Personnalisation de l'information
et gestion des profils utilisateurs

Data personalization and user profiles management

Dimitre Kostadinov

RÉSUMÉ

La personnalisation et la qualité de l'information constituent un enjeu majeur pour l'industrie informatique. Que ce soit dans le contexte des systèmes d'information d'entreprise, du commerce électronique, de l'accès au savoir et aux connaissances ou même des loisirs, la pertinence de l'information délivrée, son intelligibilité et son adaptation aux usages et préférences des clients constituent des facteurs clés du succès ou du rejet de ces systèmes. Dans ce rapport on fait l'état de l'art des approches de personnalisation existantes en matière de services de personnalisation, contenu des profils, techniques de construction et mise à jour des profils, utilisation des données du profil et sécurité des informations. Ensuite on propose un modèle de profil générique qui inclut tous les aspects de la personnalisation et qui montre les relations qui existent entre les différents composants. Ce modèle servira comme base pour le développement de nouveaux systèmes. Finalement on présente des axes de recherche à suivre pour aller des approches existantes locales vers un langage universel de personnalisation.

ABSTRACT

Data quality and data personnalization are two major challenges in modern information systems. In many contexts such as buiseness systems, e-commerce, knowledge management or any other service management, the information relevance as well as its interpretability and its adaptability to clients' usages and preferences are the key elements for the success or the reject of these informatioon systems. This report is a survey of the main ideas developed for data personnalization, user profiles definition and management and the quality related quality factors. This survey is completed by a proposal of a generic metamodel for data personnalization. This metamodel is defined by a set of dimensions which capture the main profile features synthesized from ther litterature.

1. Introduction

La personnalisation et la qualité de l'information constituent un enjeu majeur pour l'industrie informatique. Que ce soit dans le contexte des systèmes d'information d'entreprise, du commerce électronique, de l'accès au savoir et aux connaissances ou même des loisirs, la pertinence de l'information délivrée, son intelligibilité et son adaptation aux usages et préférences des clients constituent des facteurs clés du succès ou du rejet de ces systèmes. La production massive de nouvelles ressources, conjuguée à la nature fortement hétérogène, multiforme et multilingue de l'information constituent des verrous autant technologiques que sémantiques que les systèmes d'accès et de filtrage actuels doivent lever pour produire une information pertinente et de qualité. La personnalisation a pour objectif, d'une part, de faciliter l'expression du besoin utilisateur et de rechercher des informations sur un sujet en écartant l'information non-pertinente et donc réduire considérablement l'espace de recherche et, d'autre part, de rendre cette information sélectionnée intelligible à l'usager et exploitable. La pertinence de l'information n'est cependant pas une mesure objective, généralisable à tous les utilisateurs. Elle se définit par un ensemble de critères et de préférences personnalisables spécifiques à chaque utilisateur ou communauté d'utilisateurs.

Ils existent plusieurs approches de classification des préférences. Selon le type de données utilisées pour la comparaison de deux éléments on distingue des préférences **intrinsèques et extrinsèques**. Les préférences intrinsèques portent uniquement sur les valeurs des attributs d'un élément tandis que les préférences extrinsèques prennent en compte des facteurs extérieurs comme l'origine d'un élément. Une autre manière de classer les approches de personnalisation est basée sur la manière de comparer deux éléments. On retrouve ici **l'approche qualitative** où la préférence est spécifiée directement par des relations binaires de préférence et **l'approche quantitative** où les préférences sont exprimées indirectement par le biais de fonctions qui attribuent un nombre à chaque élément et la pertinence d'un élément dépend du nombre attribué. L'approche qualitative est plus générale que l'approche quantitative car elle peut définir les relations de préférences sur des fonctions numériques si elles sont données explicitement, tandis que toutes les relations de préférences ne peuvent pas être capturées par des fonctions numériques.

Du point de vue recherche, la personnalisation de l'information a été abordée principalement dans la communauté Recherche d'Information (RI) et la communauté Bases de Données (BD). Bien que très proches dans leur objectif final (accès efficace à l'information), les deux domaines de recherche se distinguent généralement par la nature de l'information traitée (documents textuels pour la RI et tableaux structurés pour les BDs) et le mode d'accès à cette information (accès par mots-clés plus ou moins complexes et pas à pas pour la RI, accès par expressions logiques et de façon globale pour les BDs). L'aspect interactif des requêtes RI a permis la prise en compte du retour de l'utilisateur (feedback) et l'affinage progressif de la recherche. L'aspect ensembliste des

requêtes BD a permis le développement d'optimiseurs traitant de très gros volumes de données. L'approche adaptative de la RI est une première étape vers la personnalisation de l'information. Les critères approximatifs et les critères de préférences introduits dans certains langages de requêtes BD ouvrent la voie à une adaptation du résultat aux désirs de l'utilisateur. Il est à remarquer que, dans les deux domaines, la personnalisation de l'information est rarement liée à la notion de qualité (fraîcheur des données, précision des données, crédibilité des sources, cohérence, complétude, sécurité, etc.).

Quelle que soit l'approche de personnalisation, on a toujours besoin de collecter et sauvegarder des données décrivant les utilisateurs sous forme de **profils**. Il convient de distinguer la notion de *profil* de la notion de *requête*. Un profil peut être défini comme une mise en équation du centre d'intérêt et des préférences de l'utilisateur, alors qu'une requête est l'expression d'un besoin circonstancié que l'utilisateur souhaite voir satisfait en tenant compte de son profil. Un profil a un caractère plus invariant que les requêtes même si le centre d'intérêt et les préférences de l'utilisateur peuvent légitimement évoluer. Un profil peut être modélisé directement à partir des besoins et des préférences des utilisateurs ou découvert par des méthodes de data mining à partir de leur comportement passé.

La contribution de ce rapport porte sur la proposition d'un modèle de profil générique. Ce modèle va permettre de structurer les données nécessaires pour décrire les préférences d'un utilisateur ce qui va nous donner une vision globale sur tous les problèmes liés à la personnalisation de l'information.

Le rapport est organisé en sept sections. Section 2 pose les problèmes traités dans l'état de l'art, section 3 contient l'état de l'art des approches de personnalisation existantes. Ensuite, dans la section 4 nous proposons un modèle de profil générique qui prend en compte tous les aspects de la personnalisation. Section 5 présente des exemple de profils. Dans section 6 nous donnons des axes de recherche à suivre et la conclusion du rapport.

2.Problématique de recherche

La prise en compte de la personnalisation de l'information lors de l'exécution des requêtes a engendré de nouveaux problèmes liés à l'hétérogénéité des applications et à la diversité des exigences des utilisateurs. L'état de l'art du rapport est organisé autour de cinq problèmes principaux :

1. Les services de personnalisation proposés actuellement par les systèmes
2. Le contenu des profils que les systèmes de personnalisation utilisent
3. Les techniques de construction et de mise à jour des profils
4. L'utilisation des données des profils dans l'évaluation des requêtes

5. La sécurité des données du profil

Le premier pas vers la construction d'un nouveau système capable de traiter les données en fonction des besoins et les préférences d'un utilisateur est le choix du service de personnalisation que ce système va fournir. Les services de personnalisation les plus couramment proposés sont la recommandation d'objets du résultat, le filtrage des résultats, le re-ordonnancement des éléments du résultat, l'enrichissement des requêtes. Généralement, la recommandation d'éléments est un service interactifs et peut répondre aux besoins instantanés d'un utilisateur. Le filtrage et le re-ordonnancement sont des processus de post traitement qui ont pour but d'adapter le contenu du résultat à un utilisateur donné, tandis que l'ajout des informations complémentaires aux requêtes a pour but le ciblage précis des informations intéressantes et peut se révéler dangereux dans le cas où les informations ajoutées ne seraient pas pertinentes.

Une fois le type de l'application choisi, le pas suivant consiste à identifier les paramètres dont le système a besoin afin de fournir un service personnalisé. Les informations qui caractérisent un utilisateur sont représentées dans un profil. Le contenu et la manière de représenter les données du profil d'un client varient selon l'application. Dans tous les cas le profil de l'utilisateur doit représenter la partie statique de son centre d'intérêt et doit rester relativement stable au cours du temps. Les intérêts occasionnels d'un utilisateur ne figurent pas dans son profil et sont pris en compte dans les systèmes de recommandation.

Un autre axe de recherche est le développement de techniques d'obtention et de mise à jour des informations du profil. Intuitivement la manière la plus simple d'initialiser un profil est la saisie manuelle des paramètres par l'utilisateur. Il est considéré que le client connaît mieux ses exigences et dans ce cadre il dispose d'une interface lui permettant de saisir les paramètres dont a besoin le système. Malgré la simplicité de l'approche les techniques actuelles vont vers une implication minimale du coté de l'utilisateur parce que le processus de saisie manuelle peut être long et bien souvent le client a des idées floues sur ses demandes et par conséquent a du mal à exprimer clairement ses intentions. Pour cette raison il est important que l'acquisition des informations du profil se fasse de façon automatique et avec le moins d'intervention de la part de l'utilisateur. Il s'agit à ce niveau de trouver des techniques d'obtention automatique des besoins de l'utilisateur en utilisant son retour (feedback) qui peut être implicite ou explicite, en se basant sur le passé du client ou sur l'expérience des autres utilisateurs ou en interagissant avec le client afin de l'aider à mieux définir et exprimer ces exigences. Une fois le profil de l'utilisateur initialisé, le système doit être capable de capturer les modifications des centres d'intérêts de l'utilisateur. L'évolution du profil est un processus délicat qui nécessite l'interprétation des actions de l'utilisateur. Dans certains cas il est possible que le système interprète mal les interactions de l'utilisateur. Ceci peut arriver dans le cas où le client agit de façon occasionnelle et par conséquent son profil ne doit pas être mis à jour parce que ce centre d'intérêt n'est pas caractéristique pour lui. Un autre problème peut survenir lors de l'interprétation des actions de l'utilisateur qui sont souvent liées et de ce fait doivent être prises en compte ensemble. Dans ce contexte le client doit avoir le contrôle sur le contenu du profil et doit être capable de le modifier à chaque instant. Lorsque l'utilisateur fait des mises à jour

manuellement le système doit être capable de corriger son fonctionnement pour faire évoluer le contenu du profil de façon adéquate.

L'utilisation des informations du profil constitue un autre défit qui nécessite l'évolution des systèmes de traitement des données actuels. Un premier problème consiste à intégrer les préférences de l'utilisateur qui concernent les résultats attendus dans les requêtes. La plupart des langages actuels ont une syntaxe fixe qui ne permet pas la prise en compte de la personnalisation dans l'écriture des requêtes. Pour cette raison des travaux ont été fait dans le but d'étendre les langages par de nouvelles clauses qui permettent d'exprimer les préférences de l'utilisateur sur les données du résultat. Le fait d'ajouter de nouvelles clauses aux langages introduit le besoin de développer des techniques d'évaluation de ces clauses. Comme dans certains cas le profil de l'utilisateur peut contenir des informations décrivant ses exigences sur la qualité du processus d'exécution des requêtes (par exemple le temps maximal de réponse), les techniques d'évaluation des clauses doivent tenir compte de ces informations.

Un autre problème majeur qui provient de la nature confidentielle des informations du profil est la sécurité des données. Il s'agit d'une part de protéger les informations du profil par des techniques de définition des droits d'accès, de crypter les données échangées et d'effectuer les transferts d'informations sur des liaisons sécurisées.

3.Etat de l'art

Dans cette partie du rapport, on va aborder les problèmes mentionnés dans la section 2 en donnant des exemples d'approches existantes et en expliquant leur fonctionnement. Ce stage a nécessité un état de l'art plus approfondi en raison de la diversité des approches existantes. La personnalisation est un domaine de recherche en plein développement, mais les problèmes liés à la personnalisation ne sont pas encore maîtrisés.

3.1 Les services de personnalisation

Avec le développement de l'Internet la quantité de l'information délivrée a augmentée ce qui a introduit le besoin d'adapter la manière de fournir les données aux besoins des utilisateurs. Pour cette raison, de plus en plus de systèmes d'information proposent des services adaptés afin de mieux cibler leurs clients. [Gauch 99] propose une cinquantaine d'exemples de systèmes de personnalisation en faisant une classification des approches existantes selon deux catégories :

- accès personnalisé aux ressources (portails personnalisés sur le Web ou systèmes de fichiers)
- filtrage et estimation (journaux électroniques, les systèmes de recherche d'informations et services de recommandation d'éléments incluant la navigation)

D'après les auteurs de l'article, l'idée principale de l'accès personnalisé aux données est de fournir à l'utilisateur une interface d'accès aux informations adaptée à ses besoins et ses centres

d'intérêts. Ceci peut être fait en proposant au client des liens à suivre vers des sources qui contiennent des informations sur les thèmes qui l'intéressent ou en lui affichant des pages dont le contenu dépend de ces préférences.

Des exemples de systèmes qui proposent un accès personnalisé aux données sont InfoQuest[1] et MyYahoo[2]. InfoQuest fournit des liens vers différentes sources d'information et permet la spécification des centres d'intérêts d'un utilisateur par une liste de mots clés. De son coté, MyYahoo permet aux utilisateurs de choisir le contenu de la page en cochant les catégories jugées intéressantes parmi une liste donnée (actualités, horoscopes, programmes télé, météo, cinéma, etc.). Une fois les catégories intéressantes choisies, le système offre à l'utilisateur une page personnelle qui contient des informations dont les thèmes sont les catégories et des liens vers des données supplémentaires.

L'accès personnalisé aux données est une technique qui est très peu décrite dans la littérature. Les auteurs de l'article n'en parlent que très peu. C'est la raison pour laquelle cette approche de personnalisation ne sera pas décrite en détails dans le rapport. Les services de personnalisation de la deuxième catégorie seront explicités plus loin dans le rapport.

Une autre classification des services de personnalisation est faite par [Pretschner 99]. Selon cet article, lorsqu'un utilisateur effectue une recherche d'information, les termes qu'il emploie dans sa requête sont ambigus et par conséquent le système ne connaît pas le sens que l'utilisateur veut leur donner. Pour résoudre ce problème il existe trois approches : le re-ordonnancement des éléments du résultat, le filtrage des résultats et l'extension des requêtes avec les intérêts du client. Dans l'article, les auteurs représentent le fonctionnement des deux premières approches (sections 3.1.1 et 3.1.2) en considérant que si lors de la réécriture des requêtes, des termes sont introduits et si ces termes ne correspondent pas aux intérêts de l'utilisateur, on risque de ne sélectionner que des documents non pertinents.

Le fait d'enrichir la requête de l'utilisateur avec les informations de son profil permet de cibler mieux les informations dont il a réellement besoin. D'un coté, le fait d'ajouter des prédicats aux requêtes permet une meilleure utilisation des capacités des serveurs qui ont souvent des moyens d'exécution optimisés, mais d'un autre coté, c'est un processus très délicat à cause du risque d'insertion d'informations incorrectes qui provoquerait le retour de résultats sans aucun intérêt. C'est la technique la moins utilisée actuellement parce qu'elle nécessite la traduction des préférences de l'utilisateur en un langage compréhensible par le système qui fournit les informations ce qui n'est pas toujours possible vu que très peu de ces systèmes prennent en charge et permettent l'expression de la personnalisation.

Un exemple de système qui modifie la requête de l'utilisateur donné par [Ferreira 01] est MyGlobalNews. La requête de l'utilisateur est remplacée par son profil et le système va calculer le

---

[1] *www.inforian.com/quest*

[2] *my.yahoo.com*

matching entre le profil et les documents. Seuls les documents dont le matching dépasse un certain seuil sont inclus dans le résultat. Le profil de l'utilisateur est exprimé sous la forme de mots clés ce qui permet son utilisation directe comme requête dans une approche RI.

Les services de personnalisation les plus couramment utilisés sont le filtrage du résultat, le re-ordonnancement des éléments du résultat et la recommandation d'éléments.

3.1.1 Le filtrage des résultats

Le principe de base de cette approche est d'exécuter de la requête sans prendre en compte la personnalisation et ensuite appliquer un post traitement sur le résultat afin d'éliminer les résultats non pertinents pour l'utilisateur. Le filtrage peut être fait soit en appliquant des requêtes supplémentaires sur le résultat, soit en traitant chaque élément séparément afin d'étudier sa pertinence.

Un exemple de système qui filtre les résultats une fois la requête exécutée est CASPER(Case-Based Profiling for Electronic Recruitment) ([Bradley 00]). C'est un système de recherche d'annonces de travail. La requête de l'utilisateur est évaluée en deux étapes : extraction d'annonces sur le serveur faite sur la base de similarité entre les termes et non sur un matching exact et filtrage des résultats non-pertinents sur le client. Chaque annonce contient un nombre fixe d'attributs. Lors de la comparaison entre les termes de la requête de l'utilisateur et les caractéristiques des annonces sur le serveur on ne cherche pas un matching exact, mais une similarité qui dépasse un certain seuil afin d'élargir l'espace de recherche. Les termes sont représentés dans des arbres ontologiques et la similarité entre deux termes est définie par la distance entre eux dans ces arbres. Ensuite, les résultats sont envoyés sur le client où on a un enregistrement des annonces qu'il a aimées et celles qu'il n'a pas appréciées. Grâce à ces enregistrements, on détermine si un élément est pertinent ou pas et seuls les annonces pertinentes sont affichées à l'utilisateur. La pertinence de chaque nouvelle annonce est définie sur la base de la pertinence des annonces les plus similaires déjà traitées. La similarité entre deux annonces est fonction des similarité des attributs de ces annonces (section 3.4.2).

Un autre exemple est le système MyGlobalNews ([Ferreira 01]). C'est un système de filtrage qui offre en plus des services de souscription et notification des utilisateurs des événements qui peuvent les intéresser (dissémination de l'information). Le manière de filtrer les résultats est la même que celle décrite dans l'exemple précédent. La requête de l'utilisateur est remplacée par son profil et le système va calculer le matching entre le profil et les documents. Les profils des clients ainsi que les documents sont représentés par des vecteurs de mots clés qui rend possible la comparaison entre les deux. Seuls les documents dont le matching dépasse un certain seuil sont inclus dans le résultat. Leur nombre ne dépasse pas celui donné dans le profil.

L'avantage du filtrage des résultats est sa simplicité parce qu'il ne nécessite aucune modification du fonctionnement des fournisseurs d'information. Tout le traitement est fait après l'exécution de la requête. Les inconvénients sont le volume de données échangées entre le serveur et le client et le risque d'élimination d'éléments pertinents. Le filtrage des résultats se fait le plus souvent sur le coté

client donc on a un gros transfert de données et en plus il y a un risque de suppression d'objets pertinents par le fait que les éléments du résultat qui sont jugés non intéressants sont éliminés définitivement.

3.1.2 Le re-ordonnancement des résultats de la requête

Le principe du re-ordonnancement est de modifier l'ordre de l'affichage des résultats au client. Il s'agit d'un post traitement qui, étant donné les éléments retournés par une requête, essaie de trouver une manière d'échanger leurs emplacements en fonction des préférences de l'utilisateur sans pour autant négliger l'ordre qui a été attribué aux objets du résultat par le moteur de recherche. L'échange de l'ordre d'apparition des éléments du résultats est fait généralement en appliquant une fonction qui permet de calculer le nouveau rang de l'objet.

Un exemple de fonction permettant d'effectuer le re-ordonnancement est donné par [Pretschner 99]. On considère que chaque document est caractérisé par les quatre catégories les plus significatives qui décrivent son contenu. Les catégories sont des mots clés et leur signification est calculée par la fréquence d'apparition de ces mots clés dans le texte. Dans ce cas le nouveau rang d'un document d_i noté $\rho(d_i)$ est fonction du rang qui lui a été attribué par le moteur de recherche ($\omega(d_i)$), de l'intérêt que l'utilisateur porte aux quatre catégories les plus significatives $\pi(c_j)$ (le poids que l'utilisateur donne à ces mots clés) et au niveau auquel les catégories décrivent le contenu ($\gamma(c_j, d_i)$).

$$\rho(d_i) = \omega(d_i) * [0.5 + ¼ \sum_{i=1}^{i=4} (\pi(ci) * \gamma(ci,dj)]$$

Dans cette formule $\omega(d_i)$ est l'ancienne position du document d_i, ($\gamma(c_j, d_i)$ est la fréquence d'apparition du terme c_j dans le document d_i Le système extrait les termes des documents les plus significatifs sur la base des fréquences TF-IDF (section 3.3.1). Pour un utilisateur donné l'intérêt qu'il porte à une catégorie est calculé en fonction du temps qu'il a passé sur le document et la longueur du document (en nombre de caractères). Initialement cet intérêt est égal à zéro et périodiquement le système analyse les documents visités pour mettre à jour les valeurs des intérêts du profil. Une des formules qui servent à calculer l'ajustement de l'intérêt $i(c_i)$ pour la catégorie c_i est:

$$\Delta i(ci) = \log \frac{\text{temps}}{\log \text{longueur}} * \gamma(ci, d)$$

Les autres formules ressemblent à celle-ci et la seule chose qui change est l'expression du logarithme :

$\log \frac{\text{temps}}{\log \text{longueur}}$ est remplacé par : $\log \frac{\text{temps}}{\log(\log(\text{temps}))}$, $\log \frac{\text{temps}}{\text{temps}}$ ou $\frac{\text{temps}}{\text{temps}}$

Dans ce cas l'intérêt que l'utilisateur porte sur les quatre catégories les plus significatives pour un document ($\pi(c_j)$) est donné par les valeurs actuelles de ces catégories dans le profil.

