Chapitre V : application pour l'assistance à la navigation sur le Web

Résumé

Pour évaluer notre modèle d'indexation et notre plate-forme à objets CBR*Tools, nous avons réalisé un assistant de navigation sur le Web appelé Broadway. Cet assistant utilise le raisonnement à partir de cas pour recommander des pages à visiter suivant le comportement d'un utilisateur courant. Broadway est accessible par un groupe d'utilisateurs et permet ainsi une collaboration indirecte en réutilisant les cas issus des navigations du groupe. Les navigations des utilisateurs sont enregistrées suivant quatre variables décrivant l'adresse, le contenu, l'évaluation explicite et le ratio d'affichage de chaque page Web visitée. Ces navigations sont utilisées pour en extraire des expériences utiles (cas). Un cas permet d'associer à un comportement (succession de pages), un ensemble de pages évaluées. Nous montrons ainsi que notre modèle d'indexation permet la modélisation de ce type de cas. Pour évaluer l'utilisation de CBR*Tools, nous montrons premièrement que, pour la réalisation de Broadway, l'expertise approfondie nécessaire est limitée à la moitié des points d'ouverture. Deuxièmement, nous montrons également l'aide apportée par CBR*Tools tant sur la modélisation que sur l'implantation, grâce à la réutilisation de son architecture abstraite et de ses composants (index, similarité). De plus, avec Broadway, nous proposons un *nouveau type d'assistants* dans lesquels les comportements des utilisateurs sont observés suivant un ensemble extensible de variables. Cette observation par variables, combinée avec notre modèle d'indexation, permet la gestion de comportements plus détaillés dans un cadre flexible et générique. Une première évaluation expérimentale de notre assistant montre l'intérêt et la faisabilité de notre approche.

Ce dernier chapitre présente l'application de notre modèle d'indexation par situations comportementales (cf. chapitre II) et l'utilisation de notre plate-forme à objets CBR*Tools (cf. chapitres III et IV) pour le système Broadway[42], notre assistant à la navigation sur le Web.

Le World-Wide-Web est un gigantesque hypermédia dans lequel trouver un document pertinent n'est pas une tâche facile malgré les outils existants pour effectuer une recherche (cf. chapitre I, §1.2). Broadway permet de suivre un groupe d'utilisateurs durant leurs navigations et recommande des pages potentiellement intéressantes en fonction de leurs comportements. Broadway utilise le raisonnement à partir de cas pour réutiliser des expériences précises issues des navigations passées du groupe. Notre approche soutient l'hypothèse suivante : *si deux utilisateurs ont navigué suivant une séquence similaire de pages, c'est qu'ils ont des intérêts similaires, nous pouvons alors proposer au second utilisateur les pages jugées pertinentes par le premier.*

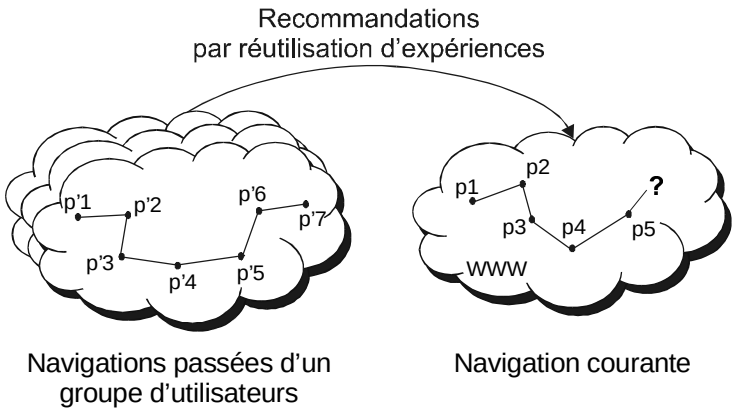

Figure V-1 : réutilisation d'expériences pour déterminer des recommandations

Nous présentons, dans la première partie, les caractéristiques générales du système Broadway : interface utilisateur, architecture et principes du cycle de raisonnement à partir de cas. Dans la deuxième partie, nous détaillons l'application de notre modèle d'indexation et l'utilisation de CBR*Tools dans le cadre de ce système. Nous faisons alors un bilan sur les qualités et les limites du modèle d'indexation ainsi que sur l'aide apportée par CBR*Tools. Enfin, dans la troisième partie, nous montrons que l'approche proposée mène à un nouveau type de systèmes d'assistance, dont Broadway est un exemple.

1 Présentation de Broadway

Broadway est un assistant de navigation répondant à des contraintes d'application précises. Après avoir introduit ces contraintes, nous présentons l'interface graphique et l'architecture répartie de Broadway. Enfin, nous donnons les principes de base du raisonnement mené par ce système pour déterminer les recommandations durant les navigations des utilisateurs sur le Web.

1.1 Contraintes de l'application

Notre objectif est de réaliser un *assistant de navigation*, c'est-à-dire un outil de navigation pour le Web qui soit capable de suivre un utilisateur, tout en essayant de déterminer automatiquement son

[42] Le nom « Broadway » est basé sur l'expression suivante : « *a BROsing ADvisor reusing pathWAYs* ».

intérêt pour lui recommander des pages à visiter. La principale contrainte de Broadway est de pouvoir *prendre en compte les historiques des actions d'utilisateur* pour mieux déterminer les recommandations. Nous pensons en effet, comme d'autres (Corvaisier *et al.,* 1997 ; Shahabi *et al.,* 1997 ; Chen & Park, 1998 ; Schechter *et al.,* 1998), qu'un état particulier de la navigation (la page courante ou une description instantanée de la navigation) n'est pas suffisant pour caractériser le comportement d'un utilisateur. C'est pourquoi nous voulons considérer les pages passées et leur ordre de consultation pour induire son intérêt implicite.

De plus, pour obtenir les recommandations et pour permettre une large utilisation du système, nous abordons cette problématique avec cinq autres exigences :

1. Broadway doit *être conçu pour le Web*, c'est-à-dire que son utilisation ne doit pas être limitée à un site Web et qu'il doit prendre en compte l'aspect dynamique de cet hypermédia dans lequel des pages peuvent être effacées ou modifiées à tout moment.
2. Broadway doit *apprendre d'un groupe d'utilisateurs* afin d'améliorer et de maintenir à jour ses connaissances lui permettant de déterminer les recommandations.
3. Broadway doit permettre *d'associer des annotations textuelles* aux pages pour décrire plus précisément leur pertinence.
4. Broadway doit *être indépendant du navigateur* pour pouvoir être utilisé sur différentes plates-formes.
5. Enfin, Broadway doit *présenter une architecture ouverte* facilitant son extension et sa modification, sachant que des évaluations en milieu réel seront menées et que des réglages seront nécessaires.

1.2 Interface utilisateur et architecture de Broadway

L'assistant de navigation Broadway peut être utilisé par un groupe d'utilisateurs qui veulent partager leurs expériences de navigation sur le Web. Chaque utilisateur a la possibilité d'utiliser son propre navigateur et peut communiquer avec Broadway au travers d'une interface graphique dédiée. Derrière cette interface, Broadway repose sur une architecture répartie pour une meilleure efficacité et pour faciliter son extension.

1.2.1 Interface utilisateur

L'utilisateur est capable d'interagir avec Broadway par deux moyens : les *barres d'outils* et le *contrôleur*. Les barres d'outils sont insérées dans chaque page visualisée par le navigateur et offrent des fonctions contextuelles. Le contrôleur est affiché dans une fenêtre séparée. Il permet à chaque utilisateur de contrôler le fonctionnement global de Broadway et de sélectionner les recommandations.

1.2.1.1 *Barre d'outils*

Une barre d'outils est insérée au début de chaque page Web chargée à partir du protocole de communication HTTP (cf. Figure V-2). Lorsqu'une page est composée de différents cadres (*frames*), une barre d'outils est également insérée en haut de chaque cadre. La barre d'outils est par défaut masquée et peut être affichée entièrement par simple clic sur un onglet dans le coin supérieure gauche de la fenêtre[43]. La barre d'outil permet ainsi d'accéder à des fonctions contextuelles à la page ou au

[43] Cette fonctionnalité n'a été testée que dans le cas du navigateur Netscape en utilisant les possibilités simples qu'il offre pour superposer différentes couches HTML.

cadre dans lequel elle est affichée, même dans le cas où l'utilisateur a besoin de charger des pages simultanément dans des fenêtres différentes.

