage collaboratif pour prédire les notes des utilisateurs pour les films, en fonction des notes précédentes.

L'échantillon

L'échantillon, sur lequel sont effectués les tests, a été construit d'une manière aléatoire, suivant trois étapes. Premièrement, nous avons sélectionné de façon aléatoire un ensemble de 100 usagers différents (20 usagers à partir de la base de données de *Mures*, 80 à partir de la base de données de *Last.fm*). Pour chaque usager, nous avons sélectionné aléatoirement 50 évaluations des artistes différentes. Dans la deuxième étape, nous avons créé la matrice Usager x Artiste. Nous avons obtenu une matrice de 100 lignes sur 327 colonnes. La troisième étape a été de garder les artistes avec au moins 8 évaluations et les usagers avec au moins 8 évaluations pour éviter la *sparsity* de la matrice initiale. Nous avons décidé d'accorder la priorité aux usagers provenant de la base de données de *Mures*. Finalement, nous avons obtenu une matrice avec 20 usagers et 25 artistes, dont 500 paires usager/artiste.

La procédure d'évaluation

Pour tester les approches, nous avons divisé les données en deux ensembles : un échantillon d'apprentissage et un échantillon de test ; nous avons sélectionné aléatoirement un ensemble de test composé de 4 usagers, dont un maximum de 100 paires usager/artiste.

Pour la comparaison de deux approches, basée sur l'utilisateur et basée sur l'item, nous avons procédé comme suit : pour chaque pair usager/artiste nous faisons abstraction de cette évaluation en supposant qu'il ne l'a pas encore évalué et nous essayons de prédire ses évaluations. À la fin nous comparerons les valeurs des deux évaluations : celles originales avec celles prédites et cela en utilisant les deux mesures *MAE* et *RMSE* (voir Figure 5.7 et Figure 5.8).

La moyenne *MAE* et la moyenne *RMSE* à travers les 100 itérations sont utilisées pour comparer la performance des différentes approches implémentées. De plus, on répète le calcul pour 5 échantillons de test différents. Nous avons toujours accordé la priorité aux usagers provenant de la BD de *Mures*. La moyenne *MAE* et la moyenne *RMSE* de chaque échantillon de test sont enregistrées, ainsi que les moyennes à travers tous les échantillons.

Résultats

En général, l'approche *FC* basé sur l'utilisateur (voir section 4.5.3) est la moins performante, elle souffre principalement lorsque la similarité entre un usager et son

voisinage n'est pas suffisamment rapprochée. De l'autre côté, malgré que *FC* artiste n'a ni le plus bas *MIN MAE* ni le plus bas *MIN RMSE*, cette approche reste celle qui offre la meilleure performance globale.

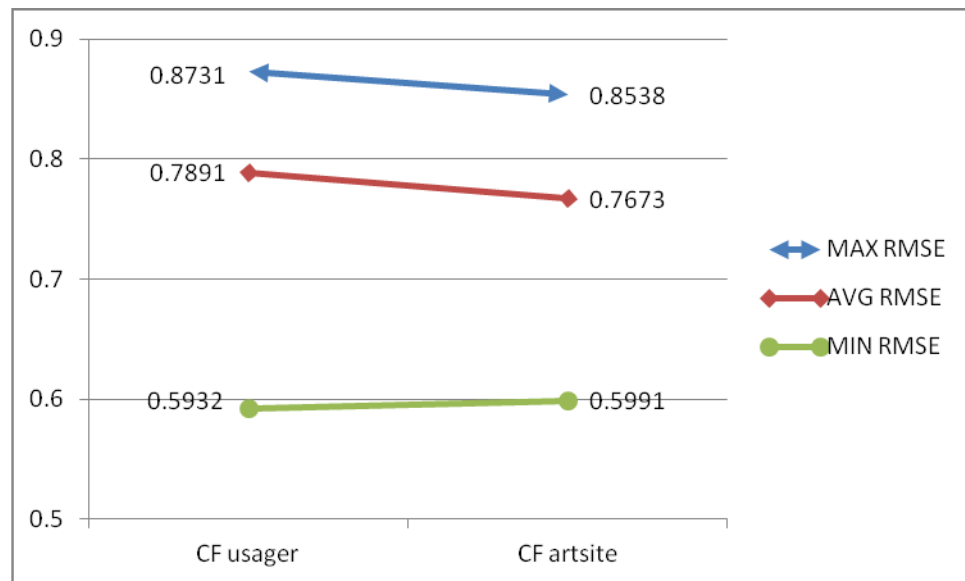


Figure 5.7 Comparaison de RMSE pour Mures

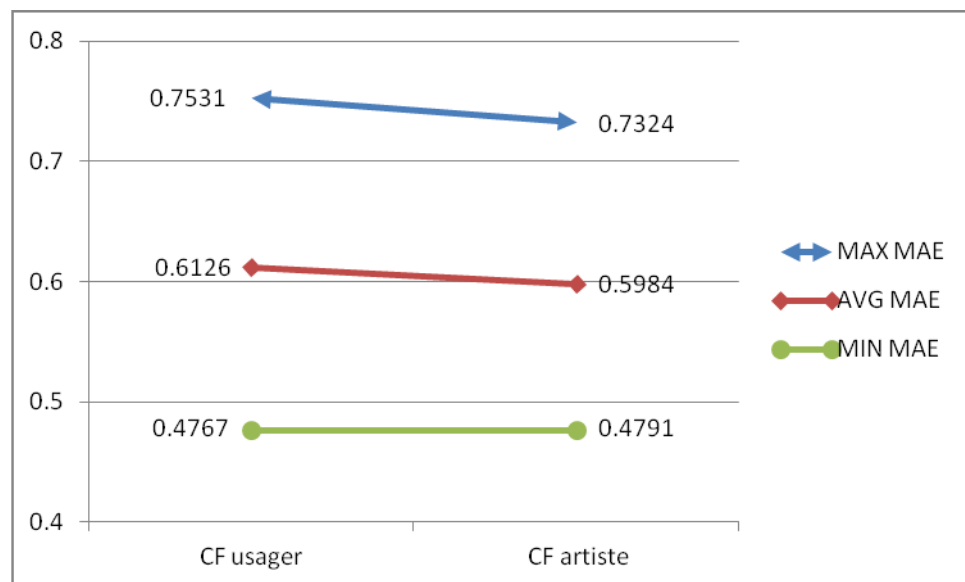


Figure 5.8 Comparaison de MAE pour Mures

Le table 5.2 montre pour l'utilisateur *Merrick* les valeurs des deux évaluations : celles originales avec celles prédites et cela en utilisant les deux mesures *MAE* et *RMSE* pour tous ses artistes.