Le re-ordonnancement des résultats de la requête a le grand avantage de ne pas exclure des éléments du résultat. C'est en quelque sorte une garantie que l'utilisateur va trouver ce qui l'intéresse et le seul objectif de cette approche est de faire gagner du temps aux clients. C'est une technique très peu utilisée dans les systèmes de personnalisation parce qu'elle nécessite le calcule du nouvel rang de chaque élément du résultat ce qui demande beaucoup du temps.

3.1.3 La recommandation

La recommandation est un service de personnalisation qui consiste à proposer à l'utilisateur des éléments vis à vis de ses préférences ou en se servant de l'expérience des autres utilisateurs.

Un exemple de système de recommandation est proposé par [Mobasher 00]. Dans cet article à chaque utilisateur est associé un ensemble de transactions T = {t_1, …, t_m}. Chaque transaction t_i est un vecteur multidimensionnel de pageviews (vues sur les pages web). Ces transactions sont regroupées ensuite dans des clusters (groupes) de façon à maximiser la similarité entre les transactions du même cluster et minimiser celle des différents clusters. Sur la base d'un des clusters, l'article définit le profil d'un utilisateur comme étant un ensemble de couples (pageviews, poids associé) :

PRc={ (p,poids(p,PRc))/p=pageview et poids(p,PRc)>μ} , avec μ un seuil donné et

poids(p, PRc) = $\frac{1}{|C|} * \sum_{t \in C} w(p,t)$ où t sont les transactions du cluster C.

w(p, t) = 1 si $p \in t$ et 0 sinon

Dans la formule du poids(p, PRc), PRc est pris comme un cluster et pas comme un profil (dans cette approche le profil est définit par rapport à un cluster de transactions qui contiennent des pageviews ce qui permet la substitution profil->cluster).

A chaque pageview p_i correspond une valeur W_i^C dans le cluster C et dans la suite on utilisera les deux notations pour designer une page. Cette valeur est définie par :

W_i^C = poids(p_i, C) et 0 sinon

Dans ce cas, d'après les auteurs de l'article, l'ensemble des pageviews que le système peut recommander à l'utilisateur UREC(S), lors d'un session S est définit par :

UREC(S) = {W_i^C / $C \in$UP et Rec(S, W_i^C)$\geq \rho$} où ρ est un seuil donné, UP est l'ensemble des profils de l'utilisateur (il en possède un pour chaque cluster) et

Rec(S, W_i^C) est la valeur de recommandation de la pageview W_i^C pour la session S donnée par

Rec(S, W_i^C)= $\sqrt{poids(W_1^C,C)*match(S,C)}$ où le matching entre la session et le cluster est défini par :

Match(S,C) = $\frac{\sum_k W_1^C * Sk}{\sqrt{\sum_k (Sk)^2 * \sum_k (W_1^C)^2}}$ où

Sk = poids(p_k, PRc) si p_k appartient à la session actuelle et 0 sinon.

Dans cet article les recommandations de pages web sont faites en ne prenant en compte que les éléments que l'utilisateur consulte et le client est pris en isolation par rapport aux autres utilisateurs. Ceci suppose que le système ait rassemblé suffisamment d'informations sur l'utilisateur. Le calcul de l'ensemble d'éléments recommandables demande beaucoup de temps ce qui lors d'une navigation sur le web n'est pas souhaitable.

GroupLens ([Fink 00]) est un autre exemple de système de recommandation. Dans ce système l'utilisateur donne explicitement des notes aux éléments qu'il consulte ou le système devine son intérêt pour les éléments sur la base de ses activités passées. GroupLen offre des services de recommandation sur la base de la gestion de groupes d'utilisateurs ayant les mêmes intérêts. Il propose trois types de recommandations : personnelle, anonyme et consultation rapide. Comme dans l'exemple précédent lorsque le système dispose de suffisamment d'informations sur le client, il est capable de lui fournir des recommandations personnelles sur la base de ses activités passées sans prendre en compte les autres utilisateurs. Par contre si le système manque d'informations sur le client, il cherche à quel groupe le client pourrait appartenir et fait des recommandations collaboratives en lui proposant des éléments que les utilisateurs du groupe ont appréciés et que le client n'a pas encore consulté. La recommandation personnelle et collaboratives demandent beaucoup du temps de calcul. Lorsque le système doit fournir une réponse rapide, il fait des recommandations sur la base des affinités prédéfinies entre les produits sans prendre en compte les appréciations des utilisateurs. Par exemple un utilisateur qui achète une caméra se verra proposer des cassette et des piles appropriés.

La recommandation d'éléments est la technique de personnalisation la plus utilisée par les systèmes actuels. Elle a l'avantage de pouvoir répondre aux demandes des nouveaux utilisateurs et de fournir aux utilisateurs des résultats sans les borner dans leur choix. En plus, la recommandation peut être faite en prenant en compte le contenu des objets (premier exemple [Mobasher 00]) ou de façon indépendante du contenu des éléments (exemple de [Fink 00] (GroupLens)). Cette approche est abordée principalement dans le domaine de la recherche d'information parce qu'il manque la notion de requête (l'utilisateur est uniquement guidé) et la recommandations se fait au cours d'un session et pas sur une simple demande.

Comme nos avons vu dans cette section, les services de personnalisation ne sont pas encore biens compris. Ceux qui sont les plus utilisés ne modifient que très peu le fonctionnement des

systèmes de base (qui ne prennent pas en compte la personnalisation) comme par exemple le filtrage des résultats ou le re-ordonnancement des éléments du résultat ou ne font pas des restrictions dans l'espace de recherche comme la recommandation d'objets. Comme nous allons montrer plus loin (section 3.4.1), les approches qui proposent d'étendre les langages pour prendre en compte la personnalisation n'utilisent pas la notion de profil et l'utilisateur doit exprimer à chaque fois ses préférences. L'idée de ce rapport est de faire le premier pas (définition de modèle de profil) vers un système capable de réécrire les requêtes des utilisateurs en ajoutant à chaque fois les prédicats invariants.

3.2 Le contenu du profil

Dans cette section, nous allons voir les informations que les systèmes stockent le plus souvent sous forme de profils afin de personnaliser leurs services. De façon formelle, le profil est une mise en équation des préférences d'un utilisateur. Il doit contenir toutes les informations nécessaires pour le fonctionnement des systèmes qui l'utilisent. Le Tableau 1 donne quelques exemples de paramètres de profils utilisés par différents systèmes de personnalisation.

Chaque système de personnalisation propose un modèle de profil adapté au service qu'il offre. De façon formelle les données stockées dans les profils peuvent être classées selon trois catégories :

- Des exemples d'objets de l'espace de recherche

Ce modèle de profils sont utilisés en général par les systèmes de recommandation. L'idée principale est de stocker des exemples d'objets que l'utilisateur a jugés comme pertinents ou comme inintéressants pour ensuite comparer le contenu des profils des différents utilisateurs entre eux ([Bradley 00], [Mobasher 00]). S'il y a des similarités entre deux profils le système va recommander à l'utilisateur des objets que d'autres clients de profils semblables ont appréciés. Comme dans certains cas les objets peuvent être volumineux (comme des livres, rapports etc.), dans ces cas il est possible de ne stocker que leurs identifiants.

- Des caractéristiques extraites des objets de l'espace de recherche

L'idée des cette approche est de trouver une forme normale permettant la représentation les profils et les objets de l'espace de recherche. Ceci est fait le plus souvent en analysant les objets pour extraire les caractéristiques pertinentes pour l'utilisateur ([Ferreira 01], [Pretschner 99], [Croft 01], [Soltysiak 98], [Shearin 01], [Jung 02] etc.). Pour la plupart des approches existantes ces caractéristiques sont des mots clés. Dans l'approche RI une telle représentation permet d'ajouter les mots clés à la requête directement ce qui simplifie le passage entre les informations du profil et les prédicats de la requête. Dans certains cas les mots clés sont combinés avec d'autres informations portant sur l'importance des mots pour l'utilisateur (fréquences d'apparition, poids, notes etc.).

Dans [Shearin 01], le système extrait les caractéristiques des annonces d'appartements pour les stocker dans le profil de l'utilisateur. Lors d'une recherche, la pertinences d'une annonce est déterminée par la présence ou l'absence des caractéristiques contenues dans le profil.

[Ferreira 01] propose un système où les profils sont traités comme représentant les documents. Un profil (comme un document) est présenté comme un vecteur à N dimensions où les dimensions sont définies par les termes des centres d'intérêts. Le but visé dans ce cas est la comparaison directe entre les profils et les éléments recherchés qui simplifie la technique de personnalisation. La requête de l'utilisateur est remplacée par son profil et le système va calculer le matching entre le profil et les documents. Seuls les documents dont le matching dépasse un certain seuil sont inclus dans le résultat.

- Des attributs décrivant les préférences externes de l'utilisateur

Les informations des deux catégories précédentes sont directement liées aux objets de l'espace de recherche et à leur contenu, mais ne portent aucune information extrinsèque sur la qualité des informations ou sur l'utilisateur. Dans cette catégorie d'informations on retrouve les préférences de l'utilisateur liées aux sources des données, la qualité des informations et le type et la structure des éléments recherchés ([Fink 00], [Amato 99]). Les données personnelles font également partie de cette classe de données.

- Des règles et des formules de préférences

Dans certaines approches (ex. [Chomicki 02], [Torlone 02], [Lacroix 87] etc.) le profil de l'utilisateur est remplacé par des formules de préférence qui permettent de définir un ordre entre les éléments selon la pertinence de ces éléments pour l'utilisateur. Dans d'autres systèmes comme par exemple Personalization server [Fink 00] les préférences des utilisateurs sont capturées par des règles actives qui tentent de découvrire des généralités en regroupant les utilisateurs selon les caractéristiques comme l'age, le genre, les données démographiques, le nom du domaine, le type du browser, le système d'exploitation, la bande passante disponible, les activités personnelles ou la situation familiale.

Références	**Projet et produits**	**Informations du profil**
[Bradley 00]	**CASPER**(Case-Based Profiling for Electronic Recruitment).	couple (élément, pertinence) où élément est une annonce de travail et pertinence=(pertinent ou non pertinent)
[Ferreira 01]	*Architecture* supportant les services SDI(Sélective Dissemination of Information) : **MySDI** + prototype basé sur cette architecture : *service* **MyGlobalNews**	**vecteur de mots clés** (caracteristiques)+ nombre maximal de réponses.
[Mobasher 00]	Non	Profil=**ensemble de couples (page, poids)** où *page* est un objet de l'espace de recherche et *poid*s est l'intérêt de l'utilisateur pour cette page

[Pretschner 99]	Projet OBIWAN dans l'université de Kansas.	**mots clés =** Catégories extraites des pages combinées avec le temps que l'utilisateur a passé sur la page et sa taille.
[Lacroix 87]		clause **prefer** qui permet d'exprimer les préférences
[Fink 00]		intérêts, préférences, compétences, connaissances, environnement, systèmes (attributs non précisés)
	Projet GroupLens	couples (objet, note) et affinités entre les objets (définies d'avance dans le système)
	Personalization server	Groupes d'utilisateurs où le tri est fait selon des caractéristiques comme l'age, le genre, des données démographiques, le nom du domaine, le type du browser, le système d'exploitation, la bande passante disponible, les activités personnelles, la situation familiale
	FrontMind	**Règles actives** qui capturent les informations sur les utilisateurs, attribuent un utilisateur à un groupe et font des recommandations.
	Learn Sesame	Règles = régularités entre les caractéristiques des utilisateurs et les événements.
[Shearin 01]	Apt Décision : système de recherche d'immobilier	six caractéristiques positives et six caractéristiques négatives ordonnées d'un appartement.
[Jung 02]	Modèle formel de préférences.	Profil=**{ Préférence(w1), … ,Préférence(wn)}** où Préférence(wi) est la valeur de la préférence de l'utilisateur de la caractéristique wi des objets recherchés.
[Chomicki 02]	Approche formelle qualitative de formulation de préférences et leur intégration dans les langages de l'algèbre relationnelle.	opérateur **winnow** qui permet l'expression des préférences
[Soltysiak 98]	Approche de l'article	**Clusters contenant un nombre suffisant de documents**. (Un cluster = une caractéristique).

	Assistant web de browsing **Syskill & Webert.**	Vecteurs de **mots clés** et **les probabilités d'apparition** de chaque mot dans les pages intéressantes et non intéressantes.
[Croft 01]		un **ensemble de termes** extraits des documents que l'utilisateur a consulté
[Bradley 99]	CASPER(Case-Based Profiling for Electronic Recruitment).	**Les identifiants des annonces de travail vues avec une note sur cinq point que l'utilisateur donne à chaque annonce**.
[Villa]	Systèmes de recommandation.	**Les chemins de navigation** des utilisateurs avec des **estampilles de temps**.
[Torlone 02]	Article inspiré de [Chomicki 02].	**Relations de préférence** entre les éléments.
[Crabtree 98]	Syskill&Webert	informations non structurées : description textuelle (mots clés, phrases, documents, ens. de documents), probabilité d'apparition, fréquences d'apparition (TF,IDF), interactions de l'utilisateur avec le système, appréciations de l'utilisateur pour les éléments, Contact Personnel(nom, adresse, téléphone, identité, E-mail, langue), Démographiques(date de naissance, genre, situation familiale, revenus, éducation), Travail(titre professionnel, industrie, informations sur la compagnie), mot de passe, ID persona/nom
	Letizia	Mots clés
	NewT	exemples de situation-action
	WebWatcher	mots clés et fréquences TF-IDF
	Amalthaea	mots clés pondérés
	MORSE	couples (film,note)
	Autonomy's AgentWare	mots clés
	PSUN	entités simples (S-entities) et des supervisors (voir section 3.3.1)

[Amato 99]	**NCSTRL** (Networked Computer Science Technical Reference Library).	nom, date de naissance, genre, certificat d'identité, emploi et employeur, contacts personnels et professionnels, format des documents, langues maîtrisées, type de documents, date de création des documents, prix, dimensions, sources préférées, éditeurs, auteurs, séries, moyen de livraison, temps de livraison, enregistrements des interactions de l'utilisateur avec le système, données de navigation, conditions d'accès aux informations du profil (profil structuré voir plus loin)

Tableau 1 : Attributs des profils

Comme nous pouvons le voir, chaque approche définit un contenu des profils des utilisateurs en fonction de l'application qui l'utilise. Ce sont souvent des modèles simplifiés qui manquent de structuration et ne sont pas généralisables et réutilisables par d'autres approches.

[Amato 99] propose un modèle de contenu d'un profil utilisateur et son application pour la réalisation d'une librairie digitale. C'est la première approche où les informations sont structurées et qui offre un modèle général. Le modèle de profil contient cinq catégories (Figure 1).

Profil utilisateur :

Données Personnelles

Données Collectées

- Contenu du document
- Structure du document
- Source du document

Données de Livraison

- Moyen de livraison
- Moment de livraison

Données de Comportement

Données de Sécurité

Figure 1: Modèle de profil selon [Amato 99]

La première catégorie *Données personnelles* contient toutes les informations concernant l'identité de l'utilisateur. La deuxième *Données collectées* contient les informations nécessaires

pour décrire les préférences et restrictions sur les documents. Elle est divisée en trois sous-catégories : *contenu* (des informations sur le sujet du document, la langue...); *structure* (format, type, date de publication, dimensions...); *source* (provenance, auteurs, éditeurs...). Dans la catégorie *Données de livraison*, se trouvent les informations sur la manière d'envoyer des résultats à l'utilisateur. Ces informations sont regroupées selon deux sous-catégories : *moyen* (mode de livraison par exemple e-mail, fax téléphone,...) et *moment* (contient des informations temporelles sur le moment de livraison comme lors d'un changement, vers midi, entre 9h et 9h15,...). Ensuite, dans la catégorie *Données de comportement*, se retrouvent des enregistrements des interactions de l'utilisateur avec le système (URLs des pages visitées, documents lus et pertinence,...). Et finalement dans la catégorie *Données de sécurité*, sont placées des informations sur les conditions d'accès aux données du profil.

Une fois le profil générique définie, les auteurs proposent un schéma pour la réalisation d'une librairie digitale. Lors de l'inscription d'un utilisateur, un profil est créé automatiquement. A chaque profil correspond un identifiant unique :

Profils = ProfilID -> ProfilUtilisateur.

Les informations dans chaque profil sont triées dans cinq catégories ce qui peut être formalisé par :

ProfilUtilisateur=(DonnéesPersonnelles×DonnéesCollectées ×DonnéesLivraison × DonnéesActions × DonnéesSécurité)

La partie des données personnelles n'est pas explicitée dans l'article. Il est uniquement mentionné que les auteurs ont pris comme référence le schéma de P3P (Platform for Privacy Preferences)[3]. La catégorie des données collectées spécifie par quels documents l'utilisateur est intéressé. Un client peut avoir plusieurs centres d'intérêts en même temps. Les auteurs définissent la notion de domaine (topic) qui représente un besoin d'informations singulier. De cette manière le fait qu'un utilisateur puisse avoir plusieurs centres d'intérêts se traduit par le fait de placer un ensemble de domaines dans son profil dans la catégorie des données collectées. Chaque domaine est identifié par son identifiant :

DonnéesCollectées = DomaineID -> Domaine

L'identifiant d'un domaine est unique pour un utilisateur donné i.e. (ProfilID, DomaineID) est unique. Les attributs d'un domaine sont son nom, le contenu du document, la structure du document et la source du document :

Domaine = (NomDomaine × ContenuDoc × StructureDoc × SourceDoc)

Le contenu du document permet l'identification des documents pertinents pour un domaine donné. Il est composé du titre du document, une description textuelle (résumé ou abstract), une liste de mots clés pertinents, de catégories standards et de langues :

[3] *www.w3.org/P3P*

ContenuDoc = (Titre × DescriptionTextuelle × MotsClés × Catégories × Langues)

La structure du document comprend son format, son type, la date de publication et le prix :

StructureDoc = (FormatFichier × Type × DatePublication × Prix)

Dans une librairie les fichiers peuvent être stocké sous différents formats (postscript, word, etc.) et ils existent plusieurs types de documents (articles, rapports, livre, etc.). Ceci peut être formalisé par :

FormatFichier = (tous + postscipt + pdf + html + word +…)*

Type = (tous + livre + article + rapport + …)*

Dans ces expressions le signe « + » exprime le « ou » logique et l'étoile signifie que FormatFichier et Type peuvent prendre zéro ou plusieurs valeurs parmi celles données entre les parenthèses. « tous » est utilisé pour exprimer tous les formats ou tous les types de documents.

La catégorie source du document comprend une liste des sources permises et une liste des sources exclues :

SourceDoc = (SourcesPermises × SourcesExclues)

Le schéma d'une source comprend la collection, l'éditeur, la série et l'auteur

Source = (Collection × Editeur × Série × Auteur)

Les données de livraison tentent à spécifier le mode et le moment de livraison des documents à l'utilisateur :

DonnéesLivraison = (Mode, Moment)

Le mode de livraison permet à l'utilisateur de spécifier la manière de notification de l'arrivée des résultat. Il peut choisir parmi les moyens de livraison disponibles comme l'e-mail, le fax, le téléphone, etc. Dans le cas où l'utilisateur a choisi le mail ou un page web pour recevoir les résultats, le fichier attaché au mail ou affiché à la page est envoyé à une adresse spécifique (destination donnée dans le profil) et est mis sous une forme (layout) que l'utilisateur marque dans son profil :

Mode = (Moyen, Layout, Destination)

Le moment de livraison dépend en principe d'un événement qui peut être la mise à jour d'un document que l'utilisateur a consulté ou l'arrivée d'un nouveau document :

Moment = (NouveauDoc, ModificationDoc, T)

Les attributs NouveauDoc et ModificationDoc ne peuvent prendre que deux valeurs possibles : oui et non pour indiquer à l'utilisateur la cause de l'envoie des résultats. Le moment de livraison T peut être représenté soit par une intervalle, soit par la valeur DQP (dès que possible).