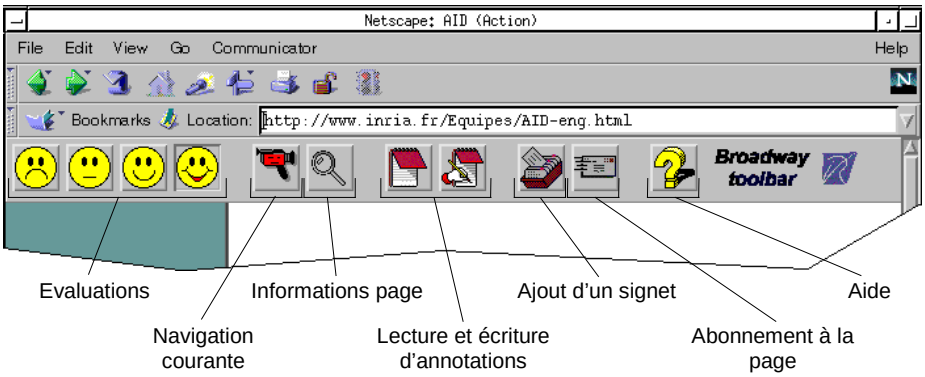

Figure V-2 : barre d'outils de Broadway

Une barre d'outils fournit un ensemble de fonctions : évaluation de la page avec quatre niveaux (pas pertinent, indifférent, pertinent et très pertinent), consultation de la navigation courante et des informations statistiques sur la page courante, écriture et lecture d'annotations textuelles, ajout de la page dans le serveur partagé de signets[44], abonnement à la page[44] (notification par mail de son changement), et enfin accès à une page d'aide pour l'utilisation de la barre d'outils. Les boutons d'évaluation permettent d'indiquer la pertinence de la page courante vis-à-vis de l'objectif d'une navigation particulière. Si l'utilisateur repasse sur une page déjà visitée, la dernière évaluation est affichée par défaut et peut être ajustée. Enfin, lorsqu'une page ayant des annotations est affichée, le bouton de lecture change d'apparence pour l'indiquer directement à l'utilisateur.

1.2.1.2 *Contrôleur*

Le contrôleur permet la configuration de Broadway ainsi que l'affichage des recommandations (cf. Figure V-3). Une navigation peut être lancée ou stoppée et correspond à une tâche de recherche d'une information sur le Web. Durant cette navigation, le calcul des recommandations peut s'effectuer à la demande ou en continu. Dans le premier mode, un bouton permet de lancer une requête. Les recommandations sont alors calculées de manière asynchrone et sont affichées dans une liste dès leur disponibilité. Dans le mode de calcul en continu, les recommandations, également calculées de manière asynchrone, sont régulièrement mises à jour. Ces mises à jour sont déclenchées par les changements de pages dans le navigateur.

Les recommandations sont présentées dans une liste donnant un pourcentage et le titre de la page (si la page n'a pas de titre, son adresse est alors affichée). Le pourcentage représente l'estimation faite par Broadway du degré de pertinence dans une certaine page vis-à-vis du comportement de l'utilisateur. Si l'utilisateur sélectionne une recommandation, la zone d'information sur la recommandation est alors mise à jour pour donner l'adresse, le titre, les mots clefs et la description de la page. Ainsi, il est possible d'analyser rapidement les recommandations pour en choisir une. Le bouton « *go* » permet finalement de se déplacer directement à la page sélectionnée qui est affichée dans le navigateur.

[44] Cette fonction n'as pas été implantée mais montre les possibilités d'extension de Broadway en cours d'étude.

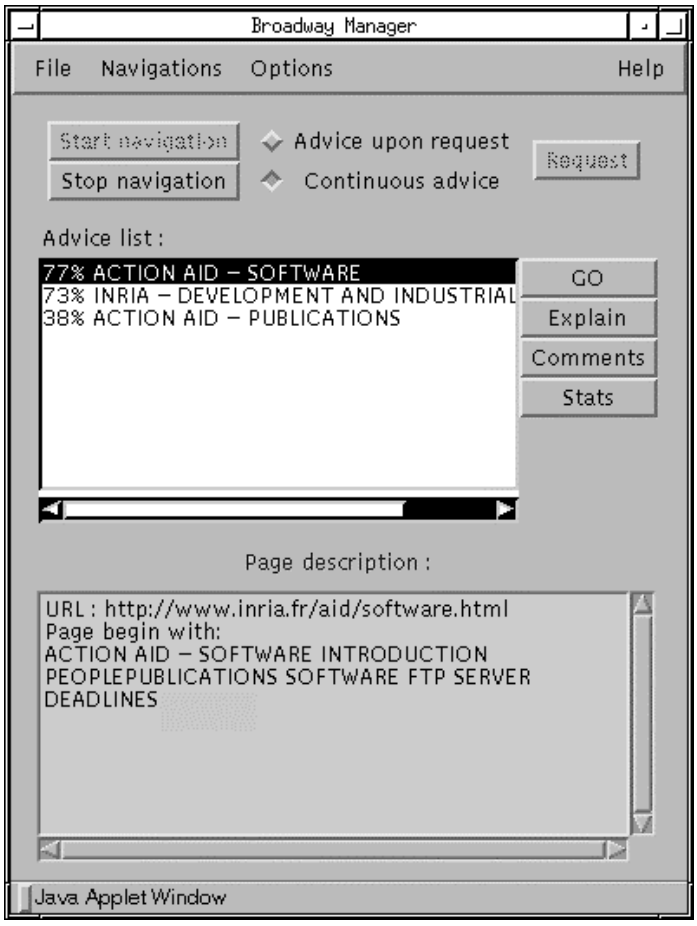

Figure V-3 : contrôleur de Broadway

1.2.2 Architecture logicielle

Nous introduisons tout d'abord l'architecture globale de Broadway qui repose sur la notion de serveur HTTP proxy. Puis nous présentons plus précisément l'architecture répartie mise en place pour une meilleure modularité des fonctionnalités. Enfin, nous décrivons les outils réalisés pour faciliter l'extension et la modification de Broadway.

1.2.2.1 *Architecture globale*

En se basant sur l'analyse donnée dans (Meeks *et al.*, 1996), il existe deux types d'architecture pour concevoir un assistant de navigation pour le Web : des transducteurs de flux (*stream transducers*) et des associés de navigation (*browsing associates*). Un transducteur est un module inséré dans les communications HTTP entre un client et le reste du Web pour analyser et modifier les flux de requêtes et de réponses. Un associé de navigation est un module autonome qui accède au Web de manière indépendante pour la réalisation d'une tâche. Notre objectif est de suivre les utilisateurs durant leurs navigations et de les assister en fonction de leurs actions. Ce type d'assistant est très dépendant du flux de requêtes et de réponses, c'est pourquoi nous avons choisi une architecture de type transducteur pour Broadway.

Plus précisément, Broadway repose sur la notion de serveur HTTP *proxy* (ou tout simplement proxy) qui est un des moyens pour réaliser un transducteur de flux. Un proxy est un serveur HTTP spécial accessible par un groupe d'utilisateurs. Un proxy attend les requêtes des clients et les transfère au besoin vers le reste du Web ; puis les réponses provenant du Web sont alors retournées vers le

client initial. Les premières utilisations des proxies étaient limitées à des fonctions de sécurité ou de mémoire cache (Luotonen & Altis, 1994). Dans le premier cas, le proxy, appelé pare-feu (*firewall*), intercepte et vérifie les communications HTTP entre le réseau intérieur ainsi sécurisé, et le réseau extérieur. Dans le deuxième cas, le proxy gère une mémoire cache des pages déjà visitées de sorte que, si un utilisateur redemande la même page, le proxy renvoie sa copie locale (si elle n'est pas obsolète) sans recharger la page à partir du serveur d'origine. Ces deux usages du proxy peuvent être combinés mais restent de bas niveau. Dans ce cadre, l'utilisation de proxies est répandue sur le Web et fait partie du protocole HTTP. Tous les navigateurs courants (par exemple, Internet Explorer et Netscape) permettent leur configuration pour utiliser de manière transparente un proxy.