Table 5.2 Exemple de calcul de MAE et RMSE pour un usager

Usager	Artiste	Evaluation	Prediction CF usager	RMSE	MAE	Prediction CF artiste	RMSE	MAE
Merrick	Ozzy Osbourne	4.9	4.69	0.0441	0.21	4.7	0.04	0.2
Merrick	Scorpions	4.9	4.67	0.0529	0.23	4.65	0.0625	0.25
Merrick	Queen	4.85	4.63	0.0484	0.22	4.61	0.0576	0.24
Merrick	Lady GaGa	4.6	4.21	0.1521	0.39	4.14	0.2116	0.46
Merrick	Bee Gees	4.45	4.17	0.0784	0.28	4.18	0.0729	0.27
Merrick	Celine Dion	4.15	4.01	0.0196	0.14	3.87	0.0784	0.28
Merrick	Nightwish	4.05	4.22	0.0289	0.17	4.13	0.0064	0.08
Merrick	Madonna	4	3.72	0.0784	0.28	3.77	0.0529	0.23
Merrick	Belinda Carlisle	4	3.84	0.0256	0.16	3.87	0.0169	0.13
Merrick	Shania Twain	3.95	3.52	0.1849	0.43	3.55	0.16	0.4
Merrick	Eminem	3.8	3.19	0.3721	0.61	3.31	0.2401	0.49
Merrick	Elton John	3.7	3.08	0.3844	0.62	3.12	0.3364	0.58
Merrick	Cher	3.6667	3.34	0.1067	0.326	3.26	0.165	0.407
Merrick	Zdob Si Zdub	3.55	2.88	0.4489	0.67	2.93	0.3844	0.62
Merrick	Iron Maiden	3.0909	4.13	1.0797	1.039	3.96	0.7553	0.869
Merrick	Theatre Of Trady	3.1	3.86	0.5776	0.76	3.75	0.4225	0.65
Merrick	Ada Milea	2.95	2.53	0.1764	0.42	2.49	0.2116	0.46
Merrick	Dire Straits	2.9	3.52	0.3844	0.62	3.42	0.2704	0.52
Merrick	Bon Jovi	2.85	3.99	1.2996	1.14	3.93	1.1664	1.08
Merrick	Black Sabbath	2.8	4.08	1.6384	1.28	4.1	1.69	1.3
Merrick	Enya	2.55	3.29	0.5476	0.74	3.32	0.5929	0.77
Merrick	Phil Collins	2.35	2.97	0.3844	0.62	2.88	0.2809	0.53
Merrick	Michael Jackson	2.25	3.45	1.44	1.2	3.31	1.1236	1.06
				0.6445	0.546		0.6043	0.516

Pour tester si la différence entre les deux méthodes (*FC* basé sur l'utilisateur et *FC* artiste) est significative statistiquement il faut faire un *Test t*³⁸ entre *RMSE* usager et *RMSE* artiste. Avant de procéder à un *Test t*, il faut formuler nos hypothèses statistiques (H_0 et H_1).

Hypothèse H_0 : *le filtrage collaboratif basé sur l'artiste offre la même performance globale que le filtrage collaboratif basé sur l'utilisateur.*

Hypothèse H_1 (l'hypothèse unilatérale) : *le filtrage collaboratif basé sur l'artiste offre une meilleure performance globale que le filtrage collaboratif basé sur l'utilisateur.*

Dans l'analyse d'un *Test t*, il y a 3 données importantes (voir Figure 5.9) :

- le résultat du test *t* (2.235 dans notre cas) ;
- le *df* ou degré de liberté, (ici 22) ;
- la valeur *p* du test ou *Sig. (bilatérale)*, dans ce cas-ci 0.036, mais parce que l'hypothèse est unilatérale on prend $Sig/2 = 0.018$.

→ T-Test

[DataSet2] C:\Documents and Settings\Roli\Desktop\mures_depot_08oct2012\21oct\rmse.sav

Paired Samples Statistics

	Mean	N	Std. Deviation	Std. Error Mean
Pair 1 RMSE CF usager	.4153722	23	.48534321	.10120106
RMSE CF artiste	.3651800	23	.43426797	.09055113

Paired Samples Correlations

	N	Correlation	Sig.
Pair 1 RMSE CF usager & RMSE CF artiste	23	.979	.000

Paired Samples Test

	Paired Differences					t	df	Sig. (2-tailed)
	Mean	Std. Deviation	Std. Error Mean	95% Confidence Interval of the Difference				
				Lower	Upper			
Pair 1 RMSE CF usager - RMSE CF artiste	.05019226	.10771637	.02246042	.00361221	.09677231	2.235	22	.036

Figure 5.9 L'analyse de données avec SPSS³⁹ (pour RMSE)

³⁸ Le Test *t*, ou test de Student désigne un ensemble de tests d'hypothèses paramétriques où la statistique calculée suit une loi de Student lorsque l'hypothèse nulle est vraie.

³⁹ SPSS (Statistical Package for the Social Sciences) est un logiciel utilisé pour l'analyse statistique.

Lors d'une valeur t calculée (2.235) plus grande que la valeur théorique t (pour le degré de liberté $df = 22$ avec une probabilité $p = 0.05$ (95 %) soit de 2.07) et lors d'une probabilité d'erreur $Sig = 0.018$ plus petite que la probabilité théorique $p = 0.05$, l'hypothèse H_0 est infirmée. On peut donc affirmer que « *le filtrage collaboratif basé sur l'artiste offre une meilleure performance globale que le filtrage collaboratif basé sur l'utilisateur* ».

En général, les tests de performances réalisés en utilisant des données générées en petits échantillons, n'est pas une méthode de validation rigoureuse. Cependant, nous considérons que cette procédure pour comparer les résultats de deux approches sur un même échantillon (même petit) de données offre une certaine validité à ces résultats. Avec ces tests nous sommes d'accord avec Sarwar (2001) : les algorithmes basés sur l'item peuvent fournir une meilleure performance de calcul que les méthodes traditionnelles basées sur l'utilisateur, en fournissant en même temps une qualité comparable ou meilleure que les meilleurs algorithmes disponibles, basés sur l'utilisateur.