Les données d'action représentent des couples (action, document) désignant une action que l'utilisateur a effectuée sur un document donné. Pour un domaine donné, il est passible d'avoir plusieurs enregistrements de couples (action ,document) :

DonnéesActions = DomaineID -> (Action, DocumentID)

Les trois actions possibles sont : lu, pertinent, inintéressant et expriment comme suit le fait que l'utilisateur a lu le document, le fait que l'utilisateur a apprécié le document et le fait que l'utilisateur n'a pas apprécié le document.

Finalement les données de sécurité sont exprimées par une liste des utilisateurs (client ou serveur) qui peuvent accéder aux données du profil.

Le modèle de profil présenté par [Amato 99] est un premier pas vers la définition d'un profil générique. Bien qu'il propose une explication des différentes catégories du profil, il manquent des informations sur les interactions qui puissent exister entre les différentes catégories et entre le profil et l'exécution des requêtes. L'approche reste attachée à la réalisation d'une librairie digitale et ne prend pas en compte les préférences des utilisateurs sur la qualité des données.

Nous avons vu dans cette section que les contenus des profils proposés actuellement traitent des problèmes locaux et les données ne dépassent pas les besoins de l'application. Par conséquent à l'exception du modèle proposé par [Amato 99], ces profils ne sont pas réutilisables.

3.3 Construction et mise à jour du profil

Un des problèmes les plus difficiles à résoudre dans la personnalisation consiste à trouver des techniques de construction et de mise à jour les paramètres du profil d'un utilisateur. La façon la plus naturelle de construction et de mise à jour des informations du profil d'un utilisateur est de remplir manuellement les cases des paramètres du profil. Bien que le client connaît le mieux ses préférences, il est parfois incapable de les exprimer et définir de façon formelle. En plus, le processus de saisie de tous les paramètres du profil est souvent long et ennuyant. Afin de résoudre ces problèmes, ils existent plusieurs méthodes de capture des paramètres du profil de façon automatique. Ces méthodes sont liées à l'interprétation des activités du client sur les données du résultat. Ils existent plusieurs techniques de traitement des données et la plupart font partie des méthodes de data mining. Les trois groupes principales d'algorithmes de data mining sont les règles d'association, les techniques de classification et les algorithmes de clustering. Une étude approfondie sur les méthodes de data mining peut être trouvée dans [Han 01].

Dans ce rapport, nous allons essayer de décrire le fonctionnement de base des approches de construction et de mise à jour des données du profil et de donner des exemples utilisés pour la personnalisation. D'après [Crabtree 98] et [Stewart 97], les mécanismes de construction de profils peuvent être classés selon trois catégories : analyse statistique des termes, algorithmes de filtrage collaboratif et techniques d'apprentissage. A ceci nous allons rajouter la méthode interactive de capture des préférences de l'utilisateur.

3.3.1 Analyse statistique de termes

Ce processus peut être vu comme un pré-traitement des données et leur mise en forme normale pour pouvoir les manipuler ensuite. L'idée principale consiste à analyser le contenu d'un document et d'extraire des mots clés significatifs qui décrivent son contenu. Ces mots clés sont stockés pour être utilisés ensuite afin de comparer des éléments entre eux ou avec les préférences de l'utilisateur lorsqu'elles sont exprimées sous la forme de mots clés.

Il existe différentes structures de stockage et de représentation des mots clés en fonction du contexte dans lequel le profil est utilisé ([Soltysiak 98]). La manière classique de stockage des mots clés est un vecteur. Si l'ensemble des mots clés susceptible d'apparaître dans le profil n'est pas connu, on utilise un vecteur simple en mettant directement les mots clés dedans. Dans le cas contraire, on peut utiliser un vecteur de valeurs booléennes dans lequel chaque valeur correspond à la présence ou l'absence d'un mot donné. En plus dans certains cas, on peut rajouter un poids qui exprime l'importance de chaque terme et qui est souvent associé à la fréquence d'apparition du terme. On distingue deux types de fréquences :

- **TF** (Term Frequency) : fréquence d'apparition du terme dans un texte donné
- **IDF** (Inverse Document Frequency) fréquence relative d'apparition du terme dans un ensemble de documents généraux

En principe il est supposé que les mots qui apparaissent plusieurs fois dans un texte et sont peu utilisés sont plus significatifs pour le contenu d'un document que les autres. Le grand avantage de la représentation par des vecteurs est sa simplicité. Malheureusement la seule information que porte cette représentation est sur la présence des mots et on n'a aucune possibilité de deviner le contexte et le sens dans lequel les mots sont utilisés.

Une méthode de représentation des mots clés plus riche en sémantique est présentée par [Sorensen 95]. L'idée est de stocker les termes sous forme de **graphe orienté**. Dans ce graphe les sommets sont les mots et les arcs expriment les relations entre ces mots. Ces relations sont caractérisées par la probabilité qu'un mot apparaisse après un autre dans le texte (s'il y a un arc entre « langage » et « naturel » dont le poids est 0.8, cela signifie que dans 80% des cas le mot naturel est précédé par le mot langage). Les couples de mots sont appelés des entités simples ou **S-entités.** Chaque S-entité est caractérisée par deux valeurs : **son poids** qui est la probabilité que le second terme apparaisse après le premier et **sa force** qui est l'importance de l'entité par rapport aux autres S-entités du document (fréquence d'apparition). Ensuite les séquences qui satisfont les critères de qualité (poids et force supérieurs à un certain seuil) sont appelées des **Superviseurs**. Les Superviseurs de la même manière que les S-entités forment des chaînes. Par exemple, étant donnés deux S-entités (Superviseurs) S1 = « conception bases » et S2 = « bases de données » et si la combinaison « conception de bases de données » apparaît souvent dans le texte, on aura un lien de S1 vers S2 dans le graphe.

L'analyse statistique des mots clés est la méthode la plus utilisée parce qu'elle est basée sur des techniques bien comprises d'extraction de mots clés. Les inconvénients de cette approche se

trouvent dans le fait qu'elle ne peut être appliquée que sur des éléments textuels et que les mots sont analysés en isolation avec le reste du document ce qui entraîne une perte d'information contextuelle pouvant dégrader l'exactitude des données du profil. Une fois les mots clés extraits, on peut comparer deux éléments sur la base de la distance entre les vecteurs les représentant.

3.3.2 Algorithmes de filtrage collaboratif

Cette approche consiste à comparer les profils des utilisateurs entre eux afin de trouver des similarités dans leur comportement et dans leurs préférences et de faire des recommandations sur la base de ces similarités. Le filtrage collaboratif est utilisé le plus souvent dans les systèmes de recommandation d'éléments lorsque les profils des utilisateurs sont composés par un ensemble de couples (objet, pertinence de l'objet). Des exemples de systèmes qui font du filtrage collaboratif sont GroupLens, Alexa, FAB, SiteSeer etc. présentés dans [Gauch 99]. Cet article ne présente pas le fonctionnement de ces systèmes en détails, mais donne les références des articles qui les décrivent.

Pour illustrer le fonctionnement des algorithmes de filtrage collaboratif, nous allons nous appuyer sur un exemple donné par [Melville 01]. Pour cet algorithme les profils des utilisateurs sont composés par des couples (élément, poids) où poids indique l'intérêt (donné explicitement) que l'utilisateur porte à cet élément. Les grands pas de l'algorithme sont :

1. Calculer la similarité entre le profil de l'utilisateur courrant et les profils des autres utilisateurs
2. Sélectionner les n profils les plus similaires au profil du client
3. Faire des prédictions sur les éléments qui pourraient intéresser l'utilisateur en utilisant le contenu des n profils choisis auparavant

Dans le premier pas de l'algorithme, la similarité entre deux profils *a* et *u* ,où *a* désigne le profil de l'utilisateur actuel et *a* celui d'un des autres utilisateurs, est calculée en utilisant le coefficient de corrélation de Pearson, dont la formule est la suivante :

$$P_{a,u}=\frac{\sum_{i=1}^{m}(r_{a,i}-\bar{r}_a)\times(r_{u,i}-\bar{r}_u)}{\sqrt{\sum_{i=1}^{m}(r_{a,i}-\bar{r}_a)^2\times\sum_{i=1}^{m}(r_{u,i}-\bar{r}_u)^2}}$$ où m est le nombre de profils dans le système.

Dans cette formule $r_{a,i}$ est le poids que l'utilisateur a donne à l'élément i dans son profil et $\bar{r}_a$ est la moyenne des poids des éléments du profil de l'utilisateur a. Pour le troisième pas de l'algorithme, les auteurs de l 'article utilisent une formule qui permet de calculer la *prédiction* d'un article i selon le profil d'un utilisateur u, notée $p_{a,i}$:

$$p_{a,i}=\bar{r}_a+\frac{\sum_{u=1}^{n}(r_{u,i}-\bar{r}_u)\times P_{a,u}}{\sum_{u=1}^{n}P_{a,u}}$$

Les éléments dont la valeur de prédiction obtenue est grande sont recommandés à l'utilisateur actuel.

Les avantages du filtrage collaboratif sont son indépendance du format des objets manipulés et sa flexibilité lors du passage à l'échelle (la qualité des informations produites augmente avec le nombre d'utilisateur). Dans les formules de l'exemple donné plus haut les seuls paramètres qui interviennent sont les notes que les utilisateurs donnent aux éléments. Les objets du profil ne servent que lors de la recommandation pour indiquer au client des éléments pouvant l'intéresser. Par contre, cette approche manque d'informations sur le contenu des éléments et nécessite un nombre important d'utilisateurs ce qui fait partie de ses inconvénients. Quand un nouveau utilisateur s'inscrit, le système ne possède aucune information sur lui et ne peut pas comparer son profil avec ceux des autres clients. Même lorsque le système a collecté un petit nombre d'éléments pour un utilisateurs donné, il risque de lui recommander des objets non pertinents. Le filtrage collaboratif est le plus souvent utilisé dans le cas où il n'y a pas de profil stable de l'utilisateur pour faire des recommandations. Ce qui sert comme information de base est la session actuelle du client ou une note que l'utilisateur a donnée aux éléments.

3.3.3 Techniques d'apprentissage

Le principe de base de ces techniques est l'étude du comportement de l'utilisateur et la classification de ses caractéristiques ou celles des objets recherchés. L'avantage de l'approche est la fraîcheur et l'exactitude des données dérivées. Grâce au suivi de l'utilisateur cette approche permet la mise à jour de son profil. L'inconvénient se trouve dans la complexité des algorithmes utilisés qui nécessitent beaucoup de temps. Des exemples de telles techniques sont les réseaux de neurones, les méthodes de classification (raisonnement par cas, classificateurs bayésiens…), les règles d'association.

1. Les techniques de classification

Le but de la classification est d'attribuer un élément donné à un groupe existant. Les groupes sont connus à l'avance et sont donnés en paramètre à l'algorithme qui ensuite attribue les objets à un groupe selon certains critères. Les algorithmes de classification les plus utilisés dans la littérature sont les classificateurs bayésiens et le raisonnement par cas ([Bradley 00], [Soltysiak 98], [Schiaffino 00], [Croft 01], etc).

- Le raisonnement par cas

Le raisonnement par cas consiste dans la résolution du nouveau problème sur la base des situations survenues dans le passé. Le processus « mémorise » des instances de cas concrets et ensuite prend une décision en faisant une comparaison entre le nouveau cas et les cas connus.

C'est un processus incrémental qui nécessite au début des exemples servant comme base des prochaines classifications.

Un exemple de système utilisant cette technique est CASPER(Case-Based Profiling for Electronic Recruitment) ([Bradley 00]). C'est un système de recherche d'emploi qui sauvegarde les annonces que l'utilisateur dans un profil. A chaque annonce est associé une note de pertinence, cette note peut prendre deux valeurs : pertinent (+1) ou non pertinent (-1) ce qui correspond aux deux catégories selon lesquelles les éléments sont triés. Par la suite nous allons supposer qu'il existe une fonction *pertinence(e)* qui retourne la valeurs de la note de pertinence de l'élément *e* donné par le profil de l'utilisateur. Pour chaque élément du résultat, le système cherche les k plus proches(similaires) objets du profil de l'utilisateur et établit la pertinence de l'élément pour le client sur la base des pertinences des objets les plus similaires. La formule de calcul de la similarité entre deux annonces e_1,e_2 (Sim(e_1,e_2)) est la même que celle donnée dans l'exemple de la section 3.4.2. Pour le calcul de la valeur de pertinence d'un nouveau élément j par rapport à un profil donné, les auteurs de l'article utilisent la pertinence des k objets les plus similaires à j (*kppe(j)*) du profil où k est une constante. La formule de calcul de la pertinence (Per(j) indique la pertinence du nouveau élément) est :

Per(j) = 1 si $\left[\sum_{p \in kppe(j)} Sim(j,p) * pertinence(p)\right] > 0$ et -1 sinon

Sur l'exemple de la Figure 2, k=4 et l'élément en question sera marqué comme pertinent parce que les éléments les plus proches de lui sont pertinents.

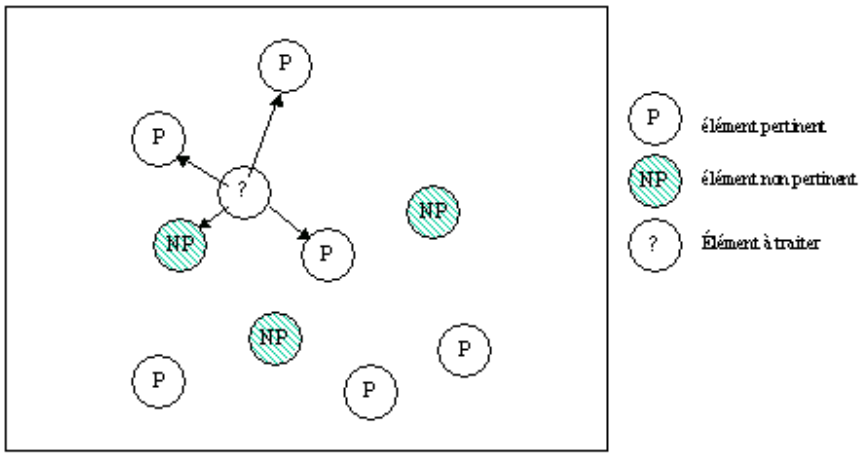


Figure 2: Exemple de calcul de la pertinence en utilisant la technique de raisonnement par cas

Les techniques de raisonnement par cas sont basées sur des méthodes bien comprises. Les inconvénients de cette approche se trouvent dans le fait que le système doit connaître l'utilisateur pour pouvoir comparer les situations précédentes avec les nouveaux cas et dans le fait que c'est

une approche incrémentale qui demande des ressources de calcul considérables parce que chaque nouveau cas est comparé avec tous les cas du profil.

- Les classificateurs bayésiens

Les classificateurs bayésiens représentent la technique jugée comme étant la plus performante par plusieurs systèmes d'information (FrontMind ([Fink 00]), Syskill & Webert ([Crabtree 98]), [Croft 01], etc). Un réseau bayésien est un graphe orienté acyclique qui représente une distribution probabiliste. Les nœuds sont des variables aléatoires et les arcs décrivent les corrélations probabilistes entre ces variables. Pour chaque nœud, on a une table qui spécifie la probabilité de chaque état du nœud en fonction de toutes les combinaisons possibles des états de ces prédécesseurs.

[Schiaffino 00] donne un exemple d'utilisation des réseaux bayésiens en combinaison avec la technique de raisonnement par cas. Chaque requête de l'utilisateur est prise comme un cas. Chaque nœud du réseau représente un attribut ou une caractéristique utilisés dans une requête. Les arcs décrivent les relations entre les attributs comme étant les fréquences d'apparitions des attributs dans les requêtes de l'utilisateur. Ces fréquences représentent également l'importance des attributs pour le client. Lorsqu'une nouvelle requête arrive, elle est stockée sous forme de cas et pour chaque attribut un nœud est ajouté dans le réseau. Ensuite, les liens entre ces caractéristiques sont rajoutés et les fréquences d'apparition sont mises à jour.

2. Les méthodes de clustering

A la différence de la classification, dans les techniques de clustering les groupes d'objets ne sont pas connus à l'avance et c'est l'algorithme qui se charge de la répartition des éléments en essayant de minimiser la similarité entre les éléments de deux clusters différents et de maximiser la similarité entre les éléments du même cluster.

Un exemple de système que utilise le clustering est donné par [Mobasher 00]. L'article présente deux techniques de clustering . La première PACT(Profile Aggregation based on Clustering Transactions) utilise les transactions tandis que ARHP(Association Rule Hypergraph Partitioning) prend en compte les pageviews apparaissant souvent ensemble. Les deux techniques (PACT et ARHP) peuvent être utilisées pour la personnalisation anonyme basée sur les données de navigation des clients.

- PACT

Cette technique consiste à regrouper les transactions similaires d'un utilisateur. Chaque transaction est un vecteur multidimensionnel de pageviews (vues sur les pages web), et le regroupement est fait à la base de la distance ou de la similarité entre les vecteurs. Pour chaque cluster, calcule le vecteur moyen mc est calculé. Dans cevecteur la valeur moyenne de chaque pageview est calculée comme étant le rapport de la somme des poids de la pageview dans toutes les transactions du cluster sur le nombre total de transactions dans le cluster :

Moy(p,C) = $\frac{\sum_{t \in C} poids(p,t)}{|C|}$ (les formules sont les mêmes que celles dans la section 3.1.3)

Cette valeur est normalisée de façon à être comprise dans l'intervalle [0,1]. Ensuite, les pageviews dont le poids normalisé est inférieur à un seuil donné sont éliminées. Un profil est donc obtenu sur la base d'un cluster c de transactions et représente un ensemble de couples (pageviews, poids associé) :

PRc={ (p, poids(p,PRc)) où p=pageview et poids(p,PRc)>seuil}

poids(p,PRc) = 1/|C| * $\sum_{t} w(p,t)$ avec t = transaction du cluster C et |C| est le cardinal de C.

La méthode PACT est très performante dans les systèmes de recommandation anonymes au début de l'arrivée d'un nouvel utilisateur ou avant de connaître son profil. Cette technique est à utiliser lorsque l'on veut une solution généralisée intégrant les pages de navigation et de contenu.

- ARHP

L'algorithme trouve d'abord des groupes d'éléments (URLs) qui apparaissent souvent ensemble dans les transactions appelés *ensembles d'éléments fréquents* (Ik) IS = {I1,…,Ik}. A chaque Ii est associé son *support* comme le rapport du nombre de transactions dans lesquelles Ii apparaît sur le nombre total de transactions. Les ensembles d'éléments dont le support est inférieur à un certain seuil sont éliminés. L'étape suivante est la construction d'un *hypergraphe* H=(V,E) où V est un sous-ensemble de pageviews et chaque arête est un élément de IS. *Un hypergraphe* est un graphe dans lequel les arêtes peuvent relier plus de deux éléments. A chaque arête est associé un poids ou intérêt :

Intérêt(I) = Support(I) / $\prod_{i \in I} Support(i)$

Support(i) = $\frac{|\{ t \in T : i \subseteq t \}|}{|T|}$ où T est l'ensemble des transactions.

Le graphe est partitionné en ensemble de clusters dans lequel chaque partition est examinée afin d'éliminer les nœuds de faible connectivité par rapport aux autres nœuds de la partition. *La connectivité* d'un nœud **v** dans un cluster **c** est obtenue en divisant la somme des poids des arêtes de c auxquelles v appartient par la somme totale des poids des arêtes de c. L'hypergraphe est partitionné de façon récursive jusqu'à un critère d'arrêt pour chaque partition(défini par le rapport des poids des nœuds éliminés sur le poids des nœuds non éliminés). Finalement, certains nœuds sont rajoutés si le pourcentage de nœuds encore dans l'arête d'origine excède un certain niveau spécifié par l'utilisateur. De cette manière, certains nœuds peuvent appartenir à plusieurs clusters. La connectivité d'un élément est prise comme facteur de base pour déterminer son poids dans le profil. ARHP a tendance à produire des recommandations plus intéressantes. ARHP peut être utilisé lorsqu'on veut produire un petit ensemble de recommandations spécifiques.

3. Les règles d'association

Les règles d'association ont pour but de définir des règles générales décrivant le comportement d'un utilisateur, l'appartenance d'un client à un groupe de clients ou une relation entre des objets. En principe, les règles d'association sont utilisées dans les systèmes de recommandation parce qu'elles permettent d'exprimer les affinités entre les éléments, entre les caractéristiques d'un élément ou entre les utilisateurs. Un exemple de règle d'association est « Les clients qui achètent des appareils photos achètent aussi des pellicules ».