Toutefois, l'intérêt de ce type d'architecture dépasse ces fonctions de bas niveau et mène à la définition de *proxies d'applications*. En effet, un proxy est un outil adapté à l'analyse transparente des communications HTTP et du contenu des pages Web visitées. De plus, le proxy est capable de modifier le contenu d'une page avant que l'utilisateur soit en mesure de l'afficher. Ces propriétés ont été étudiées dans de nombreux travaux (Brooks *et al.*, 1995 ; Luotonen & Altis, 1994) et ont mené à l'utilisation de proxies pour des applications de haut niveau comme la gestion d'un serveur d'annotations pour un groupe d'utilisateurs (Schicker *et al.*, 1996). Broadway est dans la même lignée que ces travaux. D'autres assistants de navigation ont d'ailleurs également choisi ce type d'architecture (Mladenic, 1996).

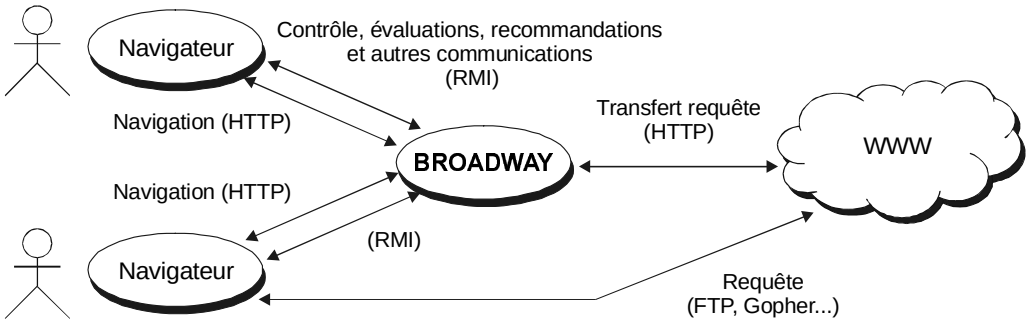

Figure V-4 : architecture globale de Broadway

Ainsi, Broadway est un système qui est intercalé entre les navigateurs et le reste du Web pour les requêtes utilisant le protocole HTTP (cf. Figure V-4). Broadway accepte de multiples connexions simultanées provenant d'un groupe d'utilisateurs. Chaque utilisateur est identifié par un nom et un mot de passe suivant le protocole d'authentification prévu dans HTTP. Au lancement d'un navigateur, une fenêtre est alors affichée à l'écran pour que l'utilisateur puisse s'identifier. Ces données sont alors stockées par le navigateur qui les transmettra avec chaque requête. Cette authentification est importante car elle permet de contrôler l'accès au système qui contient des informations privées, notamment sous la forme de navigations. De plus, ce mécanisme permet de suivre un utilisateur dans la succession de ces requêtes. En effet, HTTP est un protocole de communication *sans état*, et le proxy n'a pas de moyen par défaut pour savoir s'il traite des requêtes issues du même utilisateur ou d'utilisateurs différents. La transmission automatique et transparente des informations d'identification avec chaque requête permet alors de pallier ce problème[45].

[45] Pour ce dernier problème, une solution souvent utilisée est celle d'un *cookie* (RFC 2109, 1997). Un cookie est une information que le serveur HTTP demande de sauver dans le navigateur de l'utilisateur et qu'il peut consulter à chaque requête. Cependant, les cookies sont accessibles uniquement par les serveurs du domaine qui les a déclarés (un proxy n'est pas censé les manipuler) et les utilisateurs peuvent refuser ou effacer les cookies. D'autres solutions existent ou sont en cours d'étude : solution coûteuse de la réécriture des hyperliens (Iyengar, 1997), extension du protocole HTTP (Hallam-Baker & Connolly, 1996).

Broadway intercepte toutes les requêtes dans le protocole HTTP et permet ainsi d'analyser les différentes pages HTML chargées par les utilisateurs. Les requêtes effectuées dans d'autres protocoles (FTP ou Gopher par exemple) sont exécutées directement ou avec d'autres mécanismes indépendants de Broadway. De plus, Broadway est muni d'une mémoire cache ce qui lui donne une plus grande efficacité dans le chargement des pages pour le groupe. Enfin, Broadway gère un espace d'information local qui est accessible par le protocole de communication RMI (*Remote Method Invocation*) du langage de programmation Java (RMI, 1997). Ainsi, toutes les communications nécessaires aux barres d'outils et aux contrôleurs sont réalisées suivant ce protocole. Nous avons choisi ce protocole de communication plutôt que HTTP ou CORBA (Geib *et al.*, 1997) pour plusieurs raisons. Tout d'abord, par rapport à HTTP et l'utilisation de pages HTML, RMI permet de récupérer simplement sur les clients les objets représentant les informations requises ; ces objets peuvent alors être traités localement dans le navigateur pour améliorer l'interactivité et la personnalisation de l'affichage[46]. Puis, par rapport à CORBA qui suppose un environnement hétérogène, l'ensemble de nos composants est programmé en Java, et RMI offre des fonctionnalités spécifiques à cet environnement homogène : gestion de la sécurité lors du chargement dynamique de classes distantes et ramasse-miettes réparti par exemple. De plus, dans cet environnement, RMI est disponible en standard et reste simple d'emploi.

1.2.2.2 *Architecture détaillée de Broadway*

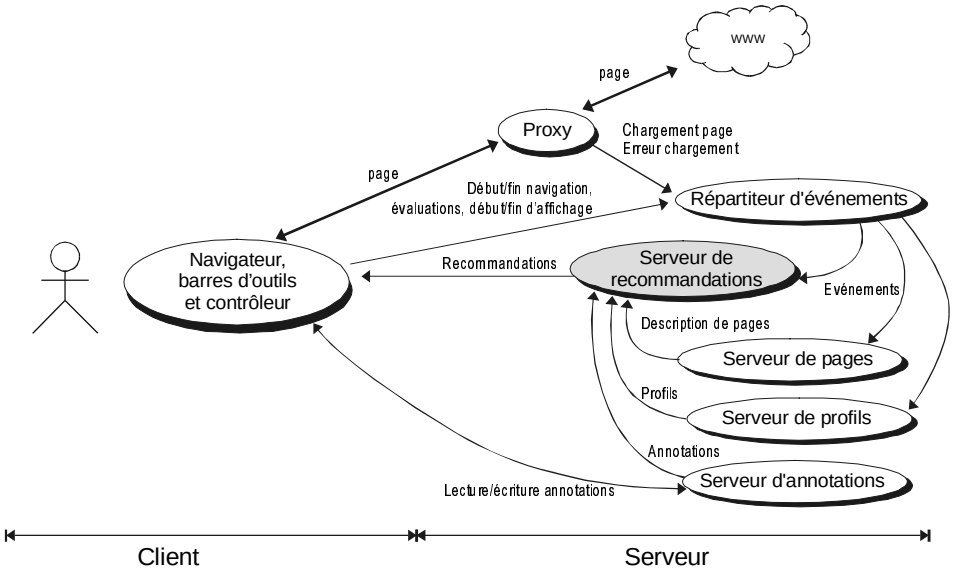

Figure V-5 : architecture détaillée de Broadway

L'architecture détaillée de Broadway présente un ensemble de modules réalisés dans le langage de programmation Java et dont la plupart peut être exécutée dans des processus différents (cf. Figure V-5). Du coté client, le navigateur de l'utilisateur abrite différents modules : les barres d'outils, le contrôleur et l'observateur. Ces modules sont des *applets*, c'est-à-dire des programmes Java qui s'exécutent au sein du navigateur dans des processus légers (*threads*) différents. Nous avons déjà décrit les rôles des barres d'outils et du contrôleur au travers de leur interface graphique. L'observateur est une applet ne possédant pas d'interface et qui est insérée comme les barres d'outils dans chaque page HTML par Broadway. Cette applet est alors activée lors du début et de la fin de l'affichage d'une page et en informe Broadway, même dans le cas où la page est chargée à partir de la mémoire cache locale du navigateur d'un utilisateur (Shahabi *et al.*, 1997).