5.4.2 L'évaluation des mesures centrées sur l'utilisateur

L'évaluation des mesures centrées sur l'utilisateur se concentre sur la qualité perçue par l'utilisateur et l'utilité des recommandations. Dans ce cas, l'évaluation nécessite l'intervention de l'utilisateur pour fournir un feedback pour les recommandations reçues. La principale différence avec une approche centrée sur la précision du système est que maintenant on augmente l'ensemble de données d'évaluation avec les items que l'utilisateur n'a pas encore vus (c.-à-d. évaluées, achetées, écoutées préalablement, etc.).

La principale limitation de l'approche centrée sur l'utilisateur est la nécessité d'intervention de l'utilisateur dans le processus d'évaluation. La collecte du feedback des utilisateurs peut être fastidieuse pour certains utilisateurs.

Pour la **validation globale** de système *Mures*, nous avons effectué une expérience impliquant les éléments suivants: les participants, le questionnaire et la procédure d'évaluation.

Les participants

Il y a 80 participants, y compris les 20 du département d'informatique de *l'Université de Montréal*. Tous les participants avaient déjà un compte *Last.fm*.

Le questionnaire

Nous avons divisé le processus de *validation globale* de *Mures* en deux étapes. Dans *la première étape*, nous avons demandé aux participants de remplir un questionnaire qui concerne leurs habitudes d'écoute. Afin de caractériser les participants, nous leur avons proposé quelques questions, incluant les informations démographiques de base (groupe d'âge et sexe), ainsi que les connaissances musicales des participants, le nombre moyen d'heures d'écoute par jour et le contexte d'écoute de la musique. Tous les champs sont facultatifs, afin que les participants puissent remplir ou non les informations (voir Figure 5.9).

En ce qui concerne le nombre moyen d'heures d'écoute par jour, l'enquête offre les options suivantes à choix unique :

- *presque jamais* ;
- *moins de 1 heure par jour* ;
- *entre 1 et 3 heures par jour* ;
- *plus de 3 heures par jour*.

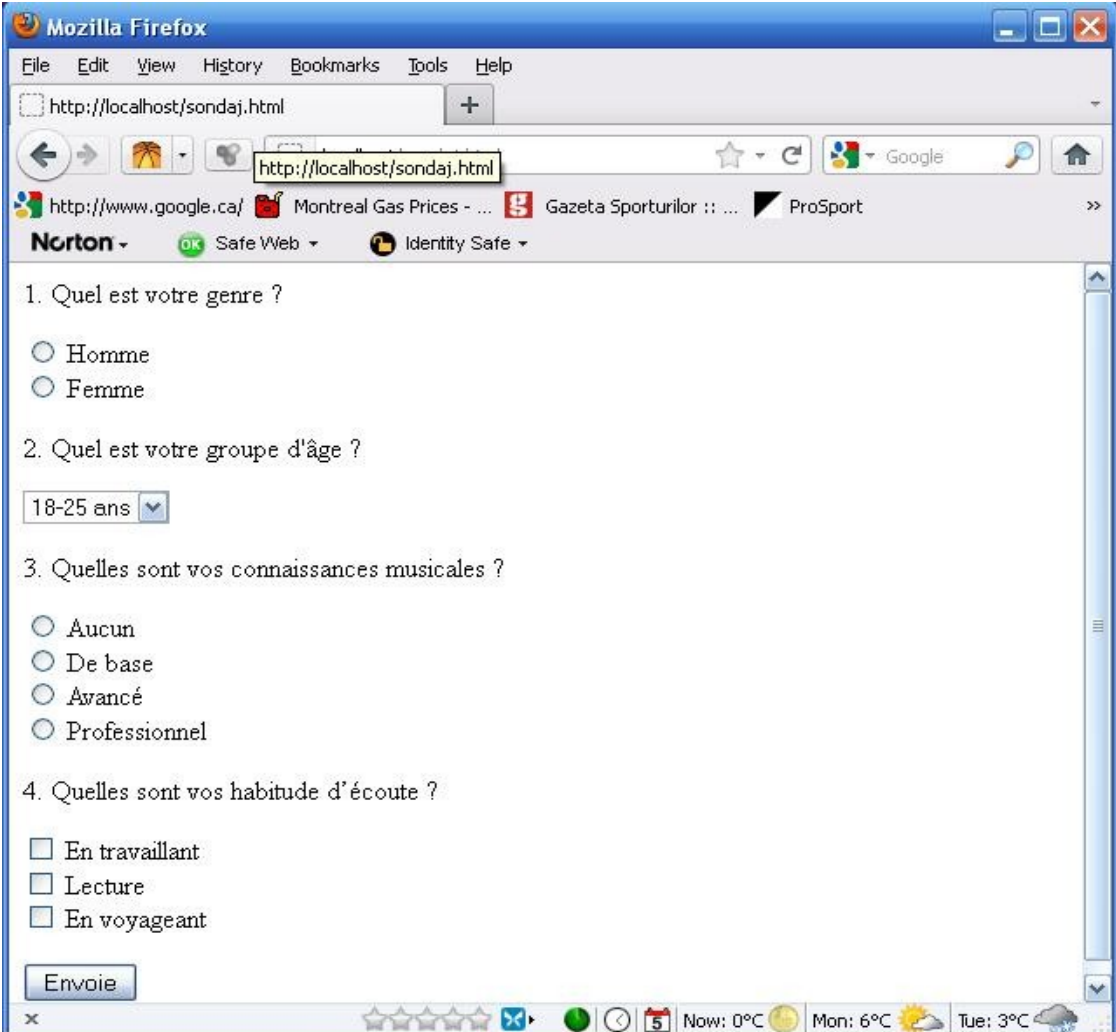
En ce qui concerne les connaissances musicales, l'enquête offre les options suivantes à choix unique :

- *aucun* : aucun intérêt particulier dans les sujets liés à la musique ;
- *de base* : leçons à l'école, la lecture des magazines de musique, blogs, etc. ;
- *avancé*: chant choral régulier, jouer à un instrument (amateur), mixer avec l'ordinateur, etc. ;
- *professionnel*: musicien professionnel, compositeur, étudiant au Conservatoire de musique, ingénieur du son, etc.