Un exemple de système qui utilise les règles d'association est « **Personalization server** ». Les informations concernant l'environnement des utilisateurs sont capturées automatiquement par le système. Les règles sont attachées aux profils de groupes et permettent au système de définir l'appartenance d'un utilisateur à un ou plusieurs groupes. La construction et la maintenance des profils sont faites manuellement. Le format des règles actives dépend de leur utilisation. Exemple les règles d'adaptation : what; who; when (optionnel); conditions (optionnel); où what représente des informations décrivant le contenu de la catégorie, who décrit les personnes concernées, when spécifie la date d'activation de la règle et conditions regroupe des conditions supplémentaires. Le système est muni d'un module permettant l'intégration de données externes de caractérisation des utilisateurs.

Un autre exemple est le système « **Front Mind** ». Il comprend cinq composants interfaces. Le premier est *Front Mind Client*. Il permet aux clients d'envoyer des événements au serveur (login, requêtes, …) et de recevoir des recommandations en retour. *Front Mind Server* fournit un environnement personnalisé basé sur des règles actives grâce au sous-composants *Rule Evaluator* et *Learning and Inference Engine*. Rule Evaluator effectue le matching entre les événements envoyés par le client et les règles qui sont actives au moment de l'évaluation. Ensuite, sur la base des règles actives dont les préconditions sont vérifiées, le système renvoie le résultat au client. Learning and Inference Engine construit et maintient des modèles pouvant être construits de façon hiérarchique qui décrivent le comportement des utilisateurs. Les dimensions importantes de ses modèles incluent des pages référencées, des critères extraits des requêtes, des historiques des achats et des données démographiques. Un autre composant, *Business Command Center*, permet aux non-techniciens d'effectuer la définition et la maintenance de règles d'adaptation ainsi que de simuler leur performance sur la base des données historiques. *Business Object Developer* est l'outil grâce auquel les techniciens peuvent développer des business objets qui représentent la base des règles d'adaptation. Ces objets peuvent contenir des requêtes, des appels de fonctions et des procédures externes ou internes. Finalement, *Console Manager* gère la configuration du système. Les communications entre les composants sont effectuées via des messages représentés en XML. Toute action d'un utilisateur peut servir comme événement pour enclencher un certain nombre de règles du serveur (acquisition d'informations sur le client).

3.3.4 L'interactivité

Le profil d'un utilisateur doit contenir des informations précises sur ses préférences. Souvent il y a un décalage entre l'intention de l'utilisateur et ce qu'il désire réellement. C'est la raison pour laquelle certains systèmes d'information misent sur un processus de découverte des centres d'intérêts d'un utilisateur interactif et incrémental. Le fonctionnement de base de cette approche est fait sur un dialogue entre le système et le client qui diffère selon l'application. L'interactivité est principalement utilisée dans la recherche d'information et récemment en bases de données.

1. L'interactivité en recherche d'information

En recherche d'information, l'interactivité permet d'une part de capturer les centres d'intérêts d'un utilisateur et d'autre part de l'aider dans la définition de ses besoins. Initialement, l'utilisateur soumet une requête simple, souvent vague et pour laquelle le nombre de résultats est important. Ensuite, le système suggère des critères supplémentaires à l'utilisateur afin de cibler mieux sa recherche. La suggestion peut se faire de façon explicite ou implicite. Dans le premier cas, le système propose une liste de critères et demande à l'utilisateur de choisir ceux qui lui correspondent. Il se peut dans certains cas que la sélection des critères se fasse en plusieurs étapes et dans ce cas le processus de choix explicite peut être long et ennuyeux. C'est pourquoi certains systèmes d'information essaient de déduire les centres d'intérêts de l'utilisateur de façon implicite. De la même manière, l'utilisateur soumet une requête initiale au système qui ensuite lui propose des exemples de résultats et lui demande de choisir ceux qui l'intéressent. Ensuite, le système analyse les résultats que l'utilisateur a jugés comme pertinents afin d 'extraire les caractéristiques importantes.

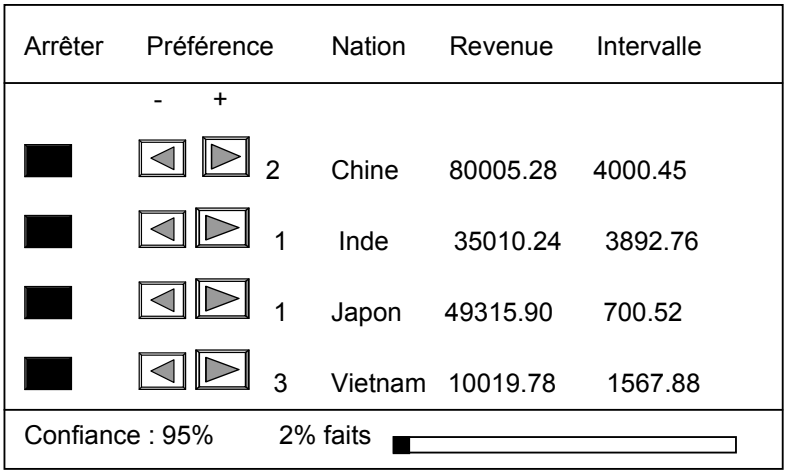
Un exemple de système de construction du profil de façon interactive est Apt Décision, un système de recherche d'immobilier ([Shearin 01]). En utilisant un profil initial basé sur des données saisies par l'utilisateur (nombre de chambres, ville et prix), le système affiche une liste d'annonces dont le nombre ne peut pas dépasser douze et qui correspondent à ces critères. Le système a une connaissance sur l'importance de chaque critère sur la base d'une étude du contexte de fonctionnement de l'agent et considère par exemple que le prix est plus important que les autres. En cliquant sur une annonce, l'utilisateur découvre toutes ses caractéristiques et réagit en les "glissant" dans les cases du profil. Les cases du profil sont ordonnées de la plus importante (à gauche) vers la moins significative (à droite) pour les critères positifs et négatifs. Si l'utilisateur ne veut pas réagir aux caractéristiques des annonces, il peut choisir une option qui fait en sorte que le système prend en charge la mise à jour du profil en donnant à chaque fois le choix entre deux annonces à l'utilisateur. L'annonce choisie est ensuite analysée pour extraire les attributs qui ont motivés le choix du client.

2. L'interactivité en bases de données

L'interactivité est très peu utilisée en bases de données en raison du caractère statiques des requêtes. En général, l'espace de recherche est borné et tous les prédicats sont exprimés en même temps dans la requête qui ensuite est exécutée sur les sources de données. Une technique

potentiellement utilisable pour faire interagir l'utilisateur est de lui proposer des valeurs d'attributs qui ne figurent pas dans sa requête initiale et ensuite lancer une nouvelle requête sur le résultat de la précédente. Ce processus peut être répété jusqu'à ce que l'utilisateur soit satisfait par le résultat. Une autre approche consiste à gérer la mémoire de façon à traiter d'abord les éléments qui sont pertinents pour l'utilisateur. Ceci est fait en chargeant dans un buffer les tuples auxquels le client porte plus grande attention pour ensuite envoyer en premier ces tuples aux opérateurs.

[Hellerstein 99] propose une technique d'exécution des requêtes de façon interactive. Cette technique d'ordonnancement est utilisée pour l'agrégation en ligne lorsque l'utilisateur veut obtenir rapidement des estimations précises sur une valeur des attributs dans le group by. Le principe consiste à modifier l'ordre dans lequel les données des sources sont traités par les requêtes. On divise le processus d'exécution de la requête en quatre phases : production, ordonnancement, traitement et consommation. La production consiste à lire les données du disque ou à les charger du réseau. L'ordonnancement est la phase pendant laquelle les préférences de l'utilisateur sont prises en compte. Il se fait à l'aide d'un buffer dans lequel sont stockées les données qui doivent être traitées en premier. L'utilisateur possède une interface lui permettant de modifier l'importance qu'il donne à chaque valeur des attributs de la clause « GROUP BY » de sa requête.



Requête : Select avg(revenue), nation

From ventes, branches

Where ventes.id = branches.id

Group by nation

Figure 3 : Exemple d'interface d'interaction en Bases de Données

Sur l'exemple de la figure 3, l'utilisateur soumet une requête qui sélectionne la moyenne des revenues par nation (expression de la requête sous la figure). Une fois que le système a identifié les valeurs possibles dee attributs de la clause *group by*, il donne à l'utilisateur la possibilité de modifier l'importance de chacune de ces valeurs en utilisant deux boutons : un pour diminuer et un

autre pour augmenter la pertinence des valeurs. Le taux d'éléments de chaque groupe dans le buffer dépend de l'importance que l'utilisateur attribue à ce groupe. Lors d'un chargement, les éléments qui ne sont pas insérés dans le buffer sont stockés dans une mémoire auxiliaire pour être extraits ensuite au cas où le taux d'éléments d'un groupe dans le buffer est inférieur à la valeur désirée. Lorsque les préférences de l'utilisateur changent on modifie uniquement le taux des éléments des différents groupes dans le buffer. La phase de traitement comprend l'application des opérateurs de la requête sur les éléments et la consommation est le moment où l'utilisateur prends en compte le résultat.

Bien qu'il existent déjà plusieurs techniques permettant de capturer les préférences d'un utilisateur, leur utilisation n'est pas encore bien comprise pour permettre la construction et la mise à jour de son profil de façon automatique et complètement transparente pour lui. Actuellement nous pouvons espérer que ceci se fasse avec une implication minimale de l'utilisateur. Le principe est que l'utilisateur doit avoir le contrôle de son profil à tout moment afin de pouvoir invalider les mises à jour incorrectes du profil. Après une telle invalidation, le gestionnaire doit prendre en compte et modifier la manière (le processus) de gestion et de mise à jour du profil.

3.4 Utilisation des données du profil

Comme nous avons vu précédemment, les profils réellement utilisés par les systèmes de personnalisation sont composés soit de couples (objet, pertinence), soit par des vecteurs et peuvent être comparés facilement entre eux ou avec les éléments de l'espace de recherche. Ce sont des approches orientées vers la recherche d'information et n'intègrent pas la notion de requête à prédicats. Dans cette section nous allons voir des approchent qui proposent des extensions des langages existants pour permettre la prise en compte de la personnalisation. Le deuxième problème qui sera traité porte sur la manière de calculer et implémenter les nouvelles clauses et opérateurs.

3.4.1 Les extensions des langages pour supporter la personnalisation

Comme nous avons spécifié précédemment, l'idée principale de la personnalisation est de faciliter l'écriture des requêtes de l'utilisateur en les enrichissant avec des données invariantes communes à toutes les requêtes. Dans ce contexte, l'utilisateur doit être capable d'exprimer toutes ses préférences et exigences de façon simple. Ces exigences doivent être traduites dans un langage afin d'être ajoutées aux requêtes et exécutées. Les langages classiques comme SQL ont une syntaxe précise et ne supportent pas la notion de préférence. Pour cette raison, ces dernières années ont vu naître des extensions du langage qui supportent l'évaluation de fonctions externes ou utilisent d'autres clauses permettant d'exprimer des préférences. Parmi ces approches on retrouve des opérateurs complexes comme l'opérateur Winnow (Best) [Chomicki 02] ou l'opérateur

Skyline [Borzsonyi 01] et l'ajout de la nouvelle clause prefer [Lacroix 87]. L'ajout de nouveaux opérateurs au langage entraîne des ajouts aussi au niveau de l'exécuteur des requêtes tandis que la permission d'utilisation de fonctions externes laisse une plus grande liberté de l'implémentation de ces fonctions. Dans ce deuxième cas, la manière d'après laquelle une fonction sera exécutée dépend de l'utilisateur et par conséquent le sens que l'utilisateur donne à ces opérateurs sera contenue dans le profil. Dans les deux cas, les informations du profil vont être utilisées comme des paramètres ou des critères de sélection et vont être intégrées aux requêtes.

- L'opérateur **Skyline**

L'opérateur Skyline ([Borzsonyi 01]) est définit dans le contexte d'une base de données relationnelle interrogeable par le langage SQL. Les éléments du skyline sont ceux qui ne sont pas dominés par d'autres éléments. Un élément domine un autre s'il est au moins aussi bon que le dominé dans toutes ses dimensions et meilleur dans au moins une dimension. Un élément peut être vu comme un point dans un espace à m dimensions où m est le nombre d'attributs qu'il possède. Une des propriétés intéressantes du skyline est que pour toute fonction monotone f : M->R si p maximise cette fonction, alors p est contenu dans le skyline (le poids des critères n'est pas important car tout maximum est contenu dans le skyline) et en plus pour chaque point du skyline il existe une fonction monotone qu'il maximise (il ne contient pas de points non-pertinents). Pour son intégration on étend le langage SQL avec une clause optionnelle dont la syntaxe est la suivante :

SKYLINE OF [DISTINCT] d_1 [MIN | MAX | DIFF], ... , d_m [MIN | MAX | DIFF] où les d_i sont les dimensions du skyline (attributs de la relation).

Sur l'exemple de la figure 4, l'utilisateur cherche des annonces de logement dont le prix et la distance jusqu'à une station de métro sont minimales, la surface et le nombre de chambres sont maximales et la différence du propriétaire peut être expliquée par le fait que le client veut demander à chaque propriétaire s'il propose d'autres logements et ceci permet à l'utilisateur de consulter moins d'annonces.

```
SELECT *
FROM Annonces
WHERE ville = 'Paris'
SKYLINE OF prix MIN, distance_metro MIN,surface MAX, chambres MAX, proprietaire DIFF
```

Figure 4: Exemple de requête avec une clause SKYLINE OF

- L'opérateur **Winnow** (Best)

L'opérateur Winnow [$w_C(R)$] ([Chomicki 02]) (appelé Best par[Torlone 02]) permet la sélection des éléments préférés d'une relation (R) par rapport à une formule de préférence (C). Comme l'opérateur Skyline, il est défini dans le contexte des bases de données relationnelles. De cette façon, une requête avec des préférences est une requête de l'algèbre relationnelle contenant au

moins une occurrence de l'opérateur winnow. Si une relation de préférence sur un schéma relationnel est définie en utilisant une formule de préférence qui est un ordre partiel strict, alors pour toute instance non vide et finie de ce schéma l'opérateur winnow renvoie un résultat non vide. Deux tuples qui appartiennent au résultat de l'opérateur ont soit les mêmes valeurs, soit sont indifférents entre eux (aucune dominance ne peut être établie). La formule de préférence C est représentée par une disjonction de formules sur des variables libres correspondant aux valeurs de deux tuples t_1 et t_2 i.e. $C=D_1VD_2V...VD_k$. Chacune des clauses D_i peut être vue comme une conjonction de trois clauses ϕ_i (sur des variables de t_1), ψ_i (sur des variables de t_2) et γ_i(sur des variables communes). Dans ce cas, il existe une traduction de ϕ_i vers une condition de sélection Φ_i sur R et de ψ_i vers une condition de sélection Ψ_i sur q(R) où q est un retitrage de R. De même, γ_i peut être traduite comme une condition de jointure Γ_i entre R et q(R). Les techniques qui permettent de faire ces traductions ne sont pas expliquées dans l'article. Dans ce cas on peut trouver une substitution de l'opérateur Winnow par des opérateurs de l'algèbre relationnelle (Figure 5) pour ensuite intégrer ces opérateurs à la requête.

$w_C(R) = q^{-1}\{ q(R) - \Pi_{q(R)}[\cup_{i=1}^{i=k}(\sigma_{\Phi I}(R) |><|_{\Gamma i} \sigma_{\Psi I}(q(R)))]\}$ où q^{-1} est l'inverse de q

$\Pi_{\{A\}}$: opérateur de projection sur un ensemble d'attributs A

$\cup_{i=1}^{i=k}$: union d'ensembles d'éléments numérotés de 1 à k

σ_{Φ} : opérateur de restriction sur un ensemble de critères Φ

$|><|_{\Gamma}$: opérateur de jointure selon un critère Γ

Figure 5: Traduction de l'opérateur Winnow en algèbre relationnelle

Soit une relation Repas dont le schémas est : Repas (Plat, TypePlat, Vin, TypeVin) et une formule de préférence C_2, définie sur cette relation comme suit :

$(p1, tp1, v1, tv1) \succ_2 (p2, tp2, v2, tv2) \equiv$ (p1=p2 $\wedge$ tp1='poisson' $\wedge$ tv1 = 'blanc' $\wedge$ tp2='poisson' $\wedge$ tv2='rouge')
$\vee$(p1=p2 $\wedge$ tp1='viande' $\wedge$ tv1='rouge' $\wedge$ tp2='viande' $\wedge$ tv2='blanc')

Si l'opérateur winnow est appliqué sur la relation Repas en utilisant la formule de préférence $C_2(W_{C2}(Repas))$, alors le résultat va prendre en compte le fait que l'utilisateur préfère le vin blanc lorsqu'il consomme du poisson et le vin rouge en présence de la viande.

- La clause **prefer**

Dans l'approche de [Lacroix 87], les préférences de l'utilisateur sont décrites par une clause "prefer" qui s'ajoute au langage de requête habituel. Il n'y a pas vraiment de profil stable de l'utilisateur. Les clauses de préférence peuvent être utilisées plusieurs fois dans la même requête. Ils existent deux cas possibles: préférences ayant la même importance et préférences imbriquées. Lorsque l'on évalue des préférences d'importances différentes ou préférences imbriquées, on se sert de la syntaxe "from witch prefer..." en considérant que cette clause est moins importante que la précédente. Le deuxième cas possible est la définition de préférences de même importance. Leur expression est faite par la répétition de la clause prefer. Le résultat est formé par les tuples qui satisfont un nombre maximal de préférences. Sur l'exemple de la figure 6, l'utilisateur cherche des annonces de logement à Paris de deux chambres minimum. Parmi les annonces qui vérifient les deux premiers critères ,il préfère celles dont le prix est inférieur à mille euro et où il y a deux ou plus salles de bain. Finalement les annonces pour lesquelles le logement se trouve au maximum au deuxième étage sont préférées. Les annonces du résultat doivent toutes vérifier les deux premiers critères (lieu = 'Paris' et chambres $\geq$ 2). Ensuite elles doivent répondre aux deux clauses prefer de même importance (prix < 1000 et sdb > 1). S'il n'existe pas de tels annonces, le résultat comprendra celles qui en vérifient au moins une. Finalement parmi ces annonces l'exécuteur ne gardera que celles, si elle existent, qui satisfont la dernière condition (etage < 3).

```
Select *
From Annonces
Where lieu = 'Paris'
        And chambres ≥ 2
Prefer those having prix < 1000
Prefer those having sdb > 1
From which prefer those having etage < 3
```

Figure 6 : Exemple de requête avec des clauses 'prefer'

- Preferences SQL

Une autre approche d'extension du langage pour qu'il puisse supporter la personnalisation est de permettre l'utilisation de fonctions prédéfinies ou des fonctions externes. Dans ce cas, on obtient une meilleure flexibilité car l'utilisateur peut déterminer lui-même le sens qu'il donne à chaque fonction.

D'après [Kießling 01], chaque préférence peut être exprimée par un opérateur. On propose la définition de préférences complexes et leur expression dans un langage formel. Les préférences peuvent être numériques (AROUND, BETWEEN, LOWEST, HIGHEST) ou non-numériques (POS, NEG, POS/NEG, POS/POS, EXPLICIT). Le pas suivant est leur écriture dans le langage **Preference SQL.** La nouveauté dans Preference SQL est la **clause « PREFERING … »** qui permet

l'utilisation des fonctions de préférence. On utilise également la clause **« CASCADING »** permettant d'exprimer des préférences hiérarchiques. Les fonctions d'expression des préférences utilisées par le langage seront décrites dans la section 3.4.2.

```
SELECT *
FROM Voitures
WHERE marque = 'Opel'
PREFERING ( cateorie = 'roadster' ELSE categorie <> 'passenger'
            AND prix AROUND 9000
            AND HIGHEST(puissance))
            CASCADE couleur = 'rouge'
                  CASCADE LOWEST(kilometrage);
```

Figure 7 : Exemple de requête en Preferences SQL

Sur l'exemple de la figure 7, l'utilisateur cherche à acheter une voiture de marque Opel, de préférence de catégorie roadster, mais pas passenger. Avec la même importance que la catégorie il désire que le prix soit autour de 9000 euro et la puissance aussi grande que possible. Finalement, il a une préférence moins importante que les précédentes pour les voitures de couleur rouge et au cas où plusieurs voitures répondent à tous les critères précédents, il souhaite que le kilométrage soit le plus petit possible.

Préférences SQL peut être vue comme une surcouche d'un SGBD classique qui permet l'expression des préférences d'un utilisateur. Une fois la requête écrite en Preferences SQL le système la traduit en code SQL normal qui est ensuite exécuté sur le SGBD. Preferences SQL est supporté par certains SGBDs (DB2, Oracle 8i, MS SQL Server) via un traducteur de requêtes Preference SQL en code SQL92.