[46] L'utilisation de RMI n'est pas obligatoire, la première version de Broadway n'utilisait d'ailleurs que le protocole HTTP.

Du côté du serveur, Broadway est composé de six modules exécutés comme des processus distincts : le répartiteur d'événements, le proxy, le serveur de descriptions de pages, le serveur de profils, le serveur d'annotations, et enfin le serveur de recommandations. Pour les communications entre ces processus, nous utilisons pour les mêmes raisons que précédemment le protocole de communication RMI.

Le répartiteur d'événements a une position centrale dans cette architecture répartie et permet d'abonner un module à certains événements correspondant à des actions des utilisateurs ou du système. Les producteurs d'événements liés aux actions des utilisateurs sont les barres d'outils (les boutons d'évaluation par exemple émettent chacun un événement), les contrôleurs (début et fin d'une navigation par exemple), les observateurs dont l'unique rôle est d'envoyer des événements au début et à la fin de l'affichage d'une page et enfin le proxy qui indique les chargements de page et les erreurs éventuelles. Les consommateurs de ces événements sont les modules d'analyse du comportement de l'utilisateur comme par exemple le serveur de recommandations.

La fonction proxy de Broadway s'appuie sur le serveur HTTP Jigsaw (Baird-Smith *et al.*, 1997) conçu et réalisé par le W3C (World-Wide-Web Consortium). Au-dessus de ce serveur, préalablement configuré en tant que proxy, nous avons personnalisé les deux fonctions essentielles dans le cadre de Broadway : l'identification des utilisateurs lors de leurs requêtes, et le traitement des pages HTML avant leur transfert aux navigateurs des utilisateurs pour l'insertion des barres d'outils et des observateurs. Le proxy informe également le système des chargements de pages et transmet leur contenu au serveur de descriptions de pages qui est chargé d'extraire et de sauvegarder les caractéristiques pertinentes de chaque page visitée par le groupe d'utilisateurs : titre, mots clefs, date d'expiration, dernière date de modification. Le serveur de profils permet de gérer les identificateurs et les propriétés des utilisateurs (nom, prénom, adresse e-mail)[47]. Le serveur d'annotations permet d'associer à une page un ensemble d'annotations textuelles.

Enfin, le serveur de recommandations est la partie la plus complexe du système. Il permet en effet la sauvegarde des informations pertinentes liées aux navigations des utilisateurs et intègre le système de raisonnement à partir de cas capable de calculer les recommandations. Le serveur est abonné aux événements de navigation des utilisateurs, ainsi qu'aux événements de contrôle permettant de spécifier le mode de calcul (en continu ou à la demande). Quel que soit le mode choisi, les demandes de recommandations sont placées dans une file d'attente dans le serveur de recommandations. Un processus léger séparé traite ces demandes en séquence pour effectuer les raisonnements. Si des recommandations sont trouvées, le serveur transmet directement au contrôleur de l'utilisateur concerné les résultats pour les afficher[48]. Les recommandations sont donc calculées de manière asynchrone et sont affichées dès qu'elles sont disponibles.

1.2.2.3 *Broadway*Tools : des composants ouverts et réutilisables pour Broadway*

L'architecture du prototype actuel de Broadway et son fonctionnement reposent sur la spécialisation et l'intégration d'un ensemble de composants ouverts et réutilisables facilitant d'une part l'extension et la modification du système existant Broadway, et d'autre part, l'application de notre approche dans d'autres systèmes. Ces différents composants sont appelés des outils au regard des tâches de réalisation et de maintenance d'un système comme Broadway. L'ensemble de ces outils est dénommé Broadway*Tools et se compose de quatre groupes : les outils d'interface graphique, les

[47] Ce serveur est en cours d'extension pour associer à chaque utilisateur les groupes auxquels il appartient, les sessions auxquelles il participe, ses préférences, et son profil en termes de mots clefs pertinents pour son activité (Elhaddad, 1998).

[48] Techniquement, le serveur ouvre une connexion RMI avec l'applet située dans le navigateur de l'utilisateur. Ce mécanisme demande que des droits particuliers soient établis par l'utilisateur. Si ce n'est pas le cas, un mécanisme de scrutation (*polling*) du contrôleur est automatiquement lancé pour récupérer régulièrement les nouvelles recommandations.

outils pour la réalisation des serveurs, les outils pour la réalisation des services et enfin les outils de persistance.

Pour les outils d'interface graphique, un ensemble de composants est proposé pour construire des barres d'outils destinées à être insérées dans des pages HTML. Nous proposons des classes d'objets facilitant le décodage des paramètres de l'applet ainsi que des boutons prédéfinis.

Pour les serveurs, quatre ensembles de composants permettent de construire respectivement un serveur de recommandations, un serveur d'annotations, un serveur de profils utilisateurs et un serveur de pages. Pour chaque type de serveurs, nous proposons des classes d'objets simplifiant la gestion des requêtes avec le protocole RMI et des classes permettant de personnaliser le traitement effectué dans le serveur (sous la forme d'une librairie de classes ou d'une plate-forme à objets). Par exemple, le serveur de recommandations s'appuie sur notre plate-forme à objets CBR*Tools et le serveur de descriptions de pages utilise une bibliothèque de fonctions pour analyser les pages HTML et effectuer l'extraction des mots clefs.

Les services sont les modules fonctionnant entre les interfaces graphiques et les serveurs : proxy, observateur, répartiteur d'événements. Alors que l'observateur et le répartiteur d'événements sont des outils directement intégrables, l'outil proxy offre des fonctions de personnalisation pour la gestion de l'identification et du prétraitement des pages Web. Le répartiteur d'événements répond à des besoins similaires au service d'événements de CORBA (Geib *et al.*, 1997, pages 152–155), dans la mesure où il permet d'effectuer des communications asynchrones sans connexion directe entre un producteur d'événements et un nombre quelconque de consommateurs. La séparation entre les producteurs d'événements et leurs consommateurs est, dans notre approche, très importante pour répondre aux exigences d'ouverture. En effet, les producteurs peuvent être modifiés durant la maintenance du système sans affecter le fonctionnement du reste de Broadway. Par exemple, les événements de début et de fin de navigation, qui sont générés dans le présent prototype par le contrôleur, pourraient être générés automatiquement par un module d'analyse. De même, de nouveaux consommateurs peuvent être ajoutés au système sans modifier ou notifier les producteurs. Enfin, les consommateurs existants peuvent être facilement maintenus en modifiant les types d'événements auxquels ils sont abonnés. Techniquement, nous utilisons les possibilités offertes par le langage Java pour transmettre des classes d'objets et des objets entre différents processus. Un consommateur doit alors implanter un objet de filtrage d'événements (*listener*) qui est transmis au répartiteur d'événements (cf. Figure V-6). Lorsqu'un nouvel événement doit être traité par le répartiteur, les objets de filtrage correspondants sont activés et transmettent à leur tour uniquement les informations utiles à leur consommateur. Ainsi, l'envoi de messages inutiles entre le répartiteur et les consommateurs est supprimé, et l'abonnement d'un consommateur peut être dynamiquement changé au cours de l'exécution du système.

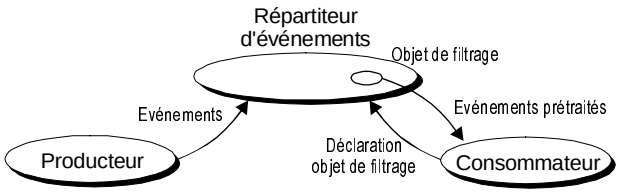

Figure V-6 : abonnement d'un module à des événements

Enfin, les outils de persistance reposent sur le produit ObjectStrore PSE Pro commercialisé par la société Object Design. Ce produit permet de réaliser la persistance d'objets Java dans une base de données, soit localement dans un fichier, soit dans un serveur distant ObjectStore. Un ensemble de classes et des outils sont fournis avec ce produit dont notamment un préprocesseur de code Java automatisant les annotations du code nécessaires à la persistance. Toutefois, des outils supplémentaires ont été réalisés pour faciliter l'utilisation de ObjectStorePSE dans le cadre de

Broadway (Golinelli, 1998)[49] : activation et désactivation de la persistance à la compilation, méthodologie pour rendre un serveur persistant.