En ce qui concerne le contexte d'écoute, l'enquête offre les options suivantes à choix multiple :

- en travaillant ;

- lecture ;
- nettoyage ;
- en voyageant ;
- faire du sport ;
- cuisson ;
- je viens écouter de la musique (et ne fait rien d'autre) ;
- autres.



The screenshot shows a Mozilla Firefox browser window with the address bar set to `http://localhost/sondaj.html`. The page content is a survey form with the following questions and options:

1. Quel est votre genre ?
 - Homme
 - Femme
2. Quel est votre groupe d'âge ?
 - 18-25 ans
3. Quelles sont vos connaissances musicales ?
 - Aucun
 - De base
 - Avancé
 - Professionnel
4. Quelles sont vos habitudes d'écoute ?
 - En travaillant
 - Lecture
 - En voyageant

At the bottom of the form is a button labeled "Envoie". The browser's status bar at the bottom shows the current time as 5:00, a temperature of 0°C, and weather forecasts for Monday (6°C) and Tuesday (3°C).

Figure 5.10 Formulaire pour la validation globale de Mures

La procédure d'évaluation

Après avoir complété le questionnaire, les répondants ont eu l'opportunité de voir et d'utiliser les différentes fonctionnalités de *Mures*. Dans *la deuxième étape* les répondants donnent une évaluation générale pour le système par rapport aux recommandations fournies.

Les deux aspects les plus importants pour nous ont été les réponses aux questions concernant les deux contributions de ce mémoire :

(a) si *l'utilisation du contexte* dans les recommandations est importante pour les usagers. Pour chaque usager nous avons envoyé par courriel deux listes de lecture - une liste de 30 chansons générée avec notre algorithme de recommandation basé sur l'étiquetage social et une autre liste de 30 chansons générée sans l'aide de notre algorithme de recommandation basé sur l'étiquetage social. Nous avons demandé aux usagers leurs évaluations pour chaque chanson et nous avons écrit l'estimation moyenne de chaque utilisateur pour chaque liste de lecture dans le tableau 5.3.

Table 5.3 Le vote moyen de chaque utilisateur pour les recommandations reçues (contexte)

Usager	Vote sans contexte	Vote avec contexte
Allsystem	2.70	3.70
Dany	2.67	3.63
Iuli76	2.97	3.80
Merrick	2.87	3.70
Testfoaf	2.57	3.73
Rolly	2.63	3.67
Marin	2.93	3.57
MirceaStrat	2.97	4.03
Iuliana77	2.90	3.70
Aorolius	3.03	4.17
Antonio00ro	3.10	4.07
Alina73	2.97	3.83
Silviu78	2.77	3.87
Testfoaf2	2.53	3.63
Mahao77	2.70	3.67
Dea93	2.77	3.73
Emma85	2.73	3.77
Rolius	2.87	3.83
Jase808	2.77	3.67
Scarface86	2.93	3.77

Un effort particulier est fourni pour présenter à chacun des participants les objectifs et le but de cette validation. Nous avons gardé dans notre tableau les 20 premiers usagers ayant fait les plus d'évaluations. Pour tester si la différence entre les deux méthodes (une qui utilise l'algorithme de recommandation basé sur l'étiquetage social et une qui n'utilise pas) est significative statistiquement il faut faire un *Test t* entre les deux votes. Avant de procéder à un *Test t*, il faut formuler nos hypothèses statistiques (H_0 et H_1).

Hypothèse H_0 : *l'utilisation du contexte dans les recommandations n'est pas importante pour les usagers.*

Hypothèse H_1 (l'hypothèse unilatérale) : *l'utilisation du contexte dans les recommandations est importante pour les usagers.*

Dans l'analyse du *Test t*, on regarde les 3 données importantes (voir Figure 5.11) :

- le résultat du test *t* (32.712 dans notre cas) ;
- le *df* ou degré de liberté, (ici 19) ;
- la valeur *p* du test ou *Sig.(bilatérale)*, dans ce cas-ci 0.000, mais parce que l'hypothèse est unilatérale on prend $Sig/2 = 0.000$.

T-Test

[DataSet1] C:\Documents and Settings\Roli\Desktop\mures_depot_08oct2012\21oct\vf3.sav

Paired Samples Statistics

	Mean	N	Std. Deviation	Std. Error Mean
Pair 1 Context utilisé	3.7770	20	.15590	.03486
Pas de contexte utilisé	2.8190	20	.15814	.03536

Paired Samples Correlations

	N	Correlation	Sig.
Pair 1 Context utilisé & Pas de contexte utilisé	20	.652	.002

Paired Samples Test

		Paired Differences					t	df	Sig. (2-tailed)
		Mean	Std. Deviation	Std. Error Mean	95% Confidence Interval of the Difference				
					Lower	Upper			
Pair 1	Context utilisé - Pas de contexte utilisé	.95800	.13097	.02929	.89670	1.01930	32.712	19	.000

Figure 5.11 L'analyse de données avec SPSS (pour le contexte)

Lors d'une valeur t calculée (32.712) plus grande que la valeur théorique t (pour le degré de liberté $df = 19$ avec une probabilité $p = 0.05$ (95 %) soit de 2.09) et lors d'une probabilité d'erreur $Sig = 0.000$ plus petite que la probabilité théorique $p = 0.05$, l'hypothèse H_0 est infirmée. On peut donc affirmer que « *l'utilisation du contexte dans les recommandations est importante pour les usagers* ».

Les résultats de test de *Mures* sont très encourageants. Les résultats de l'évaluation prouvent que l'utilisateur a perçu que la qualité pour les recommandations qui n'utilisent pas l'algorithme de recommandation basé sur l'étiquetage social est faible (était de 2.819 sur l'échelle allant de 1 : *Très pauvre* à 5 : *Excellent*). Cela met l'accent sur le besoin d'ajouter plus de contexte en recommandant de la musique.

En même temps la qualité pour les recommandations qui utilisent l'algorithme de recommandation basé sur l'étiquetage social est très bonne (était de 3.777 sur une échelle allant de 1 : *Très pauvre* à 5 : *Excellent*).

(b) si les *recommandations reçues à partir de plusieurs jeux de données* ont été meilleurs que les recommandations reçues à partir d'un seul jeu de données (*MusicBrainz* et *Last.fm*). Pour chaque usager nous avons envoyé par courriel trois listes de lecture - une liste de 30 chansons générée avec les données disponibles de *MusicBrainz*, une autre liste de 30 chansons générée avec les données disponibles de *Last.fm* et une autre liste de 30 chansons générée à partir de la combinaison de plusieurs jeux de données. Nous avons demandé aux usagers leurs évaluations pour chaque chanson et nous avons écrit l'estimation moyenne de chaque utilisateur pour chaque liste de lecture dans le tableau 5.4. Un effort particulier est fourni pour présenter à chacun des participants les objectifs et le but de cette validation. Nous avons gardé dans notre tableau les 20 usagers ayant fait les plus d'évaluations.