Comme nous l'avons vu précédemment, certaines extensions de langages des requêtes permettent de définir des préférences hiérarchiques (clause **prefer :** « from witch prefer …»**,** Preferences SQL :clause CASCADING). L'inconvénient de ces approches est le fait qu'on ne peut pas définir les dépendances qui existent entre tous les attributs. Par exemple, il est possible que le fait d'avoir un prédicat de la requête vérifié ne soit pas pertinent pour l'utilisateur, mais en le prenant en combinaison avec une autre condition de la requête son importance s'accroît.

Une approche d'expression de l'importance des prédicats qui permet d'exprimer les corrélations entre les conditions de la requête est donné par [Franklin 03]. Le profil d'un utilisateur comprend deux clauses **DOMAIN et UTILITY**. La clause DOMAIN définit les objets des centres d'intérêts de l'utilisateur et donne un nom fictif à chaque objet. Ensuite, dans la clause UTILITY, on spécifie les valeurs relatives de l'intérêt de chacun de ces objets en utilisant des équations d'utilité. Ces

équations d'utilité sont sous la forme **U(*Nom d'objet*) = *Expression d'utilité*** et les expressions d'utilité sont données soit par un **Entier**, soit par une expression conditionnelle **IF *condition* THEN *expression d'utilité* ELSE *expression d'utilité***, soit par une fonction dont la signature est **UPTO(*Entier*, *Expression d'utilité*, *Expression d'utilité*).** La fonction UPTO est un opérateur de définition de seuil. UPTO (x, y, z) signifie que les x premiers éléments du résultat ont une utilité y et tous les autres une valeur de z. Sur l 'exemple de la Figure 8, on définit le profil d'un voyageur qui veut se rendre à Boston. Il est composé de cinq objets d'intérêt. Pour chacun de ces centres d'intérêts, on définit une équation d'intérêt. L'utilité de chaque objet décrivant un restaurant a une utilité de un. Ensuite, l'utilité des sites de location de voitures dépend de la présence de plans des directions de l'aéroport vers la ville ([# Di > 0]). En présence de plans des directions, l'utilité des deux premiers éléments est deux et pour tous les autres zéro. De la même manière, les utilités des plans des directions et des hôtels dépendent respectivement des hôtels et de la présence de plans des directions ou des horaires des navettes. Finalement, l'utilité du premier horaire des navettes a une utilité de trois et tous les autres n'en ont aucune. Une autre manière d'exprimer les dépendances entre les objets du centre d'intérêts est d'inclure une condition dans la fonction UPTO. Par exemple l'utilité des hôtels **U(Ho [# Di > 0 OU # Na > 0]) = UPTO (1, 2, 0)** peut s'écrire comme **U(Ho) = UPTO (1, IF #Di>0 ou #Na>0 THEN 2 ELSE 0, 0)**.

PROFILE Voyageur

DOMAIN

LV = www.hertz.com (compagnies de location de voitures)

Na = « horaires des navettes qui vont de l'aéroport vers Boston»

Di = « cartes des directions de l'aéroport vers Boston »

Ho = « hôtels à Boston »

Re = « restaurants à Boston »

UTILITY

U(Re) = 1

U(LV [# Di > 0]) = UPTO (2, 2, 0)

U(Na) = UPTO (1, 3, 0)

U(Di [# Ho > 0]) = UPTO (1, 1, 0)

U(Ho [# Di > 0 OU # Na > 0]) = UPTO (1, 2, 0)

END

Figure 8 : Exemple de profil selon [Franklin 03]

Les approches de personnalisation présentées dans cette section permettent d'exprimer des préférences, mais n'utilisent pas la notion de profil. L'utilisateur est contraint d'écrire à chaque fois la requête complète qui définit son besoin d'information ce qui est un inconvénient non négligeable. Comme nous l'avons dit précédemment, l'objectif de la personnalisation est d'éviter à l'utilisateur d'écrire à chaque fois la partie commune à ses requêtes.

3.4.2 Calcul et implémentations des nouvelles clauses et opérateurs

Les modifications du langage d'expression des requêtes entraînent des changements dans la manière d'exécuter de ces requêtes qui doit prendre en compte les préférences et les exigences de l'utilisateur. D'un coté, on doit choisir les implémentations des nouveaux opérateurs qui correspondent le mieux à la demande du client et d'un autre coté, il faut utiliser les algorithmes et les stratégies d'exécution de façon à lui fournir un service de qualité. Dans cette section, nous allons voir les algorithmes d'évaluation des nouvelles clauses ou opérateurs et les implémentations des fonctions de personnalisation.

- Calcul du Skyline

Selon [Borzsonyi 01], la clause SKYLINE OF est calculée après le SELECT... FROM... WHERE... GROUP BY... HAVING... , mais avant le reste des clauses comme ORDER BY... ou TOP N . Dans ce contexte, étant donnés deux tuples $p=(p_1,...,p_n)$ et $q=(q_1,...,q_n)$, p va dominer q selon la requête SKYLINE OF d_1 MIN,..., d_k MIN, d_{k+1} MAX,..., d_l MAX, d_{l+1} DIFF,..., d_m DIFF si les trois conditions sont vérifiées :

- $p_i<=q_i$ pour i de 1 à k

- $p_i>=q_i$ pour i de k+1 à l

- $p_i=q_i$ pour i de l+1 à m

Il est à remarquer que les attributs de m+1 à n n'ont aucun impact sur le résultat, mais il n'y a pas de projection implicite. Pour les implémentations de l'opérateur, les auteurs de l'articles proposent d'étendre un système de BD objet-relationnel ou orienté objet avec le nouvel opérateur logique (skyline). Le calcul du skyline à une dimension est équivalent à l'un des agrégats min, max ou distinct du langage SQL sans clause SKYLINE OF. Le calcul du skyline à deux dimensions est aussi trivial et nécessite un parcours de la table si elle est triée selon une des dimensions. Dans ce cas, il faut juste déterminer la dominance entre le tuple courant et le dernier tuple du skyline. Si la table n'est pas triée, il est possible d'éliminer certains tuples pendant la phase de tri. L'article présente deux algorithmes de calcul du skyline dans le cas général à n dimensions : sur la base de boucles imbriquées (BNL) et sur l'algorithme diviser et conquérir (D&C).

L'idée de l'algorithme avec des boucles imbriquées est la comparaison de chaque tuple avec tous les autres. Afin d'optimiser son fonctionnement, les auteurs de l'article proposent qu'à chaque itération l'algorithme ne produise pas un tuple unique, mais un ensemble de tuples. La taille de cet ensemble ou fenêtre est fixe et lors de la prise en compte d'un tuple p, trois cas sont possibles :

1. si p est dominé par un ou plusieurs tuples de la fenêtre, il est éliminé et ne va plus être pris en considération (la comparaison s'arrête au premier tuple dominant);

2. si p domine un ou plusieurs tuples de la fenêtre, ces tuples sont éliminés et ne vont plus être considérés et p est inséré dans la fenêtre;

3. si p est incomparable avec tous les tuples de la fenêtre, il y est inséré s'il y a de la place, sinon il est mis dans un fichier temporaire qui contient les tuples qui vont être traités dans l'itération suivante.

A la fin de l'itération tous les tuples de la fenêtre qui ont été comparés à tous les autres (du fichier temporaire) peuvent être retournés au résultat, les autres doivent être comparés lors de l'itération suivante. Afin de connaître les tuples à retourner, ils sont indexés à leur insertion. De cette façon tous les tuples de la fenêtre avec un numéro plus petit que celui du tuple inséré le plutôt dans le fichier temporaire peuvent être mis dans le résultat. Dans certain cas cet algorithme peut être amélioré en mettant au début de la fenêtre les tuples dominants ce qui diminue le nombre de comparaisons. Une autre variante consiste à ne garder dans la fenêtre que l'ensemble de tuples le plus dominant. La découverte d'un tel ensemble n'est pas facile et nécessite des connaissances complémentaires et des calculs de comparaison supplémentaires.

L'algorithme diviser et conquérir est basé sur l'approche suivante :

1. Calculer la valeur moyenne dp selon une des dimensions p.

2. Diviser la relation en deux parties P1 et P2 de façon à ce que P1 contient les tuples de valeur meilleure que dp dans la dimension p et P2 les autres tuples.

3. Calculer le skyline des deux partitions en appliquant à nouveau le même procédé de partitionnement de façon récursive.

L'algorithme de découpage s'arrête lorsqu'un sous-ensemble ne contient qu'un tuple ou très peu de tuples. Et finalement le résultat est obtenu en fusionnant les sous-ensembles (il faut juste éliminer les tuples de P2 dominés par des tuples de P1 car les tuples de P1 sont meilleurs que ceux de P2 dans au moins une dimension). Cet algorithme offre de très mauvaises performances si la table ne tient pas en mémoire centrale en raison du nombre d'E/S nécessaire. Pour résoudre ce problème, l'article propose de découper la relation en m parties de façon que chacune de ces parties tienne en mémoire et appliquer l'algorithme séparément à chaque partition. Ce partitionnement peut être fait dans le premier ou le troisième pas de l'algorithme. Une autre astuce lorsque le résultat du skyline est petit est de charger autant de tuples que possible en mémoire centrale et d'appliquer l'algorithme dessus. Cette technique est appelée "early skyline", elle doit être exécutée avant la phase de partitionnement et permet un premier filtrage des tuples. D'autres améliorations peuvent être faites en utilisant des structures indexées, ordonnées comme les arbres B ou R. Dans le cas des arbres-B, il est supposé qu'il existe des indexes ordonnés pour chaque dimension du skyline. Dans ce cas, il suffit de trouver le premier matching en scannant tous les indexes simultanément. Tous les tuples qui n'ont pas été inspectés ne font pas partie du skyline et pour les autres un des algorithmes précédents est appliqué. Dans le cas des arbres R, l'idée est de faire un parcours en

profondeur et d'éliminer les branches pour lesquelles il est sur d'avoir une dominance. Ces techniques sont efficaces si le résultat du skyline est petit. Leurs performances diminuent si le nombre de dimensions augmente et en présence de prédicats de jointures ou des group by. L'opérateur peut être exécuté avant les jointures non-restrictives et de ce fait réduire la taille des relations à joindre ou en partie lors de la jointure en éliminant les tuples pour lesquels on est sur d'avoir une dominance. Il peut être aussi combiner avec la clause TOP N si les dimensions de cette clause sont un sous-ensemble des dimensions de la clause skyline. Dans ce cas, l'algorithme s'arrête lorsqu'il a produit les N meilleurs tuples ce qui est facile en présence d'index et sinon nécessite plusieurs étapes chacune calculant le skyline pour un intervalle donné des valeurs d'une des dimensions (à chaque étape les valeurs des bornes de l'intervalle sont augmentées jusqu'à avoir N tuples dans le résultat).

Les variantes de BNL sont à utiliser dans le cas où le skyline et le nombre de dimensions sont petits(BDs corrélées). Elles sont aussi très sensibles à la corrélation des données (Les BD anti-corrélées dégradent les performances du BNL) et ne dépendent pas de la taille de la mémoire centrale. Les variantes de D&C sont moins sensibles au nombre de dimensions et la corrélation des données. La meilleure variante est D&C avec partitionnement en m parties et early skyline. Leurs performances augmentent avec la taille de la mémoire centrale. Comme la taille du skyline n'est pas linéairement dépendante de la taille de la BD, les BNLs deviennent plus attractives si la taille de la BD augmente.

- Calcul de l'opérateur Best

L'algorithme de calcul de l'opérateur Best comporte plusieurs phases de parcours d'un ensemble C_i de tuples candidates à la ième étape. A chaque itération un seul tuple est ajouté au résultat Out_i. Au début C_1 est initialisé par tous les tuples de la relation (C_1=r). Ensuite chaque tuple t (un seul à chaque itération) est comparé aux autres tuples de C_1. Lors de la comparaison de deux tuples t_1 et t_2 (t_1 étant le tuple choisi pour l'itération) trois cas sont possibles :

1) si les deux tuples sont indifférents, t_1 reste sélectionné et t_2 est mis dans un ensemble de tuples non-résolus U_1;

2) si t_1 domine t_2, t_2 est ajouté à l'ensemble Dt_1 de tuples dominés par t_1 (il est supprimé des autres Dt_i) et t_1 reste le tuple sélectionné;

3) si au contraire t_2 domine t_1, t_2 devient sélectionné et t_1 est ajouté à Dt_2.

Lorsque C_1 est vide, le tuple sélectionné t est comparé avec les tuple de U_1 et à la fin il est ajouté au résultat Out_1. Ensuite si U_1 n'est pas vide, C_1 prend sa valeur(C_1=U_1)et le traitement est répété jusqu'à ce que U_1 soit vide à la fin d'une itération. Dans les phases suivantes C_2 est initialisé avec les tuples des Dt_i tels que t_i est un tuple de Out_1 et on recommence.

Il est possible d'utiliser des contraintes d'intégrité sur le résultat de l'opérateur winnow. Ils existent deux types de contraintes locales et globales. Les premières imposent des conditions sur le contenu d'un tuple vu que les contraintes globales concernent un sous-ensemble de tuples. Les contraintes locales sont faciles à évaluer et sont facilement remplaçables par une sélection sur le

résultat de l'opérateur. L'expression des préférences extrinsèques avec l'opérateur winnow, peut se faire en suivant la stratégie suivante : d'abord regrouper toutes les informations pertinentes dans une relation en se servant de requêtes de l'algèbre relationnelle ; ensuite appliquer l'opérateur winnow sur cette relation et finalement projeter les attributs intéressants.

L'opérateur winnow peut être appliquer plusieurs fois sur une relation donnée et à chaque fois le résultat représente les meilleurs tuples non-retournés jusqu'à cette étape. L'article définit aussi le weaker de l'opérateur winnow pour lequel la propriété d'asymétrie et relâchée en gardant la transitivité et l'irréflexivité. Dans le résultat du weaker les tuples qui ne sont dominés que par des tuples qu'ils dominent (tuples similaires) sont inclus aussi.

- Evaluation de la clause « **prefer »**

Le principe de base de l'évaluation de la clause prefer est que le résultat doit rester non nul. Premièrement, la requête est calculée sans cette clause et ensuite les clauses de préférence sont appliquées sur le résultat. Ces préférences sont en effet des sous-requêtes exécutées sur le résultat de la requête sans préférences. Comme nous avons vu précédemment, le premier cas possible est l'évaluation de préférences imbriquées. Dans ce cas, les préférences peuvent être vues comme des filtres appliqués dans l'ordre de leur apparition dans la requête. Si après l'application d'un tel filtre le résultat est nul, il n'est pas pris en compte et l'évaluation du reste des clauses continue. Dans le deuxième cas, lorsqu'il s'agit de préférences de la même importance, le résultat (non nul) est formé par les éléments qui satisfont le plus grand nombre de clauses « prefer ».

- L'implémentation des fonctions de préférence

Dans ce paragraphe nous allons expliquer la manière de calculer les fonctions et les opérateurs permettant l'expression des préférences. A l'exception de la similarité, tous les autres opérateurs sont implémentés dans le langage Preferences SQL.

1. La similarité

Une des fonctions clés lorsqu'il s'agit de personnalisation est la **similarité** entre deux attributs ou deux éléments qui permet l'évaluation d'opérateurs approximatifs et de relâcher certaines contraintes afin d'agrandir l'espace de recherche des requêtes parce que souvent l'intention de l'utilisateur peut être floue. La notion de similarité est souvent associée à la notion de distance entre deux éléments. Bradley 00] distingue la distance entre des valeurs numériques qui est égale à la valeur absolue de la différence entre ces valeurs de celle entre des valeurs symboliques. Pour définir la distance (similarité) entre deux valeurs symboliques, les auteurs de l'article se servent de domaine d'ontologie sous forme d'arbres. Ils proposent ainsi deux approches :

- Le matching hiérarchique où tous les descendants d'un nœud ont une distance égale à zéro avec ce nœud.
- Concept de proximité où la distance entre deux nœuds est égale au nombre de nœuds les séparant.

Une fois la similarité entre deux valeurs simples définie, l'article introduit la similarité entre deux éléments complexes comme la somme pondérée des similarités des attributs de ces éléments (Figure 9).

Similarité(t,j) = $\sum_i$Similarité(ti, ji)*wi **où**

t et j sont des éléments complexes,

t_i,j_i sont les i-èmes attributs de t et j et

Figure 9: Formules de similarité entre éléments complexes

Pour la suite des opérateurs, nous allons utiliser le formalisme d'expression des préférences décrit dans [Chomicki 02], [Kießling 01] et [Torlone 02], basé sur des relations binaires de préférence. Les relations binaires de préférence sont définies par des formules de préférence. Une relation est un ordre partiel strict si elle possède les propriétés d'irréflexivité, d'asymétrie et de transitivité. Si une formule de préférence est définie en utilisant une formule de préférence intrinsèque, ses propriétés d'irréflexivité, asymétrie et transitivité peuvent être vérifiées en temps polynomial. Une formule de préférence est une formule de premier ordre qui définit une relation de préférence. Pour une relation de préférence P et deux éléments x et y, **x <P y** est interprété comme y est préféré devant x selon P. En utilisant l'ordre de préférence des éléments, il est possible de définir le **graphe de préférence G** où les nœuds sont les éléments et les arcs expriment les relations entre les éléments. Dans ce graphe y est préféré devant x si y est le prédécesseur de x. Les nœuds de G qui n'ont pas de prédécesseurs sont des éléments maximaux (de niveau 1) et ceux qui n'ont pas de successeurs sont des éléments minimaux. De cette façon, on dit qu'un élément x est de niveau i si le chemin le plus long entre x et un nœud maximal passe par i-1 nœuds. Deux éléments son indifférents si aucun des deux ne domine l'autre (il n'existe pas de chemin dans le graphe entre les deux) et sont similaires si la préférence est vérifiée dans les deux sens (il existe un chemin dans les deux sens).

Soient A un attribut d'un schéma d'un ensemble d'éléments, dom(A) le domaine des valeurs de A, POS-set et NEG-set des ensembles de valeurs de dom(A) et x et y deux éléments. Les préférences non-numériques décrites par [Kießling 01] sont la préférences positive, préférence négative, préférence positive/négative, préférence positive/positive et préférence explicite.

- La préférence positive (POS) est définie par rapport à un ensemble de valeurs (POS-set) d'un attribut A, donné en paramètre. De cette façon l'utilisateur exprime sa préférence des éléments qui ont comme valeur de l'attribut A, une des valeurs de l'ensemble POS-set. De façon formelle ceci est exprimé par :

P est une préférence positive i.e. P = POS(A, POS-set{V1,…, Vm}) si x <P y ssi

x.A $\notin$POS-set et y.A $\in$ POS-set

Ceci se traduit comme l'élément y est préféré devant l'élément x ssi la valeur de l'attribut A de y appartient à l'ensemble de valeurs et celle de x non.

- Comme la préférence positive, la préférence négative (NEG) est définie par rapport à un ensemble de valeurs NEG-set donné en paramètre. La seule différence est le fait que cette fois les éléments préférés sont ceux dont la valeur de l'attribut A n'appartient pas à cet ensemble :

P est une préférence négative i.e. P = NEG(A, NEG-set{V1,…, Vm}) si x <P y ssi

x∈NEG-set et y∉ NEG-set

- Pour la préférence positive/négative (POS/NEG), deux ensembles de valeurs pour un même attribut A sont donnés en paramètres : un ensemble de valeurs positives POS-set (préférées) et un autre de valeurs négatives NEG-set. Cette fois-ci les éléments préférés sont ceux dont la valeur de l'attribut appartient à l'ensemble positif, puis ceux dont la valeur de A qui n'appartient pas à l'ensemble négatif et finalement les élément les moins appréciés sont ceux dont la valeur de A est une des valeurs de NEG-set :

P est une préférence positive/négative i.e. P = POS/NEG(A, POS-set{V1,…, Vm) ; NEG-set{Vm+1,…, Vm+n}) si x <P y ssi

(x∈NEG-set et y∉ NEG-set) ou (x∉NEG-set et x∉POS-set et y∈POS-set)

- La préférence positive/positive (POS/POS) prend en paramètre deux ensembles de valeurs de A préférées mais le premier de ces ensembles est prioritaire par rapport au second. Dans ce cas, les éléments préférés sont ceux dont la valeur de l'attribut appartient au premier ensemble positif, puis ceux dont la valeur de A appartient au second ensemble positif et enfin les élément les moins appréciés sont ceux dont la valeur de A n'appartient à aucun des deux ensembles :

P est une préférence positive/positive i.e. P = POS/POS(A, POS1-set{V1,…, Vm) ; POS2-set{Vm+1,…, Vm+n}) si x <P y ssi

(x∈POS2-set et y∈POS1-set) ou (x∉POS1-set et x∉POS2-set et y∈POS2-set) ou

(x∉POS1-set et x∉POS2-set et y∈POS1-set)

Les quatre préférences décrites plus haut peuvent être utilisées dans le cas où on voudrait spécifier des listes de valeurs préférées ou lorsqu'on connaît les valeurs des caractéristiques recherchées.