1.3 Serveur de recommandations

Bien que Broadway soit accessible simultanément par de multiples utilisateurs, les demandes de recommandations sont ordonnées chronologiquement pour un traitement séquentiel. Ce traitement permet d'éviter les accès et les mises à jour concurrentes de la base de cas[50]. Le serveur de recommandations réutilise des cas qui sont principalement composés de la description du comportement d'un utilisateur et d'une liste de pages recommandées pour ce comportement. Alors que les navigations représentent les données brutes d'observation, les cas permettent de manipuler des associations pertinentes issues des navigations entre un comportement et des pages évaluées :

Cas = (comportement utilisateur issu d'une navigation, liste de pages évaluées et recommandées)

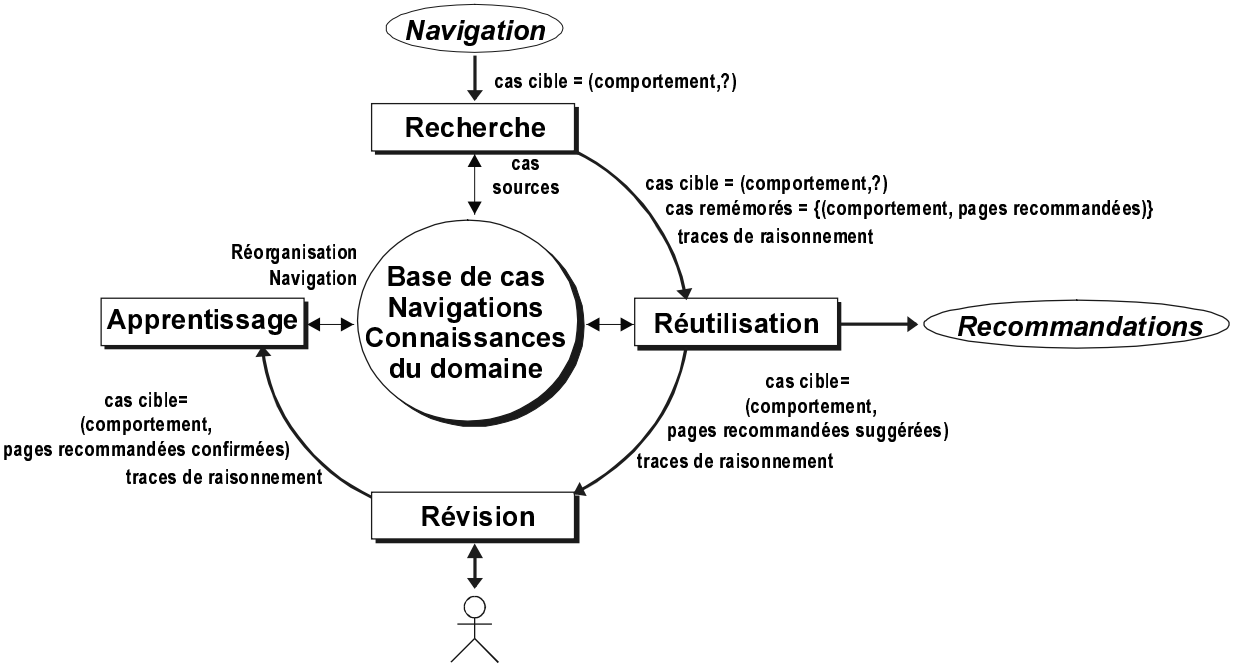

Figure V-7 : cycle du raisonnement dans Broadway

Le calcul des recommandations est alors effectué durant un raisonnement en quatre phases, étant donné la navigation courante d'un utilisateur (cf. Figure V-7) :

1. *Recherche*. La base de cas est parcourue pour identifier les cas dont la partie comportement peut être mis en correspondance avec la navigation courante. Les meilleurs cas sont alors retournés.
2. *Réutilisation*. Chaque cas retrouvé propose une liste de pages recommandées, et la phase de réutilisation opère la fusion de ces pages pour déterminer la liste des recommandations à présenter à l'utilisateur courant (pages recommandées suggérées).
3. *Révision*. La phase de révision est effectuée par l'utilisateur lorsqu'il continue sa navigation et qu'il évalue les pages visitées s'il le désire. La révision se poursuit jusqu'à

[49] La réalisation de ces outils est le résultat du projet de fin d'études (DESS) de M. Gollinelli que nous avons encadré (Golinelli, 1998).
[50] L'étude de la réalisation de raisonnements en parallèle dépasse la cadre de nos travaux.

la fermeture de la navigation courante et met en attente tous les raisonnements effectués pour une navigation (pages recommandées confirmées).

4. *Apprentissage et maintenance de la mémoire.* L'apprentissage prend en compte la navigation courante et l'ensemble des raisonnements associés pour en faire leur synthèse. La mémoire est alors mise à jour par la création de nouveaux cas, par la mise à jour des cas et par l'ajout de la navigation courante. A côté de ce processus de raisonnement, et étant donné que le Web est un espace en continuel changement, les cas et les navigations devenus obsolètes sont régulièrement effacés de la mémoire.

Dans la section suivante, nous détaillons l'application de notre modèle d'indexation par situations comportementales, et en particulier le processus de raisonnement.

2 Evaluation de notre modèle d'indexation et de CBR*Tools

Nous présentons tout d'abord l'application du modèle d'indexation par situations comportementales puis l'utilisation de notre plate-forme CBR*Tools pour la réalisation concrète du prototype Broadway. Enfin, nous faisons le bilan des apports et des limites du modèle d'indexation et de la plate-forme dans le cadre de cette application.

2.1 Application du modèle d'indexation par situations comportementales

L'utilisation de notre modèle d'indexation reposent sur quatre étapes d'ingénierie des connaissances (cf. chapitre II, §2.2) : identification des variables d'observation, détermination de la sémantique d'un enregistrement et de son contexte, définition de la représentation des cas, et enfin conception des phases du raisonnement. Nous suivons alors ces étapes pour montrer, dans le cadre de Broadway, l'utilisation des concepts introduits. La modélisation proposée de Broadway repose sur une analyse la plus réaliste possible du problème de l'assistance à la navigation. Toutefois, de nombreuses décisions liées au domaine doivent être approfondies notamment par des études ergonomiques et psychologiques. Ce travail dépasse le cadre de notre thèse, toutefois nous identifions certains paramètres importants de notre modélisation pour permettre leur modification (seuils, fonctions d'agrégation, ou autres méthodes). La liste de ces paramètres est donnée dans l'annexe D. Ainsi, Broadway représente un type de modélisation dont l'objectif est de donner un cadre opérationnel pour l'évaluation de notre modèle d'indexation.

2.1.1 Identification des variables d'observation

Pour l'observation du comportement des utilisateurs, nous avons retenu quatre variables représentant l'adresse, le contenu, l'évaluation et le ratio d'affichage des pages Web visitées. Ces variables qui sont notées respectivement Adresse, Contenu, Evaluation et Affichage, prennent des valeurs successives au cours du processus de recherche d'information pour chaque utilisateur. Nous décrivons l'utilisation de chacune de ces variables. Nous donnons également les caractéristiques de leur domaine de valeurs ainsi que les modules de Broadway utilisés pour les calculer.