Table 5.4 Le vote moyen pour les recommandations reçues (jeux de données)

Usager	Vote MusicBrainz	Vote Last.fm	Vote combinaison
Allsystem	3.23	3.30	3.73
Dany	3.37	3.40	3.77
Iuli76	3.47	3.50	3.93
Merrick	3.43	3.47	3.80
Testfoaf	3.20	3.20	3.83
Rolly	3.27	3.30	3.73
Marin	3.50	3.50	3.70
Mircea72	3.57	3.53	4.23
Iuliana77	3.40	3.43	3.83
Aorolius	3.63	3.73	4.13
Antonio00ro	3.77	3.73	4.03
Alina73	3.53	3.63	3.97
Silviu78	3.33	3.47	3.83
Testfoaf2	3.17	3.33	3.67
Mahao77	3.23	3.37	3.70
Dea93	3.40	3.43	3.93
Emma85	3.37	3.40	3.73
Sorin76	3.47	3.53	3.97
Jase808	3.33	3.37	3.77
Alex81	3.57	3.53	3.80

Pour tester si la différence entre les deux méthodes (une qui utilise un seul jeu de données – *MusicBrainz* ou *Last.fm* et une qui utilise la combinaison de plusieurs jeux de données) est significative statistiquement il faut faire un *Test t* entre les deux votes. Avant de procéder à un *Test t*, il faut formuler nos hypothèses statistiques (H_0 et H_1).

Hypothèse H_0 : *l'utilisation de plusieurs jeux de données n'est pas importante pour les usagers.*

Hypothèse H_1 (l'hypothèse unilatérale) : *l'utilisation de plusieurs jeux de données est importante pour les usagers.*

Dans l'analyse du *Test t*, on regarde les 3 données importantes (voir Figure 5.12) :

- le résultat du test *t* (16.920 pour la ressource *MusicBrainz* et 15.513 pour la ressource *Last.fm*) ;

- le df ou degré de liberté, (ici 19 pour les deux tests) ;
- la valeur p du test ou $Sig.(bilatérale)$, dans ce cas-ci 0.000 pour les deux tests, mais parce que l'hypothèse est unilatérale on prend $Sig/2 = 0.000$.

➔ **T-Test**

[DataSet1] C:\Documents and Settings\Roli\Desktop\mures_depot_08oct2012\21oct\vf3.sav

Paired Samples Statistics				
	Mean	N	Std. Deviation	Std. Error Mean
Pair 1	Les deux ressources	20	.15150	.03388
	Une ressource = MusicBrainz	20	.15585	.03485
Pair 2	Les deux ressources	20	.15150	.03388
	Une ressource = Last.fm	20	.13642	.03050

Paired Samples Correlations				
	N	Correlation	Sig.	
Pair 1	Les deux ressources & Une ressource = MusicBrainz	20	.711	.000
Pair 2	Les deux ressources & Une ressource = Last.fm	20	.689	.001

Paired Samples Test									
		Paired Differences				t	df	Sig. (2-tailed)	
		Mean	Std. Deviation	Std. Error Mean	95% Confidence Interval of the Difference				
					Lower				Upper
Pair 1	Les deux ressources - Une ressource = MusicBrainz	.44200	.11683	.02612	.38732	.49668	16.920	19	.000
Pair 2	Les deux ressources - Une ressource = Last.fm	.39650	.11431	.02556	.34300	.45000	15.513	19	.000

Figure 5.12 L'analyse de données avec SPSS (pour les jeux de données)

Lors d'une valeur t calculée (16.920 pour la ressource *MusicBrainz* et 15.513 pour la ressource *Last.fm*) plus grande que la valeur théorique t (pour le degré de liberté $df = 19$ avec une probabilité $p = 0.05$ (95 %) soit de 2.09) et lors d'une probabilité d'erreur $Sig = 0.000$ plus petite que la probabilité théorique $p = 0.05$, l'hypothèse H_0 est infirmée. On peut donc affirmer que « l'utilisation de plusieurs jeux de données est importante pour les usagers ».

Les résultats de test de *Mures* sont encore très encourageants. Les résultats de l'évaluation prouvent que l'utilisateur a perçu que la qualité pour les recommandations

faites à partir d'un seul jeu de données est bonne (était de 3.412 pour *MusicBrainz* et 3.457 pour *Last.fm* sur une échelle allant de 1 : *Très pauvre* à 5 : *Excellent*).

En même temps la qualité pour les recommandations faites à partir de la combinaison de plusieurs jeux de données est très bonne (était de 3.854 sur l'échelle allant de 1 : *Très pauvre* à 5 : *Excellent*). Cela met l'accent sur le besoin d'ajouter plus de ressources en recommandant de la musique inconnue.

Les résultats de la validation renforcent les hypothèses sur lesquelles Mures est basé (les deux contributions de ce mémoire) : « *l'utilisation de plusieurs jeux de données et le contexte sont très importante afin de fournir les meilleures recommandations musicales pour les usagers* ».

Conclusion

La problématique abordée dans ce mémoire est centrée autour du problème de la recommandation musicale avec toutes les informations qui lui sont reliées. La recherche en systèmes de recommandation est pluridisciplinaire. Elle comprend plusieurs domaines, tels que : la recherche et le filtrage, le forage de données, la personnalisation, les réseaux sociaux, le traitement de texte, l'interaction avec l'utilisateur et le traitement du signal. De plus, la recherche actuelle dans les systèmes de recommandation a un impact fort dans l'industrie, résultant en de nombreuses applications pratiques. Nous concluons que le problème de la recommandation musicale est à ce jour loin d'être résolu et suscite beaucoup d'intérêt autant dans le monde académique qu'industriel.

Plus précisément, dans le chapitre 3, nous avons passé en revue quatre classes de systèmes de recommandations de musique : les systèmes basés sur le filtrage collaboratif, les systèmes basés sur le filtrage à base de contenu, les systèmes basés sur le filtrage contextuel et finalement les systèmes hybrides. Nous avons constaté que ces solutions ne sont pas bien adaptées pour les besoins spécifiques de ce qui est l'objet du problème de la recommandation musicale.