- La préférence explicite (EXPLICIT) est définie par un ensemble de couples de valeurs EXPLICIT-graphe. Les couples de cet ensemble forment un graphe de préférence qui définit un ordre partiel strict. Si (Vali,Valj)∈ EXPLICIT-graphe l'élément y qui a comme valeur Vali de A est préféré devant x ayant comme valeur Valj (x <E y). Dans ce cas on dit que P est une préférence explicite P = EXPLICIT(A, EXPLICIT-graphe{(Val1,Val2),(Val2,Val3),(Val2,Val4),…}) si x <P y ssi

x <E y ou (x∉Graphe et y∈Graphe)

Un deuxième type de préférences décrites dans par [Kießling 01] sont les préférences numériques qui sont le plus souvent associés à la comparaison de valeurs numériques. En présence d'opérateurs calculant la similarité entre les éléments, il est possible d'appliquer ces opérateurs aussi à des valeurs textuelles. Les préférences présentées sont : autour de, entre, plus petit et plus grand.

- La préférence autour de (AROUND) exprime l'intérêts de l'utilisateur vers les éléments dont la valeur de l'attribut A est la plus proche possible de la valeur donnée en paramètre. Pour le calcul de la distance, les auteurs de l'article prennent un fonction de similarité ce qui permet d'utiliser la préférence autour de pour des chaînes de caractères. De façon formelle cette préférence est définit par :

Soit z une valeur du domaine de A. P = AROUND(A,z) si x <P y ssi

Similarité(y, z)>Similarité(x, z). En d'autres termes on choisit les éléments dont la valeur pour l'attribut A est la plus proche de z.

- La préférence « entre » (BETWEEN) comme son nom l'indique permet à l'utilisateur d'exprimer ses préférences pour les éléments dont la valeur pour un attribut donné A est comprise dans une intervalle :

Soient max et min deux valeurs du domaine de A telles que max>min. Dans ce cas P = BETWEEN(A, [min, max]) si x <P y ssi

distance(x, [min, max]) < distance(y, [min, max]) où la fonction de distance est définie comme suit :

distance(v, [min, max]) = 0 si v∈[min, max] ; (min – v) si v<min et (max –v) si v>max

- Les préférences plus petit (LOWEST) et plus grand (HIGHEST) cherchent la plus petite ou la plus grande valeur d'un attribut d'un ensemble d'éléments.

P = LOWEST(A) si x <P y ssi x>y

P = HIGHEST(A) si x <P y ssi x<y

Dans certains cas, il est possible que l'utilisateur traduise son intention par un opérateur, mais que l'interpréteur ait besoin de substituer cet opérateur par un autre ou par une combinaison d'opérateurs afin de répondre à la requête de l'utilisateur. [Chomicki 02] étudie trois types de composition entre des opérateurs de préférence : composition booléenne, fermeture transitive et composition par priorités. [Kießling 01] donne des expressions logiques de ces compositions (Figure 10 et Figure 11). La composition de préférences de la même importance est appelée préférence de Pareto et celle de préférences hiérarchiques préférences prioritaires. Les compositions booléennes décrites sont l'intersection, l'union disjointe et la somme linéaire. Selon la composition booléenne, l'union ne préserve que la propriété d'irréflexivité, l'intersection les préserve toutes (irréflexivité, transitivité et asymétrie) et la différence ne préserve pas la transitivité des opérateurs de préférence. Une relation de préférence r^* est définie comme étant la fermeture transitive d'une fonction r comme :

$t_1 r^* t_2$ ssi t_1 rn t_2 pour un n>=0 où $t_1 r_1 t_2 = t_1 r t_2$ et $t_1 r_{n+1} t_2 = \exists t_3 / (t_1 r t_3$ et $t_3 r_n t_2)$

La composition de deux relations de préférence r_1 et r_2 notée $r_{1,2}$ s'écrit :

$t_1 r_{1,2} t_2 = t_1 r_1 t_2$ ou (t_2 non(r_1) t_1 et $t_1 r_2 t_2$).

Une relation de préférence composée préserve les propriétés d'irréflexivité et asymétrie, mais pas forcement celle de transitivité.

Soient P1 et P2 deux préférences P1=(A1, <P1) et P2=(A2, <P2) où

A1, A2 sont deux attributs avec leurs domaines de définition dom(A1), dom(A2) et

X = (x1, x2) , Y = (y1, y2) deux tuples tels que x1,y1∈ dom(A1) et x2, y2∈ dom(A2)

La préférence de Pareto P = P1⊗P2 est définie par X < P1⊗P2 Y ssi

(x1 <P1 y1 ∧(x2 <P2 y2 ∨ x2 = y2)) ∨ (x2 <P2 y2 ∧ (x1 <P1 y1 ∨ x1 = y1))

Préférence prioritaire P = P1&P2 est définie par X < P1&P2 Y ssi

x1 <P1 y1 ∨ (x1 = x2 ∧ x2 <P2 y2) en considérant que P1 est plus importante que P2

Figure 10 : Expressions des compositions de préférences prioritaires

Si P1 et P2 sont disjointes et dom(A1) ∩ dom(A2) = ∅ et pour un nouvel attribut A dont le domaine est : dom(A) = dom(A1) ∪ dom(A2)

La somme linéaire P = (A, <P1⊕P2) est définie par x <P1⊕P2 y ssi

x <P1 y ∨ x <P2 y ∨ (x∈dom(A2) ∧ y∈ dom(A1))

Si P1 et P2 sont définies sur le même attribut A, alors :

Intersection des préférences P = (A, <P1♦P2) est définie par x <P1♦P2 y ssi

x <P1 y et x <P2 y

Si les deux préférences P1 et P2 sont disjointes, alors

l'union disjointe P = (A, <P1+P2) est définie par x <P1+P2 y ssi

x <P1 y ∨ x <P2 y

Figure 11: Expressions des compositions de préférences booléennes

Les approches qui proposent des extensions des langages existants pour supporter la notion de personnalisation n'intègrent pas la notion de profil. Leur syntaxe est figée et ne permet pas à l'utilisateur de modifier les implémentations des opérateurs et des fonctions utilisés.

3.5 Sécurité des données et des processus

La nature personnelle des informations du profil nécessite leur sécurisation. La sécurisation des données se fait sur trois niveaux : le contrôle de l'accès aux informations du profil, la sécurisation des échanges des données et la confidentialité de l'identité de l'utilisateur. Un des paramètres qui jouent un rôle très important pour la sécurité des données est le support utilisé pour le stockage et le traitement des informations. [Bouganim 02] propose un modèle où l'utilisateur communique avec le serveur via une carte à puce qui joue le rôle de médiateur entre le client et le serveur.

Comme la sécurisation des échanges est liée à des méthodes de cryptage complexes et la confidentialité de l'identité est en rapport avec les méthodes d'exécution des requêtes par le système, on va s'intéresser uniquement aux droits d'accès aux informations du profil. D'après [Baraani-Dastjerdi 96], ils existent trois approches d'expression des droits d'accès : **DAC** (Discretionary Access Control), **MAC** (Mandatory Access Control) et **RBAC** (Rule Based Access Control). La méthode DAC est basée sur les permissions d'accès aux fichiers usuels de lecture (r), d'écriture (w), d'exécution (x), de création etc. De cette façon, on peut exprimer les droit d'accès des utilisateurs de façon explicite sous forme de matrice. Sur l'exemple de la figure 12, on définit la matrice des droits d'accès d'un utilisateur propriétaire du profil et d'un serveur de filtrage collaboratif au profil de l'utilisateur qui est composé de deux catégories sauvegardées dans des fichiers différents. On voit que le propriétaire du profil a tous les droits vu que le serveur ne peut que lire les informations des centres d'intérêts.

Utilisateurs\Catégories du profil	Centres d'intérêts	Identité
Propriétaire	rwx	rwx
Serveur1	r	-

Figure 12 : Exemple de matrice des droits d'accès

L'approche MAC s'appuie sur le modèle de niveaux de sécurité. La technique consiste à attribuer aux informations et aux utilisateurs des niveaux de sécurité. De cette façon, seuls les utilisateurs dont le niveau de sécurité est supérieur ou égal ou niveau des données peuvent les accéder. Finalement, RBAC consiste à définir des rôles permettant d'accéder aux données et d'attribuer un rôle à chaque utilisateur. Plusieurs utilisateurs peuvent avoir le même rôle et par conséquent les mêmes droits d'accès. Sur l'exemple de la figure 13, on voit la définition de trois rôles dans un milieu hospitalier. Le patient peut accéder toutes les informations le concernant, le docteur a accès aux informations de ses patients et le chercheur juste aux informations qui ne concernent pas l'identité des patients.

Rôle	Informations accessibles
Patient	Toutes les données le concernant
Docteur	Toutes les données de ses patients
Chercheur	Age, genre, données cliniques

Figure 13 : Exemple de rôles dans un milieu hospitalier

La prise en compte des préférences de l'utilisateur pose des problèmes à différents niveaux de l'élaboration d'un système d'information. Tableau 2 fait un récapitulatif des problèmes qui surviennent lors de la modélisation d'un tel système capable de prendre en charge les préférences d'un utilisateur.

Nous venons de voir l'état actuel de la personnalisation. La personnalisation est abordée principalement dans le domaine de la recherche d'information. Les services, qui offrent un accès aux données en tenant compte des préférences des utilisateurs, utilisent des techniques de data mining connus et ne garantissent pas la satisfaction du besoin des clients. Les systèmes de personnalisation actuels, permettant d'exprimer des préférences, ne sont pas capables d'intégrer la notions de profil aux requêtes des utilisateurs et ne permettent que l'expression des préférences par des opérateurs et des fonctions statiques. Le contenu des profils n'est pas normalisé et chaque approche définit son propre modèle en fonction des besoins de l'application qui l'utilise. Afin de comprendre ces problèmes, nous allons présenter un modèle générique de profil qui englobe tous les aspects de la personnalisation et qui définit les relations entre les paramètres du profil.

4.Modélisation générique des profils

Comme nous avons vu précédemment, le contenu du profil d'un utilisateur varie selon les approches et les applications. Les approches existantes répondent partiellement aux questions liés à la personnalisation, mais il manque un modèle donnant une vision globale sur tous les aspects de la prise en compte des préférences des utilisateurs. Notre but est d'avoir une vision globale sur les paramètres qui interviennent dans la personnalisation ce qui nous donnera une compréhension globale des problèmes liés à l'impact de la personnalisation sur un SGBD. Cette vision globale servira comme modèle décisionnel qui en fonction des besoins d'une application sera capable de déterminer les paramètres nécessaires et les techniques les mieux adaptées pour sa réalisation.

4.1 Contenu des dimensions du modèle de profil

Dans la littérature de la personnalisation on trouve un certain nombre d'articles qui proposent des modèles de profil utilisateur ([Bradley 00], [Mobasher 00], [Shearin 01], [Jung 02], [Crabtree 98] etc.). Ils existent également plusieurs systèmes sur Internet qui traitent la personnalisation ([Gauch 99] présente une cinquantaine de ces systèmes avec les techniques utilisées et les informations

traitées). Malheureusement la plupart de ces profils traitent un cas particulier et sont difficilement réutilisables dans d'autres applications.

Modèle de profil

Domaine d'intérêts

- Mots clés
- Requêtes

Ontologies

- Définitions des termes
- Définitions des opérateurs

Données personnelles

- Identité
- Données démographiques
- Contacts

Qualité

- Qualité du processus
- Qualité des données
- Qualité du contenu
- Qualité du fournisseur

Customisation

- Préférences de livraison
- Codage
- Mise en page

Sécurité

- Sécurité des données
- Sécurité des requêtes
- Sécurité de l'identité

Retour du client

- Implicite
- Explicite

Divers

Figure 14: Structure d'un profil

En s'appuyant sur le modèle décrit par [Amato 99] on propose un profil générique englobant tous les paramètres que les applications de personnalisation utilisent. Dans notre approche les données du profil sont réparties selon huit catégories (dimensions) : **Domaine d'intérêts, Ontologies, Données personnelles, Qualité, Customisation, Sécurité, Retour du client et Divers** toutes complémentaires au niveau de leur contenu (Figure 14). Les parties communes aux deux modèles sont les données personnelles, les données de comportement de l'utilisateur (appelées feedback) et les données de sécurité. La partie des données de livraison correspond à la sous-catégorie préférences de livraison dans la customisation . Finalement, les trois sous-parties des données collectées du modèle proposé par [Amato 99] ont été incluses dans des catégories séparées :

- Le contenu : le sujet et les termes portant sur le contenu des documents font partie des mots clés du domaine d'intérêts ; les langues et les explications de ces termes correspondent aux définitions des termes des ontologies
- Les données de la sous-partie structure des documents font partie des informations du codage et la mise en page de la catégorie customisation
- Les sources sont explicitées dans la dimension portant sur la qualité des données.

Dans notre approche on considère que le Domaine d'intérêts et les Ontologies représentent le noyau du profil. Grâce à ces deux dimensions l'utilisateur a la possibilité de décrire le contenu des éléments qu'il recherche. Comme le profil doit exprimer la partie stable des préférences, on estime que la présence d'un profil de l'utilisateur implique qu'il cible un sujet précis faisant partie de ses centres d'intérêts. Les autres dimensions permettent d'ajouter des critères supplémentaires aux requêtes du client, mais ne peuvent pas former une demande d'information complète. Ces deux dimensions contiennent les informations minimums pour qu'une requête ait un sens. Une fois les centres d'intérêts de l'utilisateur et les ontologies définis on complète le contenu du profil avec les données des autres catégories qui sont spécifiques pour un domaine donné.

4.1.1 Le domaine d'intérêts

Cette dimension du profil est destinée à exprimer les centres d'intérêts de l'utilisateur. Ceci peut être fait par des **mots clés** ou par **des requêtes**. Les mots clés représentent des mots ou des phrases qui décrivent les informations recherchées. Ils définissent de façon simple le contenu ou le domaine des éléments ciblés. Dans cette catégorie on met toute description textuelle des objets ciblés. Il est possible que dans certains cas les objets soient des suites de caractères comme par exemple des chemins de navigation et dans ce cas ces chemins seront stockés sous forme de mots clés. Un autre exemple d'attribut faisant partie de cette catégorie est **le sujet** des éléments recherchés. Ce sujet peut être un sujet général d'un texte (par exemple la personnalisation de l'information, l'art grecque etc.) ou le sujet d'une ensemble recherché (par exemple musique classique, annonces de travail etc.). Comme on a vu précédemment, dans certains cas le profil de

l'utilisateur est constitué par des éléments de l'espace de recherche. Comme la plupart de ces approches s'intéressent aux caractéristiques des objets que l'utilisateur juge comme pertinents ou non pertinents, ces **caractéristiques** seront également considérées comme des mots clés. Certaines approches associent aux mots clés des attributs exprimant l'importance du mot (dans le cas de mots clés leur fréquence) ou la pertinence de l'objet (dans le cas où le profil contient des éléments de l'espace de recherche attribut pertinent ou pas). Ces informations font également partie des données de cette catégorie. La manière de stocker des mots clés (paragraphe 3.3.1) va dépendre de l'application pour laquelle ils vont être utilisés.

Une autre représentation des centres d'intérêts de l'utilisateur est l'utilisation de requêtes qui ont pour but de restreindre l'espace de recherche et de faire une première sélection d'éléments potentiellement intéressants. La notion de requêtes décrivant les centres d'intérêts d'un utilisateur est utilisée dans le sens d'une vue qui décrit la partie statique des requêtes de l'utilisateur. Elle contient la sélection simple d'attributs des tables et des agrégats fait sur ces attributs. Ceci peut être vu comme une présélection qui réduit la masse d'informations à prendre en compte. Par conséquent toute requête exécutée par le client sera enrichie avec les prédicats des requêtes dans son profil ou sera exécutée sur le résultat des requêtes dans le profil. Le but visé est d 'épargner à l'utilisateur la spécification des prédicats communs à toutes ses demandes d'informations. Ceci suppose que chaque requête (R) peut être décomposée en deux parties : une partie invariante commune à toutes les requêtes (Rc) et une autre propre à la demande de l'utilisateur (Rp). Donc R = (Rp•Rc) (S) ou R = Rp(Rc(S)) avec S un ensemble de sources d'informations et où (Rp•Rc) est la composition de deux requêtes en une seule. Chaque résultat envoyé à l'utilisateur doit satisfaire les conditions de sélection des deux sous requêtes Rp et Rc.

4.1.2 Les ontologies

Souvent les termes que l'utilisateur emploie pour décrire ses centres d'intérêts ne sont pas suffisants ou ont plusieurs sens (ex. JAVA : langage de programmation et danse). En plus, si on autorise l'utilisation d'opérateurs externes la manière de calculer ces opérateurs doit également être personnalisée. Pour résoudre ces problèmes dans la partie des ontologies du profil l'utilisateur peut **définir les termes** et **les opérateurs employés**.

Un des problèmes les plus importants de la recherche d'information est la grande quantité de données dans les résultats des requêtes due à l'utilisation de termes trop générales. Pour résoudre ce problème les termes employés dans les requêtes doivent être explicités de façon à donner aux mots un sens unique. Lorsqu'une requête devient trop spécifique on risque d'avoir des résultats nuls et donc de décevoir l'utilisateur. Dans ce cadre on utilise des requêtes approximatives dont le but est d'élargir l'espace de recherche. Ces requêtes sont basées le plus souvent sur des calculs de similarité entre les termes et nécessitent une connaissance linguistique. Pour répondre aux problèmes cité plus haut, cette sous-catégorie comprend des synonymes des mots, leurs traductions dans les langues que l'utilisateur comprend et le sens qu'il donne à chacun de ces

termes appelé plus souvent « contexte ». La notion de contexte est très importante pour la réduction de l'espace de recherche. Le sens des termes doit être correcte sinon les résultats renvoyés à l'utilisateur en présence de son profil risquent d'être tous inintéressants. Ce contexte peut avoir également un impact sur l'évaluation des opérateurs de la requête. Par exemple si on a un profil d'un commerçant son intention d'acheter un produit à un prix autour de X euro sera interprétée de façon différente que lorsqu'il vend un produit dont le prix est autour de la même somme. On retrouve dans la définition des termes des structures linguistiques comme des arbres d'ontologies permettant de définir les relations entre les termes.

Dans la partie de définition des opérateurs, on met la définition des outils nécessaires pour décrire le sens des expressions de préférence du client. Parmi les opérateurs on retrouve les opérateurs de base (égalité, supériorité, infériorité etc.), mais aussi des opérateurs plus complexes (Similaire, Environ, Entre, Meilleur que, Le plus grand, Le plus petit, Préféré devant, etc.). Ces opérateurs complexes peuvent être présentés par des formules de préférence où des fonctions l'expression desquelles dépend du contexte d'utilisation et de l'utilisateur. Dans la plupart des approches présentées dans l'état de l'art les opérateurs et les fonctions supportés par les langages ont des interprétations prédéfinies et par conséquent ne sont pas flexibles par rapport aux exigences des utilisateurs. Pour cette raison dans cette catégorie du profil on va définir ou surcharger les implémentations des opérateurs utilisés par les clients dans leurs profils. Il est également possible d'avoir plusieurs interprétations pour un même opérateur en fonction de la situation dans laquelle il est utilisé. Une fois les opérateurs définis il est parfois nécessaire de les documenter en donnant leur complexité et leur sélectivité pour ne pas dégrader les performances du système. Dans notre approche on considère que toute requête peut être vue comme une conjonction de prédicats du type :

<Attribut> Opérateur <Attribut | Valeur>

Par exemple prenons un opérateur Meilleur(*attribut*) qu'on va supposer utiliser dans une BD relationnelle. Cet opérateur n'est satisfait que lorsque la valeur de l'attribut donné en paramètre pour un tuple est optimale vis à vis d'un utilisateur donné. Dans le cas de recherche de livres et si l'attribut donné en paramètre est prix, on peut considérer que les meilleures propositions sont celles dont le prix est le plus bas.