2.1.1.1 *Adresse d'une page Web*

Comme nous l'avons expliqué auparavant (cf. chapitre I, §1.2.1), le Web est un ensemble interconnecté de ressources dans différents formats et accessibles par de multiples protocoles. Dans le cadre de Broadway, nous nous limitons à un espace bien précis au sein du Web. Cet espace est constitué des ressources accessibles via le protocole HTTP et qui respectent le format HTML. Cet

espace représente la partie la plus importante du Web actuel. Les ressources de cet espace sont accessibles via une adresse appelée URI (*Universal Resource Identifier*) dont la forme générale est la suivante (RFC 1738, 1994 ; RFC 2068, 1997,§3.2.2 ; RFC 1808, 1995) :

HTTP_URI = "http:" " //" machine [":" port] ["/" chemin] ["?" requête] ["#" fragment]

Toutefois, les contraintes d'écriture de ces adresses et de leur interprétation par les serveurs ne sont pas suffisamment strictes pour garantir l'unicité de l'association entre une adresse et une ressource. En effet, un même fichier peut être accessible par plusieurs URI non identiques. C'est le cas, par exemple, des URI suivantes :

http://www.inria.fr/aid/
http://www.inria.fr/aid
http://www.inria.fr/aid/index.html
http://www.inria.fr/aid/pub/../
http://www.inria.fr/aid/.
http://www.inria.fr/aid/./

Cependant aucune règle du protocole HTTP ou des URI ne permet de garantir que, dans tous les serveurs HTTP, ces types d'URI mèneront effectivement à un contenu identique. C'est pourquoi, nous clarifions les contraintes de comparaison utilisées dans Broadway pour déterminer si deux adresses sont identiques. Tout d'abord, la partie fragment, servant à référencer un point précis dans une page, est ignorée. De plus, nous reprenons et précisons les contraintes d'égalité reconnues entre deux URI pour le protocole HTTP (RFC 2068, 1997, §3.2.3 ; Bray, 1996). Ainsi deux adresses sont égales si et seulement si les parties machine, port, chemin et requête sont identiques avec les contraintes suivantes :

- Deux machines sont égales si leurs adresses IP sont identiques. Si les machines sont spécifiées symboliquement par leur nom sous forme de chaînes de caractères qui sont égales sans prise en compte de la casse, les machines sont considérées identiques, sans comparaison de leurs adresses IP.
- Deux ports sont identiques s'ils indiquent le même numéro. Un port vide est équivalent au port par défaut, soit 80.
- Deux chemins sont identiques si les chaînes de caractères correspondantes sont égales avec prise en compte de la casse. Un chemin vide est équivalent au chemin « / ».
- Deux requêtes sont identiques si les chaînes de caractères correspondantes sont égales avec prise en compte de la casse.

Le Tableau V-1 donne l'exemple de deux pages ayant différentes URI qui seront considérées comme identiques par Broadway.

URI d'une même page A	*URI d'une même page B*
http://www.inria.fr	http://www.inria.fr/index.html
http://www.inria.fr:80	http://WWW.inria.fr/index.html
http://www.inria.fr/	http://www.inria.fr/index.html#theme

Tableau V-1 : URI de pages Web

Enfin, Broadway considère comme des adresses de page Web uniquement les URI qui permettent d'accéder à des ressources dont le format est HTML. Cette information est extraite des communications HTTP avec les serveurs, dans lesquelles le champ Content-Type doit indiquer le type MIME (RFC 2046, 1996) text/html. Les adresses des pages Web visitées par un utilisateur sont transmises à Broadway par les observateurs. Les observateurs sont des programmes (applets) insérés par Broadway dans les pages Web et sont activés lors de l'affichage d'une page.

2.1.1.2 *Contenu d'une page Web*

Le contenu d'une page Web est décrit par du texte brut augmenté de directives de formatage, l'ensemble respectant le langage HTML (Raggett, 1997). Ces directives sont interprétées par le navigateur de manière à afficher le texte avec différents effets : hyperliens, insertion d'images, changement de la taille de la fonte, titre, sous-titres, liste numérotée… Plus précisément, le texte est enrichi de descripteurs (*tags*) qui apportent principalement des informations de mise en page et de structuration (cf. Figure V-8). Par exemple, les deux descripteurs <B> et </B> permettent de définir une zone de texte à afficher en gras (*bold*). Sur le plan de la structuration, les descripteurs <H1> </H1>, <H2> </H2>, <H3> </H3> permettent de définir des titres de niveaux respectifs 1, 2 et 3. De plus, une page est divisée en deux parties principales : une partie descriptive identifiée par les descripteurs <HEAD> et </HEAD>, et une partie informative entre les descripteurs <BODY> et </BODY>. La partie descriptive donne le titre de la page entre les descripteurs <TITLE> et </TITLE> ainsi que des métadonnées (*metadata*) sur les pages dans des descripteurs <META...> : mots clefs, description de la page, nom de l'auteur, logiciel utilisé pour écrire cette page. La partie informative contient le texte formaté à afficher. Ces deux parties sont réunies dans une zone encadrée par les descripteurs <HTML> et </HTML>, et la page commence par un descripteur <!DOCTYPE... > donnant la version du format utilisé.

```
<!DOCTYPE HTML PUBLIC "-//W3C//DTD HTML 3.2//EN">
<HTML>
<HEAD>
<TITLE>Bienvenue sur le site Web de l'INRIA</TITLE>
.....
<META  NAME="keywords"
       CONTENT=   "INRIA, recherche, informatique,
                  automatique, institut">
<META  NAME="description"
       CONTENT="INRIA, page d'accueil">
</HEAD>
<BODY>
.....
<H1>Actualitées</H1>
....
<H1>Présentation</H1>
....
</BODY>
</HTML>
```

Figure V-8 : exemple de structuration d'une page Web

L'objectif de la variable Contenu est de résumer les informations de la page pour pouvoir effectuer ultérieurement des comparaisons. Nous ne pouvons pas enregistrer le contenu intégral de chaque page car cette solution demanderait trop de ressources sachant que les pages considérées sont potentiellement issues du Web tout entier. L'algorithme utilisé dans Broadway permet alors de ne retenir d'une page qu'un ensemble de *m* mots clefs au maximum. Comme nous l'avons indiqué, une page Web donne déjà un ensemble de mots clefs grâce au descripteur <META...>. Cependant, ce descripteur n'est pas obligatoire et, dans la pratique, peu de pages Web l'utilisent. Des travaux en cours étudient la normalisation et l'extension de la gestion des métadonnées en vue de meilleures possibilités de traitements automatiques des pages Web (cf. *Resource Description Framework*, Lassila & Swick, 1998). Nous utilisons donc un algorithme d'extraction de mots clefs, spécifique et adapté aux pages HTML :

1. élimination des mots toujours considérés comme insignifiants (*stopwords*),
2. réduction des mots à leur racine (*stem*) pour, par exemple, ne pas différencier les accords d'un même mot (pluriel/singulier, féminin/masculin, conjugaisons),

3. affectation d'un coefficient de pertinence δ à chaque racine,
4. sélection des *m* mots ayant les plus grands coefficients de pertinence δ supérieurs au seuil δ_{min}.

Le principe de ce traitement est classique dans le domaine de la recherche d'information (Korfhage, 1997, chapitre 5 ; Porter, 1980). Le coefficient de pertinence d'une racine ω dans un document est généralement calculé de la manière suivante, étant donné la fréquence d'occurrence de la racine dans le document normalisée par sa taille, notée *tf*, et la fréquence des documents qui contiennent cette racine, notée *df*, dans une collection de *N* documents :

$$\delta(\omega) = tf\,.\,idf = tf\,.\,(\log_2 \frac{N}{df})$$

Ce coefficient permet de mieux noter une racine qui apparaît souvent dans une page, tout en regardant si elle est discriminante à la vue des autres documents. Cependant, ce type d'approche ne prend pas du tout en compte la spécificité des pages HTML qui présentent une certaine structuration. C'est pourquoi une extension a été récemment proposée (Cutler *et al.*, 1997) et permet de calculer le coefficient δ, en définissant six classes d'occurrence d'une racine : le titre de la page, les titres de niveaux 1 et 2, les titres de niveaux 3, 4 et 5, les mots en gras, le texte normal et le texte des hyperliens qui mènent à la page considérée. Un vecteur de fréquence de la racine dans ces classes ainsi qu'un vecteur de poids d'importance entre ces classes sont alors utilisés dans la formule suivante :

$$\delta(\omega) = (tf_1 \quad \cdots \quad tf_6)\begin{pmatrix} w_1 \\ \vdots \\ w_6 \end{pmatrix} idf$$

L'affectation des poids entre les classes est effectuée expérimentalement à partir de requêtes dont les experts ont manuellement identifié les documents pertinents, et dont le système tente d'optimiser les résultats obtenus automatiquement. Toutefois, cette technique n'est pas directement applicable dans Broadway puisqu'il n'existe pas de collection figée de documents. Les utilisateurs sont effectivement libres de naviguer sur un nombre quelconque de pages Web qui ne sont pas connues *a priori*. La valeur du coefficient *df* n'est donc pas fixe. De plus, cette approche ne prend pas en compte la classe des mots clefs déjà définis par la page via les métadonnées.