Parce qu'il n'existe pas de solutions qui répondent à ces besoins et après avoir étudié et analysé les systèmes dans l'état de l'art (Chapitre 2 et 3), nous nous en sommes inspirés pour concevoir *Mures*, notre solution pour la recommandation musicale, en apportant nos améliorations. Dans le chapitre 4 nous avons étudié le problème de recommandation dans le sous-domaine de recommandation de musique. Nous avons commencé par une description générale de notre système, puis nous avons détaillé la structure de ses différents modules et fonctionnalités. Comme nous avons déjà mentionné et d'après la classification décrite dans le chapitre deux, notre approche de recommandation appartient à la classe des systèmes de recommandation hybride. Dans ce chapitre, nous avons proposé un *algorithme hybride* de type *commutation*, qui combine le *filtrage collaboratif* avec le *filtrage basé sur le contexte*

d'utilisation afin de produire les meilleures recommandations. On considère ça comme une contribution de ce mémoire.

Dans le chapitre 5 nous avons présenté les détails de l'implémentation et la validation de *Mures*. Nous avons commencé par l'architecture du système, l'environnement de développement et les technologies utilisées. Par la suite, nous avons présenté les stratégies d'évaluation suivies pour tester et comparer les différents algorithmes de filtrage collaboratif, ainsi que l'interprétation des résultats obtenus. Finalement, nous avons présenté la validation globale de *Mures*, grâce au feedback reçu des participants à notre sondage.

Comme nous avons vu, l'objectif principal de notre système est de recommander à l'utilisateur de musique à partir de sa collection personnelle ou de la nouvelle musique à partir de la collection du système. La musique se distingue d'autres domaines de divertissement, comme les films, ou les livres. Une différence évidente est qu'une personne peut écouter plusieurs fois la même chanson. Donc, une fonctionnalité importante de *Mures* est de créer de listes de lecture à partir de la collection personnelle d'utilisateur. Notre approche suppose que les préférences d'utilisateurs changent selon le contexte d'utilisation. Par exemple, un utilisateur écoute un genre de musique en conduisant vers son travail, un autre type en voyageant avec la famille en vacances, un autre pendant une soirée romantique. De plus, si la sélection de chansons a été générée pour plus d'un usager (voyage en famille, fête) le système proposera des chansons en fonction des préférences de tous ces utilisateurs.

De façon générale, les recommandations de notre system sont générées pour *accompagner un état d'esprit* comme heureux, triste ou romantique, pour *accompagner une activité* telle que fête, travail, conduite, en proposant les *chansons favoris* d'utilisateur, mélangées avec des *chansons neuves* susceptibles d'être aimées par l'utilisateur.

Une autre fonctionnalité de *Mures* c'est la recommandation des voisins. Un des principaux avantages de créer des voisinages est qu'un utilisateur peut explorer ses utilisateurs semblables, facilitant ainsi le processus de découverte de la musique. En outre, il permet la création des réseaux sociaux, de groupes, reliant les personnes qui partagent les mêmes intérêts.

On retrouve dans le système *Mures* les fonctionnalités de base pour la gestion de listes de lecture, par exemple : ajouter, supprimer une chanson, trier et exporter la liste de lecture, copier le contenu local sur un support externe : clé USB, cd, iPods et afficher toute sorte d'informations liées à chaque chanson jouée. *Mures* est une application Web de gestion et de recommandation de musique. *Mures* offre plusieurs fonctionnalités Web 2.0 comme les votes, les tags, le flux RSS toutes en utilisant des techniques d'application Internet riches telles qu'AJAX pour améliorer l'expérience d'utilisateur.

L'une des contributions de ce mémoire est que nous avons *lié quelques ensembles de données* disponibles gratuitement sur le Web comme la base de données de musique de *MusicBrainz*, les tags et les habitudes d'écoute de *Last.fm* et les informations liées à la musique comme les nouveautés musicales, les prochains concerts de *Amazon* et les paroles de *LyricWiki* afin d'obtenir des identificateurs uniques pour les chansons, les albums et les artistes. À notre connaissance, ce travail constitue une première, jusqu'à présent, les informations qu'une application gère ne bénéficie pas des informations qu'un autre peut avoir.

Dans ce mémoire, nous avons présenté les méthodes de recommandation pour un seul utilisateur. Un autre type de recommandeurs existe également (basé sur les groupes). Ces méthodes se concentrent à fournir des recommandations à un groupe d'utilisateurs, en essayant ainsi de maximiser la satisfaction générale du groupe.

Même si ce système est encore au stade de prototype, comme travail future on vise à ajouter la possibilité de construire des listes d'écoutes collaboratives, c'est-à-dire en considérant les goûts musicaux de plusieurs utilisateurs en même temps.

Bibliographie

- Abeles, H.F. (1980). Responses to music. *Handbook of music psychology*, Lawrence, KS: National Association of Music Therapy, 105-140.
- Adomavicius, G. & Tuzhilin, A. (2005), Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions, *IEEE Transactions. on Knowledge. and Data Engineering*. 17 (6), 734--749.
- Anderson, M., Ball, M., Boley, H., Greene, S., Howse, N., Lemire, D. & McGrath, S. (2003). Racofi: A rule-applying collaborative filtering system. *Proceedings of COLA'03. IEEE/WIC*.
- Barrington, L., Lanckriet, G.R.G. & Oda, R. (2009), Smarter than Genius? Human Evaluation of Music Recommender Systems. *Keiji Hirata; George Tzanetakis & Kazuyoshi Yoshii, ed., ISMIR , International Society for Music Information Retrieval*, 357-362.
- Barrington, L., Lanckriet, G.R.G., Torres, D. A. & Turnbull, D. (2008). Semantic Annotation and Retrieval of Music and Sound Effects. *IEEE Transactions on Audio, Speech & Language Processing* 16 (2), 467-476.
- Barrington, L., Lanckriet, G.R.G. & Turnbull, D. (2008). Five Approaches to Collecting Tags for Music. *Juan Pablo Bello; Elaine Chew & Douglas Turnbull, ed., 'ISMIR'* , 225-230.
- Bauer, F. & Martin, K. (2012). Linked Open Data:The Essentials. *Semantic Web Company, 2012*
- Breese, J.S., Heckerman, D. & Kadie, C. (1998). Empirical Analysis of Predictive Algorithms for Collaborative Filtering, *Proceedings of the 14th Annual Conference on Uncertainty in Artificial Intelligence* (pp. 43-52).
- Burke, R. (2002). Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370.
- Cai, J., Francis, J. & Gheysens, S. (2009). Creating a Hybrid Music Recommendation System from Content and Social-Based Algorithms. *Governor's School of Engineering and Technology Research Journal*.