4.1.3 Les données personnelles

Certains systèmes ont parfois besoin de connaître mieux les utilisateurs. C'est le cas des systèmes qui cherchent à généraliser les préférences des client sur la base de critères démographiques (ex. les femmes achètent plus de vêtements que les hommes). Lorsque l'utilisateur a un profil lui permettant de faire des achats il a également besoin du numéro de sa carte bancaire ou de son adresse pour recevoir les achats. On peut considérer le profil comme un organiseur et dans ce cas il doit contenir des données sur les contacts personnels et professionnels. Pour stocker toutes les informations personnelles, dans cette dimension on retrouve

les données permettant d'identifier un client. Dans notre cas ces données sont regroupées dans quatre sous-catégories principales :

- Identité : contenant les informations permettant d'identifier la personne comme son nom ou son login.
- Données démographiques : dans cette catégorie on retrouve comme son nom l'indique des informations démographiques telles que l'adresse, l'age, le genre, la situation familiale, le travail, le montant des revenus etc. de l'utilisateur.
- Contacts : des informations sur les contacts personnels et professionnels de l'utilisateur comme numéro de téléphone, e-mail, adresses etc.
- Autres : autres informations statiques comme par exemple le numéro de la carte bancaire

Les données personnelles sont la partie statique du profil. Elles sont très peu susceptibles d'évoluer dans le temps et ne demandent pas de mise à jour automatique par le gestionnaire du profil. Dans plusieurs approches ces données ne jouent pas un rôle dans le processus de recherche d'information, mais servent comme monnaie d'échange contre les services de personnalisation. C'est le cas des systèmes de vente sur Internet qui demandent aux utilisateurs de fournir des formulaires de données personnelles qui sont ensuite utilisées pour faire des statistiques pour mieux cibler les clients.

4.1.4 La qualité

Un des facteurs clé de la personnalisation est la qualité des informations délivrées. Les autres dimensions du profil portent sur le contenu des éléments, mais n'offrent pas la possibilité d'exprimer des préférences extrinsèques comme l'origine des informations, leur précision, le temps de réponse toléré ou l'ordre des préférences. Dans cette dimension on donne la possibilité aux utilisateurs de décrire leurs préférences sur la **qualité du processus** d'exécution des requêtes et sur la **qualité des données délivrées**. Dans la partie *qualité du processus* du profil l'utilisateur exprime ses exigences au niveau du fonctionnement du processus de personnalisation. Une des métriques standard de mesure de la qualité d'un processus est le **temps de réponse** qui est le temps nécessaire au processus d'exécuter la requête et d'afficher le résultat. Cet attribut du profil joue un rôle capital pour le choix des processus d'exécution des requêtes et de livraison des résultats. [Mobasher 00] propose d'autres expressions des performances d'un processus de personnalisation comme **le pourcentage moyen de visites** (WAVP), **la précision**, **le coverage**, **la mesure F1** et **la mesure R** (Figure 15). Le pourcentage moyen de visites est la probabilité qu'un utilisateur qui visite un élément du résultat visite aussi les autres. Cette mesure qualifie la capacité du processus de prédiction dans les systèmes collaboratifs et ceux d'aide à la navigation, mais ne porte aucune information sur l'utilité et la pertinence des éléments. La précision est le taux de résultats pertinents parmi les résultats retournés tandis que le coverage est la capacité du moteur de produire touts les résultats pertinents. La notion de coverage n'a pas de sens si l'ensemble d'éléments ne peut pas être défini ce qui est le plus souvent le cas en raison de la trop grande quantité d'informations sur Internet aujourd'hui. La mesure F1 peut être vue comme une mesure de similarité entre l'ensemble de recommandations et le reste de la session(session-la fenêtre). Et finalement la mesure R exprime la possibilité du système de couvrir les intérêts des utilisateurs avec un petit nombre

d'éléments recommandés. Cette mesure est sensible à la variance de la taille de la fenêtre de prise en considération.

Précision = (Nombre d'éléments pertinents retournés) / (Nombre total d'éléments retournés)

Coverage = (Nombre d'éléments pertinents retournés) / (Nombre total d'éléments pertinents)

F1 = 2*Coverage*Précision / (Coverage+Précision)

R = Coverage / (nombre d'éléments retournés)

Figure 15: Formules de mesures de la qualité du processus

La qualité des données recherchées est une des parties les plus importantes du profil. Elle fixe les paramètres selon lesquels les préférences de l'utilisateur sont exprimées et définit leur ordre d'importance. On retrouve ici deux sous-parties caractérisant **la qualité du contenu** et celle du **fournisseur des informations**. Le fournisseur des données est vu ici comme conteneur physique ou comme extracteur de données et pas comme processus d'exécution des requêtes de l'utilisateur. La qualité du contenu des données comprend des caractéristiques de l'ensemble de l 'élément et la définition de l'ordre des préférences du contenu. Parmi les caractéristiques de l'ensemble de l'élément on trouve la fraîcheur des informations, la précision et la justesse des données. La fraîcheur des données comprend le temps passé depuis la création de l'information ou le moment de la dernière mise à jour. De leur coté la précision et la justesse nécessitent l'avis d'un expert externe et portent des informations sur la qualité des données contenues. Il se peut qu'un utilisateur ait des préférences précises sur la présence ou l'absence d'une information. Pour prendre le coté subjectif des préférences en compte, on introduit un ordre de hiérarchisation des critères. Pour prendre en compte les préférences sur l'origine des informations recherchées la partie portant sur la qualité du fournisseur du profil va contenir une liste de sources préférées. Ces sources peuvent être des bases de données physiques interrogeables directement ou bien des moteurs de recherche collaboratifs (ex. www.google.fr, citeseer.nj.nec.com/cs, http://www.yahoo.com/ etc.) qui vont collecter les informations ce qui peut être vu comme une recommandation des sources que ces moteurs vont interroger. Dans tous les cas ils seront interrogés comme des bases de données simples. La définition d'ordre de préférence des sources est possible de la même manière que celle de la préférence du contenu des données.

4.1.5 La customisation

Cette partie du profil prend en compte les caractéristiques du système, de la plateforme du client et ses préférences de mise en page. Les informations dans cette catégorie sont classées dans trois sous-catégories : **les préférences de livraison, de codage et de mise en forme** des résultats. Les préférences de livraison comprennent le moment d'exécution de la requête, la manière de notification de l'arrivée du résultat lors de requête en mode push et le média (ex. e-mail, PDA, fax etc.) sur lequel le résultat va être fourni à l'utilisateur. Comme la bande passante et les capacités de

stockage sont une fonction du média de l'utilisateur, on retrouve ici également des informations de la taille maximale des éléments du résultat ainsi que le nombre maximal de résultats. Dans certains cas on peut prendre en compte aussi des éventuels handicaps de l'utilisateur afin de notifier l'arrivée des résultats d'une manière adaptée (ex. si l'utilisateur est malvoyant, le prévenir par un signal sonore). Les informations contenues dans la partie *codage des éléments du résultat* du profil définissent les formats des éléments du résultat compatibles avec le système de l'utilisateur et le type d'éléments ciblés. Dans l'exemple de recherche de documents textuels les formats compatibles sont définis par les extensions des fichiers et les types des documents par leurs structures (ex. article, rapport scientifique, livre, film documentaire etc.). Dans la partie mise en page on donne à l'utilisateur la possibilité d'exprimer ses préférences sur la manière dont les résultats lui seront affichés. Il pourra spécifier le nombre maximal d'éléments par page, dans le cas d'exécution de plusieurs requêtes à la fois l'ordre d'affichage des résultats ou encore la manière de regroupement des résultats (layout).

4.1.6 La sécurité

Etant donné que le profil de l'utilisateur contient des données strictement personnelles, un des problèmes majeurs qui se pose est celui de la sécurité des informations. L'utilisateur peut spécifier ses exigences du niveau de sécurité, mais l'aspect de la personnalisation doit être pris en charge par le gestionnaire du profil qui doit respecter les normes morales et législatives de sécurité imposées. Dans notre approche on identifie trois aspects de la sécurité : **la sécurité des données**, celle **de la requête** et celle **de l'identité du client**.

Dans la catégorie de sécurité des données du profil, l'utilisateur aura la possibilité d'exprimer ses exigences du niveau de sécurité des données du résultat des requêtes et celles contenu dans son profil lors d'un échange avec l'extérieur. Ce niveau de sécurité dépend des techniques employées. On retrouve ici les différentes approches de codage des données et les supports employés (ex. carte à puce, serveur sécurisé, disquette, etc). Dans le cas où les requêtes de l'utilisateur seraient effectuées par l'intermédiaire de l'Internet, l'utilisateur pourra « demander » que les échanges s'effectuent sur une liaison sécurisée et que les données soient cryptées. Il se peut que parfois les prédicats d'une requête aient un sens privé et strictement personnel et dans ce cas on a besoin de protéger l'exécution de la requête et cacher le fait qu'une requête ou une partie d'elle ait été lancée. Un exemple de requête qui nécessite que son exécution reste cachée est le cas où un chef de projet veut récupérer le nombre d'heures que chaque employé a effectué sur sa machine. Dans ce cas il va interroger tous les jours les machines des employés pour calculer la différence entre l'heure où la machine a été mise en marche et celle de fin de session. Dans ce cas il aura bien sur besoin que ce calcul reste caché pour les employés. Dans la partie de la sécurité de l'identité du client, on explique les conditions d'accès au profil. Il s'agit dans certains cas de cacher l'identité du client, d'expliciter quelles applications ont accès à quelles données et pour quels buts ces données seront employées. La définition des droits d'accès se fait par une des approches présentées dans la section 3.5. On définit également la manière dont ces droits seront attribués. Un

exemple utilisant l'approche RBAC est le cas où un serveur de filtrage collaboratif inconnu pour l'utilisateur lui demande des informations de son profil. Dans ce cas on suppose que l'utilisateur a précédemment défini un rôle SERVEUR dans son profil donnant accès aux informations de la partie mots clés des centres d'Intérêts. Dans ce cas l'utilisateur n'a pas besoin d'être notifié explicitement de cette demande d'information et l'échange des informations peut se faire de façon automatique.

4.1.7 Le retour de l'utilisateur

Dans la partie retour de l'utilisateur on retrouvera des données intermédiaires portant sur le comportement de l'utilisateur et le niveau de pertinences des résultats fournis. Cette partie du profil est chargée de gérer l'évolution et la mise à jour automatique du contenu du profil de l'utilisateur. Les informations dans cette dimension ne sont transmises vers les autres composants du profil qu'après leur validation. Cette validation peut être faite automatiquement dès que la fréquence d'apparition d'un événement dépasse un seuil donné ou manuellement par l'utilisateur en lui demandant d'intégrer ou pas une nouvelle caractéristique. On distingue deux types de données de retour du client selon la manière de collecte de ces données : **implicites** ou **explicites**. Le retour implicite de l'utilisateur sont des données fournies manuellement comme une note sur une échelle d'un élément du résultat ou une critique textuelle d'un élément. La collecte de ces informations nécessite l'implication directe du client ce qui dans la plupart des cas est gênant pour l'utilisateur. Dans la partie du retour implicite du profil, on stocke toute information susceptible de nous donner des informations sur les préférences de l'utilisateur. De telles données sont les « clickstream data », le temps passé sur une page, l'appui sur certains boutons de contrôle etc. La partie la plus importante lorsqu'on parle de retour implicite est la cause qui provoque un tel comportement. Par exemple si un utilisateur arrête l'exécution des requêtes systématiquement, ceci peut signifier qu'il ne veut attendre plus long temps ou qu'il ne désir avoir que les premiers N résultats. Chacune de ces déductions peut être envisagée et lorsqu'on s'aperçoit que ceci est répétitif, on peut soit insérer les données de préférence directement dans le profil comme nombre maximal de réponses et temps de réponse ou demander à l'utilisateur la permission de le faire.

4.1.8 Divers

Comme on a mentionné précédemment certaines applications demandent des informations spécifiques ne pouvant être incluses dans aucune des dimensions précédentes comme par exemple la bande passante attribuée au gestionnaire du profil ou le fait de déconnexion du profil pour certaines requêtes. Pour cette raison on laisse à l'utilisateur la possibilité de rajouter ces propres préférences dans cette partie du profil et de décrire leur utilisation.

La liste des attributs mentionnés dans chaque catégorie n'est pas exhaustive et dépend fortement de l'application pour laquelle le profil est utilisé. Notre but est de trouver un moyen de classification des attributs du profil de façon unique et de fixer des règles de base auxquelles tout

profil utilisateur doit obéir pour être cohérent. On laisse aux utilisateurs de l'approche la liberté du choix des composants à utiliser et leur contenu en fonction de leurs besoins.

4.2 Relations entre les paramètres du modèle

Pour expliquer les dépendances qui existent entre les paramètres du profil et entre le profil et un système de personnalisation, nous allons nous appuyer sur un système simplifié (Figure 16) de médiation. Dans les paragraphes qui suivent nous allons montrer les liens qui existent entre les paramètres des catégories du profil et montrer à quel niveau du système de médiation ces paramètres interviennent. Nous supposons que toute la gestion du profil se fait sur le coté médiateur par un module de gestion « Gestionnaire du profil » qui est relié à chaque paramètre du profil (les liens n'apparaissent pas sur la figure 16). Ce module s'occupe de la mise à jour des valeurs des attributs du profil. L'objet médiateur est considéré également comme le module d'exécution et il gère aussi la sécurité des données. Dans les sections suivantes ces « objets »(module d'exécution, gestionnaire de sécurité, requête, etc.) vont apparaître pour expliquer les relations entre eux et les données du profil. Sous requête nous comprenons la requête de l'utilisateur.

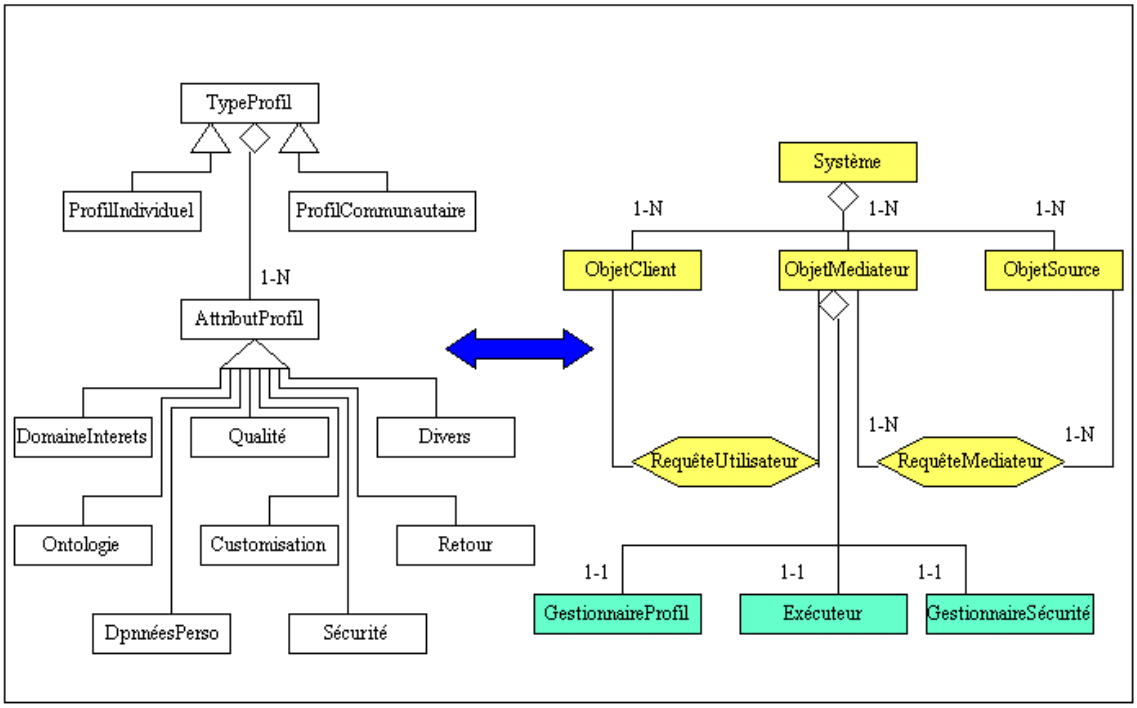


Figure 16 : Modèle de départ d'un système de personnalisation

Comme nous l'avons dit précédemment, les données personnelles n'interviennent pas dans l'évaluation des requêtes et varient très peu. Par conséquent, cette catégorie ne sera pas présentée dans cette section. La dimension Divers n'est pas bien définie et ne dépend que du

contenu que l'utilisateur lui donne donc elle ne sera pas traitée non plus. Dans la figure 17, nous avons montré les relations qui existent entre les dimension du domaine d'intérêt, des ontologies et de la qualité. Les termes employés par l'utilisateur dans son profil sont expliqués dans la partie des définitions des termes des ontologies. Pour un terme il peut y avoir plusieurs interprétations. Ces termes interviennent directement dans la requête. L'ordre des préférences des termes est définie dans la partie de la qualité du contenu des données de la dimension Qualité. De la même façon les opérateurs utilisés doivent être définis dans la partie des définitions des opérateurs des ontologies. Pour un opérateur donné il peut avoir plusieurs implémentations en fonction de la situation dans laquelle cet opérateur est utilisé. La définition des opérateurs est passé à l 'exécuteur des requêtes pour qu'il puisse optimiser l'exécution et choisir la manière de les calculer appropriée. De son coté, comme nous l'avons déjà vu, la qualité comprend la qualité des données et celle des processus. La qualité du processus comprend des valeurs de mesures de qualité que l'utilisateur voudrait obtenir de l'exécuteur des requêtes. Cette exigence du client va entraîner le choix de l'algorithme approprié par le système d'exécution. Dans cette catégorie se trouvent également la qualité des données (précision ,justesse, …) et la qualité du contenant qui sont liés aux sources des données physiques.

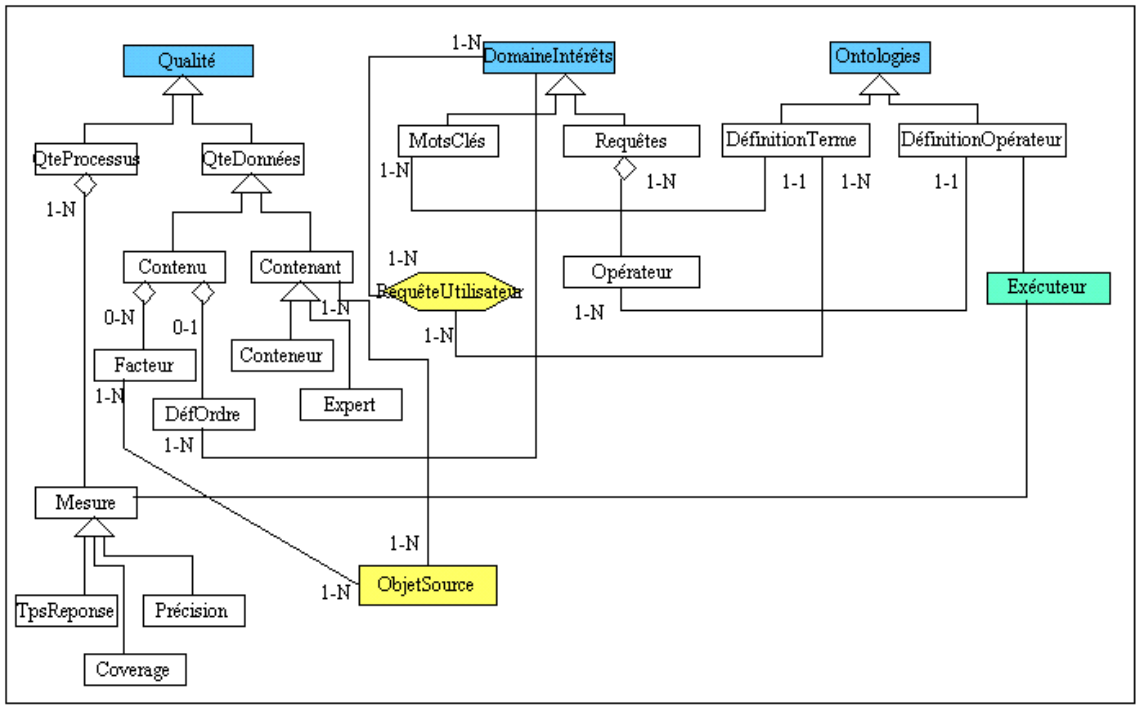


Figure 17 : Modèle des dimensions : Domaine d'intérêts, Ontologies et Qualité

Figure 18 montre les dépendances des autres dimensions : Customisation, Sécurité, Données personnelles et retour du client. Le retour du client et la sécurité des données du profil concernent toutes les autres dimensions du profil et donc les liens ne sont pas présentés pour ne pas surcharger les schémas. La catégorie sécurité définit la manière de protéger l'identité du client et le

processus d'exécution des requêtes. La catégorie Customisation intervient à deux niveaux : l'exécution de la requête (moment d'exécution et livraison) et dans l'expression de la requête par le type des éléments recherchés et leur formats.