Nous proposons alors une méthode similaire et adaptée à nos contraintes. Nous définissons uniquement trois classes de mots pour une page : les mots clefs existants (métadonnées), les mots mis en valeur (titre de la page, description et sous-titres de niveaux 1, 2 et 3), et les mots apparaissant dans les autres zones. La prise en compte d'uniquement trois classes permet, plus facilement et de manière heuristique, de donner des poids d'importance aux classes : les mots clefs existants sont très pertinents, puis les mots mis en valeur sont plus pertinents que les autres. Enfin, nous supposons que le facteur évaluant la qualité discriminante d'un mot ω est donné par la fonction Δ et ne change pas dans le temps. Ces facteurs peuvent être précalculés à partir d'une collection de documents typiques du domaine d'activité du groupe utilisant Broadway. Ainsi, pour une page donnée Π, Broadway extrait les *m* mots ayant les plus grands coefficients d'importance δ calculés sur ces trois classes :

$$\delta(\omega) = (tf_{MotClefs}, tf_{MisEnValeur}, tf_{Autre})\begin{pmatrix} w_{MotsClefs} \\ w_{MisEnValeur} \\ w_{Autre} \end{pmatrix} \Delta(\omega)$$

Dans l'architecture de Broadway, le serveur de descriptions de pages est chargé d'effectuer ces traitements et de sauvegarder les mots clefs identifiés. De plus, grâce aux informations données par les serveurs HTTP (date d'expiration et dernière date de modification), ce serveur ne met à jour ces mots clefs que lorsqu'une modification a eu lieu.

2.1.1.3 *Evaluation explicite des pages*

La variable mesurant l'évaluation explicite des pages Web permet d'obtenir une information directe sur la satisfaction de l'utilisateur au regard de son objectif implicite. Cet objectif peut être plus ou moins précis et peut évoluer durant une navigation. L'observation des évaluations va permettre d'interpréter cet objectif et de noter les recommandations faites par le système. Grâce à la barre d'outils de Broadway, les évaluations sont directement associées à la page visualisée avec quatre niveaux sémantiques qui sont représentés par des entiers :

- *pas pertinent* (valeur –1) : la page courante n'a aucun rapport avec l'objectif de recherche et représente un contresens.
- *indifférent* (valeur 0) : il s'agit de la valeur par défaut et dénote un avis indifférent de l'utilisateur.
- *pertinent* (valeur 1) : la page courante est connectée au thème de la recherche, ou présente des hyperliens qui traitent de thèmes proches de l'objectif de recherche.
- *très pertinent* (valeur 2) : la page courante répond précisément à l'objectif de recherche.

2.1.1.4 *Evaluation implicite des pages : le ratio d'affichage*

La variable précédente permet d'observer les évaluations explicites des pages Web données par les utilisateurs. Dans Broadway, nous choisissons également de considérer le temps de lecture comme critère d'intérêt *implicite* d'un utilisateur dans une page. La combinaison de critères explicites et implicites permet une meilleure représentation du comportement des utilisateurs et mène généralement à de meilleures interprétations (Sakagami & Kamba, 1997). Nous prenons l'hypothèse que, plus un utilisateur passe de temps à lire une page, plus il est intéressé par son contenu. Cependant, le temps de lecture est difficilement observable et nous le supposons égal au temps d'affichage. De plus, pour pouvoir comparer des temps d'affichage de pages différentes, nous normalisons cette durée par la taille de la page et nous obtenons le *ratio d'affichage.* Ce ratio représente en fait l'inverse de la vitesse de lecture.

Concrètement, les valeurs de cette variable sont mesurées en effectuant le rapport entre la durée de l'affichage en millisecondes, et le nombre de mots lisibles dans une page. Le temps d'affichage est obtenu par l'observateur qui informe très précisément du temps écoulé entre le début de l'affichage de la page et son effacement pour afficher une nouvelle page. Le nombre de mots lisibles d'une page HTML est calculé par le serveur de descriptions de pages qui doit préalablement analyser la structure HTML pour en éliminer tous les descripteurs ainsi que leur contenu. Enfin, nous considérons qu'un ratio d'affichage supérieur au seuil a_{max} montre que l'utilisateur a certainement interrompu sa lecture. Nous n'utilisons cette variable que pour évaluer positivement l'intérêt d'un utilisateur lorsque le ratio est supérieur à un seuil donné. En effet, pour un ratio petit, l'utilisateur peut avoir trouvé rapidement l'information recherchée.

2.1.2 Détermination de la sémantique d'un enregistrement et de son contexte

Dans le cadre de Broadway, un enregistrement est appelé une *navigation*. Une navigation contient l'évolution des quatre variables d'observation (adresse, contenu, évaluation, ratio d'affichage) pour la durée d'une recherche d'information. Une navigation débute lorsqu'un utilisateur a le désir d'effectuer une recherche et se termine lorsqu'il a trouvé l'information recherchée ou bien s'il abandonne la

recherche. Une navigation est supposée *cohérente*, c'est-à-dire que l'objectif de recherche peut évoluer ou se préciser, mais il ne doit pas y avoir de changement d'objectif. Par exemple, une navigation commencée pour rechercher des informations sur le langage de programmation Java et les modules de communication disponibles, peut se préciser par la lecture de pages qui présentent des applications spécifiques réalisées en Java avec RMI. Toutefois, cette navigation sera considérée comme incohérente si l'utilisateur parcourt finalement le site de BMW parce qu'il cherche une nouvelle voiture et que l'un des concepteurs de ces applications mentionne ce site sur sa page personnelle.

L'évolution de chaque variable est représentée par une chronique échantillonnée dans la mesure où l'unité de temps pris en compte est le changement d'une page. Un changement de page correspond à l'affichage dans le navigateur d'une page dont l'adresse est différente de la dernière page affichée chronologiquement. Le contexte d'une navigation contient une description instantanée des caractéristiques pertinentes de la navigation. Deux attributs du contexte permettent de sauvegarder respectivement les noms des machines traversées durant la navigation, et les mots clefs de la navigation. Cet ensemble de mots clefs correspond à l'union des mots clefs issus des pages Web de la navigation.

La Figure V-9 donne l'exemple d'une navigation enregistrée par Broadway dans laquelle un utilisateur désire retrouver des informations concernant Broadway lui-même. Il commence alors sa navigation sur la page d'accueil de l'INRIA, passe par les pages de l'équipe de recherche AID et finalement trouve la page relatant les développements de logiciels et notamment ceux de Broadway.