- Candillier, L., Boullé, M. & Meyer, F. (2009). State-of-the-Art Recommender Systems
- Celma, O. *Music Recommendation and Discovery in the Long Tail*. PhD thesis, Universitat Pompeu Fabra, Barcelona, Spain, 2008.
- Celma, Ò., Laurier, C. & Sordo, M. (2007), Annotating Music Collections: How Content-Based Similarity Helps to Propagate Labels. *Simon Dixon; David Bainbridge & Rainer Typke*, ed., 'ISMIR', Austrian Computer Society, 531-534.
- Celma, Ò. & Ramírez, M. (2005). Foafing the music: A music recommendation system based on RSS feeds and user preferences. *6th International Conference on Music Information Retrieval (ISMIR)*. London, UK.
- Chai, W. & Vercoe, B. (2000). Using user models in music information retrieval systems. *1st Symposium on Music Information Retrieval*, Plymouth, Massachusetts, USA.
- Chen, Y.L., Cheng, L.C. & Chuang, C.N. (2008), A group recommendation system with consideration of interactions among group members, *Expert Syst. Appl.* 34 (3) , 2082-2090.
- Cho, S.B., Park, H.S. & Yoo, J.O. (2006), A Context-Aware Music Recommendation System Using Fuzzy Bayesian Networks with Utility Theory. *Lipo Wang; Licheng Jiao; Guangming Shi; Xue Li & Jing Liu*, ed., 'FSKD', Springer, 970-979.
- Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D. & Sartin, M. (1999). *Combining content-based and collaborative filters in an online newspaper*. Communication présentée Proceedings of the *ACM SIGIR '99 Workshop on Recommender Systems: Algorithms and Evaluation*.
- Coyle, L., McCarthy, K., McGinty, L., Nixon, P., Salamo, M. & Smyth, B. (2006), Group recommender systems: a critiquing based approach, *Proceedings of the 11th international conference on Intelligent user interface*, ACM Press, New York, NY, USA , 267--269.
- Cunningham, S.J. & Masoodian, M. (2006). More of an Art than a Science: Playlist and Mix Construction. *Proceedings of the International Conference on Music Information Retrieval '06 (ISMIR '06)*, Vancouver, Canada, pp. 233-239.
- Das, D., Hill, W., Stead, L. & Wittenburg, K. (1997) Group Asynchronous Browsing on the World Wide Web. *In proceedings of the 6th International Conference on the World Wide Web (WWW'6)*, 1997

- Dey, A.K. (2001). Understanding and Using Context, *Personal Ubiquitous Comput.* 5 (1) , 4-7.
- Dodds, P.S., Salganik, M. J. & Watts, D. J. (2006). Experimental study of inequality and unpredictability in an artificial cultural market. *Science*, 311(5762):854–856.
- Eck, D., Green, S., Lamere, P. & Mahieux, T.B. (2008), Automatic generation of social tags for music recommendation *Advances in Neural Information Processing Systems 20'* , MIT Press, Cambridge, MA.
- Eggen, J.H., Markopoulos, P., Perik, E. & Ruyter, B. (2004). The sensitivities of user profile information in music recommender systems. *Second Annual Conference on Privacy, Security and Trust: PST 2004*, 137-141.
- Eggen, B. & Pauws, S. (2002). PATS: Realization and user evaluation of an automatic playlist generator. *ISMIR*.
- Ehrlich, K. & Maltz, D. (1995). Pointing the way: active collaborative filtering. *Proceedings of the SIGCHI conference on Human factors in computing systems.* Denver, Colorado, United States, *ACM Press/Addison-Wesley Publishing Co.* 202-209.
- Ellis, D.P.W., Lamere, P., Mahieux, T.B. & Whitman, B. (2006). The million song dataset. *12th International Society for Music Information Retrieval Conference (ISMIR 2011)*
- Ellis, D.P.W. & Mandel, M. I. (2008). Multiple-Instance Learning for Music Information Retrieval. *Juan Pablo Bello; Elaine Chew & Douglas Turnbull*, ed., '*ISMIR*' , 577-582.
- Furnas, G., Hill, W., Rosenstein, M. & Stead, L. (1995) Recommending and evaluating choices in a virtual community of use. In *Proceedings of ACM CHI'95*, pages 194--201, 1995.
- Giasson, F. & Raimond, Y. *Music ontology specification*, working draft, February 2007
- Girouard, A., Slonim, D.K. & Smith, N.W. (2006), Motif Evaluation by Leave-one-out Scoring., in '*CIBCB*' , IEEE, , pp. 1-7 .
- Goldberg, D., Nichols, D., Oki, M.B., & Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Commun. ACM*, 35(12), 61-70.

- Hargreaves, D., Hargreaves, J. & North, A. (2004). Uses of Music in Everyday Life. *Music Perception: An Interdisciplinary Journal*, 22(1), 41-77.
- Herlocker, J.L., Konstan, J.A., Riedl, J.T. & Terveen, L.G. (2004). Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, 22(1), 5-53.
- Jennings, D. (2007). Net, Blogs and Rock 'n' Roll: How Digital Discovery Works and What it Means for Consumers. Nicholas Brealey Publishing.
- Konstan, J.A., Miller, B. & Riedl, J. (1997) *Experiences with GroupLens: Making Usenet useful again*, Proceedings of the 1997 Usenix Winter Technical Conference.
- Konstan, J.A., McNee S.M. & Riedl, J. (2006). Being accurate is not enough: how accuracy metrics have hurt recommender systems. *In Computer Human Interaction. Human factors in computing systems*, pages 1097–1101, New York, NY, USA. ACM.
- Konstan, J., Riedl, J. & Schafer, J.B. (1999). Recommender systems in e-commerce, *Proceedings of the 1st ACM conference on Electronic commerce*. Denver, Colorado, United States: ACM.
- Lamere, P. & Celma, O. (2007), Music Recommendation Tutorial *8th International Conference on Music Information Retrieval*, Vienna, Austria.
- Lee, J. & Lee, J. (2006). Music for My Mood: A Music Recommendation System Based on Context Reasoning. *Smart Sensing and Context*. P. Havinga, M. Lijding, N. Meratnia and M. Wegdam, Springer Berlin Heidelberg. 4272: 190-203.
- Levene, M. (2010). An Introduction to Search Engines and Web Navigation (2. ed.). Wiley.
- Liu, N., Szu-Wei, L. et al. (2009). Adaptive Music Recommendation Based on User Behavior in Time Slot." *International Journal of Computer Science and Network Security* 9, 219-227.
- Lopez, B., Montaner, M. & de la Rosa, J.L. (2003). A taxonomy of recommender agents on the internet. *Artificial Intelligence Review*, 19:285–330.
- Maes, P. & Shardanand, U. (1995). Social information filtering: algorithms for automating “word of mouth”, *Proceedings of the SIGCHI conference on Human factors in computing systems*. Denver, Colorado, United States: *ACM Press/Addison-Wesley Publishing Co.*
- Malone, T.W., Brobst, S.A., Cohen, S.A., Grant, K.R. & Turbak, F.A. (1987) Intelligent information des systèmes de partage. *Communications of the ACM*, 30 (5) :390-402, mai 1987.