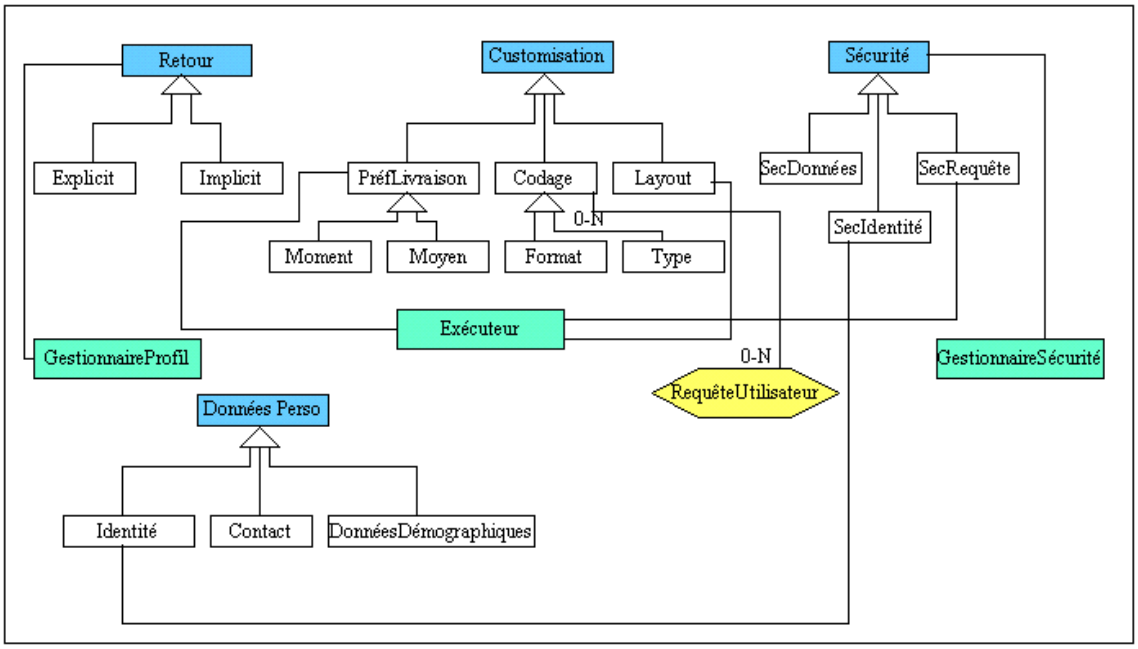


Figure 18 : Modèle des dimensions : Retour, Customisation et Sécurité

Comme nous venons de voir, le profil de l'utilisateur est une structure complexe, dans laquelle les paramètres sont liés. Les données du profil interviennent à différents endroits du cycle de vie d'une requête ce qui explique le manque d'approches de personnalisation globale dans l'état de l'art. Ce modèle est le premier pas vers la construction d'un système de personnalisation natif et pourrait évoluer lors de l'étude plus approfondie de l'impact de la personnalisation sur l'architecture d'un SGBD qui fait partie des objectifs de recherche.

5.Exemples de profils

Afin d'illustrer l'utilisation de chaque dimension du profil on propose trois exemples de profils : un chercheur informaticien, un passionné de musique rock et un médecin. Pour l'expression des mots clés on va adopter l'utilisation de listes simples, les requêtes seront exprimées en Preferences SQL. La sécurité du profil du chercheur sera exprimée par l'approche DAC, celle du passionné de musique avec la méthode MAC et pour le médecin on adoptera la technique RBAC.

5.1 Profil d'un chercheur

Dans notre exemple on va considérer que le domaine dans lequel ce chercheur travail et la personnalisation.

Domaine d'intérêt :

Mots clés

KW = {personnalisation, profil, modélisation, customisation}

Requêtes :

Pour l'expression des requêtes on va supposer que le schéma des sources de documents est connu et que ce schéma contient toutes les caractéristiques des documents (le titre, les auteurs, la date de création, des mots clé décrivant le contenu, le type du document, le lien de l'emplacement du document(URL) etc.).

Requête =

Select URL

From SOURCES

Where contenu in KW or TD

And AROUND((date_actuelle - date_création),FRAICHEUR)

And extension in FORMATS

And type in TYPES

La raison d'avoir les attributs du profil en dehors des vus est la possibilité de mise à jour des valeurs des attributs sans avoir à réécrire les requêtes.

Ontologies :

Définition des termes : (traduction en anglais)

TD = {personalization, personalisation, modelling, profile}

Définition des opérateurs :

AROUND (A, x) = VRAI si A $\in$[0,x+1/x] et FAUX sinon

= VRAI si A=x et FAUX sinon si x < 1h.

Données personnelles :

Identité :

Nom = Kostadinov Dimitre

Qualité :

Qualité du processus :

Temps de réponse maximal = 5 sec.

Qualité des données :

Qualité du contenu :

FRAICHEUR = 1 mois

Qualité du fournisseur :

SOURCES = {« citeseer.nj.nec.com », « www.google.com »}

AUTEURS: { S. Borzsonyi, D. Kossmann, K. Stocker }

Customisation :

Codage :

FORMATS = {« .ps », « .doc », « .rtf »}

TYPES = {article, rapport}

Layout :

MAX_RES = 10 résultats maximum par page

Sécurité

Utilisateur/Informations	D.Intérêts	Ontologies	D.Perso	Qualité	Cust	Sécu.
Propriétaire	**rwx**	**rwx**	**rwx**	**rwx**	**rwx**	**rwx**
Citeseer	**r**	**r**	**-**	**-**	**r**	**-**

Une fois ce profil défini, l'utilisateur va pouvoir exécuter ces autres requêtes sur le résultat de la vue contenue dans le domaine d'intérêts du profil. Les nouvelles requêtes sont plus spécialisées et doivent porter sur un sous-ensemble du domaine d'intérêt. Par exemple l'utilisateur peut demander les articles qui concernent la construction d'un profil ou l'article le plus récent etc. Dans tous les cas les nouveaux prédicats s'ajoutent aux critères de sélection de la requête du profil.

5.2 Profil d'un passionné de musique rock

Domaine d'intérêt :

Mots clés

KW = { rock, musique, chanson, nouveauté, détails croustillants}

KW2 = {Deep Purple, Led-Zeppelin, Nirvana}

Requêtes :

On suppose que l'utilisateur a deux requêtes une qui lui permet de rechercher et potentiellement d'acheter des disques et une autre lui permettent de les vendre.

Requête 1 (achat) :

```
Select URL
From SOURCES
Where contenu in KW or TD
And artiste in KW2
And LOWEST(prix)
And extension in FORMATS
And type in TYPES
```

Requête 2 (ventes) :

Pour cette requête en plus on a besoin de connaître les produits vendus. On suppose que cette requête sera exécutée par une fonction qui prend en paramètre une chaîne de caractères (Nom) correspondant au nom du produit à vendre.

```
Select achteur
From SOURCES
Where nom_produit = Nom
And HIGHEST(prix)
```

Ontologies :

Définition des termes :

TD = { music, song }

Définition des opérateurs :

Les opérateurs sont implémentés selon l'approche de Preferences SQL.

Données personnelles :

Identité :

Nom = Kostadinov Dimitre

Adresse :

5, rue de la République

75008, Paris

N°CB : 7658 9856 7654 9876

Qualité :

Qualité du processus :

Temps de réponse maximal = 5 sec.

Qualité du fournisseur :

Sources SOURCES = { www.mcm.net, www.generationrock.com, www.ebay.fr , www.amazon.fr }

Customisation :

Livraison :

Exécuter la première requête chaque jour

Codage :

Formats compatibles FORMATS = {« .mp3 », « .ps », « .doc », « .rtf »}

Type TYPES = {fichier, disque, cassette, article}

Layout :

MAX_RES = 10 résultats maximum par page

Afficher les nouveautés en rouge

Sécurité

Comme l'achat et la vente nécessitent l'échange de données confidentielles comme le numéro de carte bancaire ou l'identité de la personne, tous ces échanges doivent être sécurisés et les données cryptées. On suppose que l'utilisateur donne également le droit aux moteurs de recherche qu'il connaît de consulter et mettre à jour ses préférences sur les groupes de rock de façon à connaître les nouveautés.

Paramètre du profil	KW2	N°CB, Adresse	Autres informations
Niveau de sécurité	2	3	4

Ensuite on attribue des niveaux de sécurité aux utilisateurs :

Propriétaire du profil 4

Organisme bancaire 3

Les moteurs de recherche 2

De cette façon l'utilisateur a tous les droits, l'organisme bancaire (qu'on suppose unique pour l'utilisateur) peut voir le numéro de la carte bancaire et l'adresse pour effectuer les transactions, mais également les groupes de rock préférés et finalement les moteurs de recherche ne peuvent accéder qu'aux groupes préférées. Pour simplifier l'expression des attributions des droits d'accès on a regrouper les moteurs de recherche, mais en principe dans cette approche ceci doit être fait de façon explicite (www.mcm.net 2, www.generationrock.com 2 etc.).

5.3 Profil d'un médecin

Pour cet exemple on suppose qu'il existe une base de données d'un hôpital et que les médecins peuvent consulter et modifier les dossiers de leurs patients, de faire des prescriptions de médicaments.

Domaine d'intérêt :

Requêtes :

La première requête permet à un médecin de retrouver et afficher le dossier d'un patient dont le nom est NOM.

Requête 1 :

```
Select dossier
From SOURCES
Where nom_patient = NOM
```

Requête 2 :

La deuxième requête servira à prescrire un médicament Médicament à un patient de nom NOM :

```
INSERT INTO prescriptions VALUES (NOM, Médicament)
```

Données personnelles :

Identité :

Nom = Kostadinov Dimitre

N°Med = 3423

Contacts :

Travail : 01 45 56 76 98

Urgences : 01 43 56 87 87

Docteur Delaporte : 01 45 56 76 87

Qualité :

Qualité du processus :

Temps de réponse maximal = 1 sec.

Customisation :

Préférences de livraison :

On update dossiers.etat_patient where Numero_médecin = N°Med

Notification par SMS

Layout :

MAX_RES = 10 résultats maximum par page

Sécurité

Dans un milieu hospitalier la sécurité des informations est primordiale. Dans notre exemple on va prendre l'approche RBAC pour définir les droits d'accès. Dans la partie sécurité du profil on met un attribut rôle pour indiquer les droits d'accès.

Rôle = Médecin

On suppose qu'à l'extérieur on a une définition du rôle médecin (faite par un administrateur) qui exprime les droits d'accès d'un médecin à la base de données de l'hôpital.

Rôle = Médecin : Informations accessibles : dossiers where Numero_médecin = N°Med

Dans des situations particulières le médecin doit être capable d'accéder au dossier d'un patient (NOM)dont l'état est critique, donc on ajoute une condition d'accès supplémentaire :

On etat_patient = CRITIQUE, Rôle Médecin = dossier where Nom_patient = NOM

Dans ce cas il est nécessaire de compléter la sécurité de façon à interdire aux utilisateurs de modifier leurs rôles. Pour ce faire on va se servir de l'approche DAC en donnant juste la permission d'exécuter et consulter les rôles.

Utilisateur\Données	Sécurite.Rôle	Autre données du profil
Administrateur	rw	-
Autre	rx	rwx

Comme on voit dans ce tableau l'administrateur définit et attribue les rôles, mais ne peut pas les exécuter et les utilisateurs ne peuvent pas modifier leurs rôles.

Comme la prescription d'un médicament est aussi une information confidentielle on doit également sécuriser le processus de la requête 2 en cachant l'identité du patient.

6.Conclusion et perspectives de recherche

La personnalisation de l'information immerge comme une approche capitale dans le développement des systèmes du futur. Les problèmes apparus avec la prise en compte des préférences des utilisateurs sont encore mal compris et demandent des réflexions au niveau de l'architecture des systèmes, des langages d'expression des requêtes et les techniques de construction et mise à jour des profils. Ce sont également les trois axes de recherche principales de la personnalisation. Nos objectifs de recherche se situent au niveau du deuxième problème sur la définition d'un langage universel, ressemblant à SQL, qui permet d'exprimer tous les paramètres d'un profil ainsi que sa création. Ce langage peut ensuite s'appuyer sur les approches existantes comme une surcouche et dans ce cas une requête universelle sera décomposée en sous-requêtes et chacune de ces sous-parties sera ensuite traduite et exécutée sur les systèmes existants. L'idée est de prendre comme base Preference SQL pour utiliser les extensions qu'il introduit. Un deuxième problème est de compiler ce langage universelle pour obtenir un code exécutable sur Preferences SQL ou pour obtenir directement du SQL standard. Une fois ces problèmes traités, nous allons étudier la réécriture des requêtes en présence d'un profil. On part d'une requête écrite en SQL pour la réécrire dans le langage universel. En parallèle nous allons étudier l'impact de la personnalisation sur l'architecture des systèmes pour pouvoir implémenter des prototypes de systèmes personnelles en Bases de Données.

Bien qu'ils existent déjà plusieurs approches de construction de profils de façon automatique, mais sous la surveillance de l'utilisateur, le problème de capture des paramètres du profil reste non résolu. Les données contenues dans le profil doivent être correctes et justes car elles ont un impact direct sur les résultats qui vont être obtenus. Le fait que l'utilisateur a souvent des idées floues sur ses préférences complique l'extraction des caractéristiques pertinentes et nécessite un processus de découverte de ses préférences. Ce processus doit être fait avec l'implication minimale du coté du client et en même temps de fournir des données correctes par rapport à l'utilisateur. La construction et la mise à jour du profil sont des problèmes qui dépassent le cadre des objectifs de recherche fixés en premier temps.

L'élaboration d'un modèle de profil générique est le premier pas vers la construction de systèmes natifs de personnalisation capables de rechercher et extraire des informations de qualité selon les préférences d'un utilisateur. Dans notre approche un profil comprend huit catégories (dimensions) regroupant tous les paramètres nécessaire afin de fournir un service personnalisé. On a montré quel type d'informations sont contenues dans ces dimensions sans pour autant borner les paramètres en croyant que chaque approche ne vas prendre que les catégories qui la concernent pour y mettre les données nécessaires.

Références

[Adomavicius 01] Using Data Mining Methods to Build Customer Profiles, G. Adomavicius, A. Tuzhilin, Computer 2001IEEE pp 74-82

[Amato 99] User Profile Modeling and Applications to Digital Libraries ,Giuseppe Amato, Umberto Straccia , Proc. 3rd European Conf. Research and Advanced Technology for Digital Libraries, ECDL, 1999

[Baraani-Dastjerdi 96] Security In Databases: A Survey Study, A. Baraani-Dastjerdi, J. Pieprzyk, R. Safavi-Naini, ftp://ftp.cs.uow.edu.au/pub/papers/1996/tr-96-02.ps.Z.

[Borzsonyi 01] The Skyline Operator, S. Borzsonyi, D. Kossmann, K. Stocker, IEEE Conf. On Data Engineering, 2001

[Bouganim 02] Chip-Secured Data Access: Confidential Data on Untrusted Servers, L. Bouganim, P. Pucheral, Proceedings of the 28th VLDB Conference, Hong Kong, China, 2002

[Bradley 00] Case-Based User Profiling for Content Personalisation, K. Bradley, R Rafter, B. Smyth, Lecture Notes in Computer Science, 2000

[Bradley 99] Personalised Case Retrieval, K. Bradley, R Rafter, B. Smyth, In Proceeding of the 10th Irish Conference on AI&CS, 1999

[Bright 02] Using Latency-Recency Profiles for Data Delivery on the Web, Laura Bright, Louiqa Raschid, Proceedings of the Conference on Very Large Data Bases (VLDB), 2002

[Cherniack 03] Profile-Driven Cache Management, M. Cherniack, E.F. Galvez, M.J. Franklin S. Zdonik, International Conference on Data Engineering (ICDE), 2003, Bangalore, India

[Chomicki 02] Querying with Intrinsic Preferences, Jan Chomicki, EDBT, 2002

[Crabtree 98] Knowing Me, Knowing You: Practical Issues in the Personalisation of Agent Technology, B. Crabtree, S Soltysiak, Proc. Third International Conference on the Application of Intelligent Agents and Multi-Agent Technology (PAAM98), London, March 23-25, 1998

[Croft 01] Relevance Feedback and Personalization:A Language Modeling Perspective, W. B. Croft, St. Cronen-Townsend, V. Lavrenko, Computer Science, DELOS Workshop, 2001

[Ferreira 01] MySDI:A Generic Architecture to Develop SDI Personalised Services, J. Ferreira, A. Silva, ICEIS (1) 2001: 262-270

[Fink 00] A Review and Analysis of Commercial User Modeling Servers for Personalization on the World Wide Web, J.Fink, A. Kobsa, User Modeling and User-Adapted Interaction 10: pp. 209-249, 2000

[Franklin 03] Profile-Driven Cache Management, Mitch Cherniack, Eduardo F. Galvez, Michael J. Franklin and Stan Zdonik, International Conference on Data Engineering (ICDE), 2003, Bangalore, India. To appear

[Gauch 99] Personalization on the Web, A. Pretschner, S. Gauch, Technical report, In- formation and Telecommunication Technology Center, Department of Electrical Engineering and Computer Science, The University of Kansas, 1999

[Han 01] Data Mining: Concepts and Techniques, J. Han, M. Kamber, Morgan Kaufmann Publishers, Academic Press 2001

[Hellerstein 99] Online dynamic reordering for interactive data processing, V. Raman, B. Raman, and J. Hellerstein, In VLDB, 1999

[Joachims 97] Webwatcher: A tour guide for the World Wide Web, Joachims, T.; Freitag, D.; and Mitchell, T, A tour guide for the World Wide Web. In Proc. IJCAI-97

[Jung 02] A Formal Model for User Preference, S. Y. Jung, J-H. Hong, T-S. Kim, IEEE International Conference on Data Mining, 2002

[Kießling 01] Foundations of Preferences in Database Systems, Werner Kießling, VLDB 2002: 311-322 [DBLP:conf/vldb/Kiessling02]

[Lacroix 87] Preference:Putting More Knowledge into Queries, M. Lacroix, P. Lavency, Proceeding of the 13th VLDB Conference, Brighton, 1987

[Lieberman 95] An Agent That Assists Web Browsing, H. Lieberman, 1995 International Joint Conference on Aritifical Intelligence, Montreal, CA, 1995

[Melville 01] Content-boosted collaborative filtering, P. Melville, R. J. Mooney, and R. Nagarajan, In Proceedings of the 2001Workshop on Recommender Systems, New Orleans, LA, September 2001

[Mobasher 00] Discovery and Evaluation of Aggregate Usage Profiles for Web Personalization, B. Mobasher, H. Dai, T. Luo, M. Nakagawa, Data Mining and Knowledge Discovery, 6(1):61-82, 2002

[Oostendorp 94] A Tool for Individualizing the Web, K. A. Oostendorp, W.F. Punch, and R.W. Wiggings, 2nd International Conference on the World-Wide Web, Chicago, IL, 1994, 49-57

[Pretschner 99] Ontology Based Personalized Search, A. Pretschner, S. Gauch, Proc. 11th IEEE Intl. On Tools with Artificial Intelligence, pp. 391-398, Chicago, November 1999

[Schiaffino 00] User profiling with Case-Based Reasoning and Bayesian Networks, S. Schiaffino, A. Amandi, IBERAMIA-SBIA 2000 Open Discussion Track

[Shearin 01] Intelligent Profiling by Example, S. Shearin, H. Lieberman, MIT Media Lab, Cambridge, 2001

[Soltysiak 98] Automatic learning of user profiles-towards the personalisation of agent services, S J Soltysiak, I B Crabtree, BT Technol J. Vol 16 No 3, July 1998, pp 110-117

[Sorensen 95] PSUN : A Profiling System for Usenet News, H. Sorensen, M. Mc Elligott, CIKM'95 Intelligent Information Agents Workshop, Baltimore, December 1995

[Stewart 97] User Profiling Techniques: A Critical Review, S. Stewart and J. Davies, Proceeding of the 19th Annual BCS-IRSG Colloquium on IR, Aberdeen, Scotland, 8-9 April 1997

[Torlone 02] Which Are My Prefered Items?, R. Torlone, P. Ciaccia, Workshop on Recommendation and Personalization in eCommerce (RPEC 2002), Malaga, Spain, 2002

[Vassiliou 02] The process of personalizing web content: techniques, workflow and evaluation, Ch. Vassiliou, D. Stamoulis, D. Martakos, http://www.ssgrr.it/en/ssgrr2002w/papers/18.pdf

[Villa 01] A Framework for implicitly tracking data, R. Villa, M. Chalmers, DELOS Workshop: Personalisation and Recommender Systems in Digital Libraries 2001