- Margaritis, K.G. & Vozalis, E. (2003). Analysis of Recommender Systems' Algorithms. *6th Hellenic European Conference on Computer Mathematics & its Applications (HERCMA)*, Athens, Greece.
- Naak, A. (2009). Papyres : Un système de gestion et de recommandation d'articles de recherche. *Mémoire présenté à la Faculté des études supérieures en vue de l'obtention du grade de Maîtrise ès Sciences en Informatique*, Montréal, Canada
- Oliver, N. & Stickles, L.K. (2006), PAPA: Physiology and Purpose-Aware Automatic Playlist Generation. *ISMIR*, 250-253.
- Pachet, F. (2005). *Knowledge Management and Musical Metadata*. Idea Group.
- Pazzani, M.J. (1999). A Framework for Collaborative, Content-Based and Demographic Filtering. *Artif. Intell. Rev.*, 13(5-6), 393-408.
- Polanco M.J. & Rucker J. (1997). SiteSeer: personalized navigation for the Web, *Communications of the ACM*, 40 (3).
- Ramirez, J.M. (2005) Content-Based Music Recommender System. *Unpublished Master's Thesis*, Universitat Pompeu Fabra, Barcelona, Spain.
- Resnick, P. & Varian, H.R. (1997). Recommender systems. *Communications of the ACM*, 40(3):56–58.
- Rich, E. (1979). User modeling via stereotypes. *In Cognitive Science: A Multidisciplinary Journal*, volume Vol. 3, No. 4, pages 329–354.
- Richard, B. & Zsigmond, E.E. et al. (2011). Filtrage contextuel par cache pour application de réalité augmentée mobile. Lille, France: 181-196.
- Sarwar, B., Konstan, J., Reidl, J. & Karypis, G.(2001). Item-based collaborative filtering recommendation algorithms. *In WWW'01: Proceedings of 10th International Conference on World Wide Web*, pages 285–295.
- Terveen, L.G., Creter, J., Hill, W.C., McDonald, D. & Amento, B. (1997) *Building Task-Specific Interfaces to High Volume Conversational Data*, CHI'97.
- Uitdenbogerd, A. and van Schnydel, R. (2002). A review of factors affecting music recommender success. *In Proceedings of 3rd International Conference on Music Information Retrieval*, Paris, France.

Zhang, Y.C., Zhang, Z.K. & Zhou, T. (2011), Tag-Aware Recommender Systems: A State-of-the-Art Survey., *J. Comput. Sci. Technol.* 26 (5) , 767-777.

URLs

- URL, 1 : *Wikipedia*. Récupéré le Feb. de <http://fr.wikipedia.org/wiki/Musique>
- URL, 2 : *IFPI*. Récupéré le Feb. de <http://www.ifpi.org/content/library/DMR2010.pdf>
- URL, 3 : *IFPI*. Récupéré le Feb. de http://www.ifpi.org/content/section_resources/DMR2010.pdf
- URL, 4 : *Schemapedia*. Récupéré le Feb. de <http://schemapedia.com/schemas/foaf>
- URL, 5 : *Schemapedia*. Récupéré le Feb. de <http://schemapedia.com/examples/7f9aaf8161554f920ce493f88e355811>
- URL, 6 : *Mpeg.chiariglione.org*. Récupéré le Feb. de : <http://mpeg.chiariglione.org/standards/mpeg-7/mpeg-7.htm>
- URL, 7 : *Wikipedia*. Récupéré le Feb. de <http://fr.wikipedia.org/wiki/MPEG-7>
- URL, 8 : *Wikipedia*. Récupéré le Feb. de http://fr.wikipedia.org/wiki/XML_Schema
- URL, 9 : *Wikipedia*. Récupéré le Feb. de http://fr.wikipedia.org/wiki/Web_s%C3%A9mantique
- URL, 10 : *Jolomo*. Récupéré le Feb. de <http://jolomo.net/ringo/chi-95-paper.pdf>
- URL, 11 : *Swarthmore.edu*. Récupéré le Feb. de http://web.cs.swarthmore.edu/~turnbull/Papers/Tingle_Autotag_MIR10.pdf
- URL, 12 : *Wikipedia*. Récupéré le Feb. de <http://fr.wikipedia.org/wiki/iTunes>
- URL, 13 : *Skilledtest*. Récupéré le Feb. de http://www.skilledtests.com/wiki/Last.fm_statistics
- URL, 14 : *Wikipedia*. Récupéré le Feb. de <http://fr.wikipedia.org/wiki/Last.fm>
- URL, 15 : *Last.fm*. Récupéré le Feb. de <http://www.last.fm/community>
- URL, 16 : *ID3*. Récupéré le Juin de <http://www.id3.org/>
- URL, 17 : *MIREX*. Récupéré le Juin de <http://www.music-ir.org/evaluation/mirex-results/>
- URL, 18 : *Audioscrobbler*. Récupéré le Juin de <http://ws.audioscrobbler.com/2.0/?method=artist.gettoptracks&artist=cher>
- URL, 19 : *Wikipedia*. Récupéré le Juin de http://fr.wikipedia.org/wiki/Architecture_trois_tiers