Navigation #16 from mjaczyn

start date: Wed Nov 26 19:28:20 GMT+03:30 1997

end date: Wed Nov 26 19:30:51 GMT+03:30 1997

keywords (stems): PEOPLE, REPORT, INRIA, PUBLIC, CONFER, ACTIV, STAGE, MICHEL, THEME, SITE, SOFTWAR, DEADLIN, JACZYNSKI, ACTION, RESEARCH, PREFER

hosts: www.inria.fr

#	Page address	Content (stems)	Display	Evaluation
0	http://www.inria.fr/Recherche/activites-eng.html	INRIA, ACTIV, RESEARCH	1.0	0
1	http://www.inria.fr/Themes/Theme3-eng.html	INRIA, THEME	1.0	0
2	http://www.inria.fr/Equipes/AID-eng.html	ACTION	3.0	0
3	http://www.inria.fr/aid/aid-eng.html	ACTION	6.0	☺
4	http://www.inria.fr/aid/people.html	PEOPLE, ACTION	8.0	0
5	http://www.inria.fr/aid/personnel/Michel.Jaczynski/michel-eng.html	MICHEL, SITE, JACZYNSKI, PREFER	38.0	0
6	http://www.inria.fr/aid/personnel/Michel.Jaczynski/pub-eng.html	REPORT, PUBLIC, MICHEL, JACZYNSKI	10.0	0
7	http://www.inria.fr/aid/personnel/Michel.Jaczynski/michel-eng.html	MICHEL, SITE, JACZYNSKI, PREFER	1.0	0
8	http://www.inria.fr/aid/people.html	PEOPLE, ACTION	4.0	0
9	http://www.inria.fr/aid/aid-eng.html	ACTION	8.0	0
10	http://www.inria.fr/aid/confs.html	CONFER, DEADLIN, ACTION	0.0	0
11	http://www.inria.fr/aid/aid-eng.html	ACTION	2.0	0
12	http://www.inria.fr/aid/training.html	STAGE, ACTION	1.0	0
13	http://www.inria.fr/aid/software.html	SOFTWAR, ACTION	2.0	☺

Figure V-9 : extrait d'une navigation

2.1.3 Définition de la représentation des cas

Nous décrivons la représentation du cas cible puis celle des deux types de cas sources (potentiels et concrets).

2.1.3.1 *Cas cible*

Un cas cible référence un instant précis dans une navigation et se compose d'un couple (*problème*, *solution*) et des informations de gestion : date de création, nom de l'utilisateur, origine du cas, nombre de remémorations (nombre de fois où le cas a été retrouvé dans les raisonnements), et nombre de confirmations (nombre de fois où au moins une de ses recommandations a été confirmée après révision). Le problème donne la situation comportementale dont uniquement les indices instantanés sont connus (contexte de la navigation) : nom des machines accédées et mots clefs des pages visitées. La solution qui est recherchée doit donner une liste de couples (adresse d'une page recommandée, coefficient entre 0 et 1). Le coefficient d'une page indique l'estimation du *facteur de pertinence* de la recommandation étant donné l'état courant de la navigation. L'expression typique d'un cas cible est la suivante :

```
CasCible {
    Identificateur = "exemple-cas-cible-Broadway",
    UtilisateurNavigation = "mjaczyn", Date = "26/11/97",
    Origine = cas-cible,
    NbRemémorations = -, NbConfirmations = -,
    Problème = {
        Enregistrement= N, Instant= 14,
        SituationComportementale= {
            ComposanteInstantanée= {
                ContextEnregistrement= {
                    Machines= "www.inria.fr",
                    MotsClefs="ACTION, RESEARCH, DEADLIN,..."}},
            ComposanteComportementale= {
                ComportementsElémentaires = {},
                ContraintesTemporelles= {}}}}
    Solution= { ?}}
```

2.1.3.2 *Cas potentiels et cas concrets*

Dans Broadway, les cas concrets sont uniquement issus de l'instanciation d'un seul patron de cas potentiels. Le patron proposé repose sur des règles de construction d'une situation comportementale et d'une liste de pages recommandées pour un instant donné à l'intérieur d'une navigation. Ce patron est applicable uniquement si la position de l'instanciation dans la navigation est supérieur à $l_{\min}$ et si le couple formé de la navigation associée à l'instant d'instanciation ne correspond pas à une instanciation déjà effectuée et mémorisée sous forme de cas concret (liste notée L_C).

Applicable_P : Navigation, Intant(u) $\rightarrow$ Booléen

$\{ \text{Applicable}_P(n,i) = \text{Len}(n) > l_{\min} \text{ et } \langle n,i \rangle \notin L_C \}$

La fonction d'instanciation du patron crée un cas potentiel explicite, en suivant des règles de construction de la situation comportementale et de la solution (l'algorithme d'instanciation est décrit précisément dans l'annexe D, §2.1). La situation comportementale comprend une composante instantanée (noms des machines et mots clefs) et une composante comportementale. La composante comportementale référence un instant précis dans la navigation et sélectionne les indices comportementaux d'un cas par rapport à l'instant de référence :

- les p dernières pages sur les variables de l'adresse et du contenu pour identifier le contexte immédiat de réutilisation d'un cas dans une navigation (notion de restriction, cf. chapitre II, §2.2.1.1),

- un ensemble de pages passées qui sont sélectionnées[51]pour caractériser le comportement de l'utilisateur : une évaluation explicite élément d'une liste e_p, ou ratio d'affichage dépassant le seuil a_p,
- un ensemble de contraintes temporelles permettant d'ordonner les pages passées sélectionnées.

Ainsi, les conditions de réutilisation d'un cas sont manipulées plus facilement à travers la restriction (structure figée) et peuvent être précisées avec la liste des pages passées sélectionnées et ordonnées.

La partie solution du cas qui donne une liste de pages recommandées est construite en sélectionnant les pages pertinentes futures par rapport à l'instant de référence (seuil φ_f). Le facteur de pertinence de chaque page est calculé par agrégation de l'évaluation explicite et de l'évaluation implicite (fonction φ). Pour la sélection des pages passées et futures, si une même page est visitée plusieurs fois dans une navigation, nous ne retenons que la plus récente dépassant les seuils imposés.

La Figure V-10 donne l'exemple d'un cas concret créé à partir de l'instanciation du patron de cas potentiels dans la navigation de la figure précédente (cf. Figure V-9) pour un instant correspondant à la page #9. La restriction permet de sélectionner les trois dernières pages (p=3) et les règles du patron sélectionnent la page #3 pour son évaluation (e_p={2}) et la page #5 pour le ratio d'affichage (a_p=20). A cette situation comportementale, la page #13 est associée (φ_f=0,8) et représente la page recommandée de ce cas. Suivant la notation introduite (cf. chapitre II, §2.1.3.3), nous représentons ce cas de la manière suivante :

CasConcret {
Identificateur = "exemple-cas-Broadway",
UtilisateurNavigation = "mjaczyn", **Date** = "26/11/97",
Origine = cas-potentiel,
NbRemémorations = 0, **NbConfirmations** = 0,
Problème = {
Enregistrement= R, Instant= 9,
SituationComportementale= {
ComposanteInstantanée= {
ContextEnregistrement= {
Machines= "www.inria.fr",
MotsClefs="PEOPLE, REPORT, INRIA…"}},
ComposanteComportementale= {
ComportementsElémentaires = {
b_1={ ts= K_1, loc=K_1(7,9), Ic=Ψ_P(.,Adresse,-2,0))},
b_2={ ts =K_2 loc=K_2(7,9), Ic=Ψ_P(.,Contenu,-2,0)},
b_3={ ts = K_1, loc=K(5,5), Ic=Ψ_B(.,Adresse,K_1(5,5),K_2(5,5))},
b_4={ ts = K_2, loc=K_2(5,5), Ic=Ψ_R(.,Contenu,f(.,3)},
b_5={ ts = K_3, loc=K_3(5,5),Ic=Ψ_R(.,Evaluation,f(.,3))},
b_6={ ts = K_4, loc=K_4(5,5),Ic=Ψ_R(.,Affichage,f(.,3))}
b_7={ ts = K_1, loc=K_1(3,3),Ic=Ψ_B(.,Adresse,K_1(3,3),K_2(3,3))},
b_8={ ts = K_2, loc=K_2(3,3),Ic=Ψ_R(.,Contenu,f(.,7))},
b_9={ ts = K_3, loc=K_3(3,3),Ic=Ψ_R(.,Evaluation,f(.,7))},
b_{10}={ ts = K_4, loc=K_4(3,3),Ic=Ψ_R(.,Affichage,f(.,7))}},
ContraintesTemporelles= {
c1= avant(b_7,b_3) }}},
Solution= { p1= { Adresse=K_1(14,14),
Evaluation= φ(K_3(13,13), K_4(13,13))}}}

[51] Cette sélection peut recouvrir les pages qui constituent la restriction.

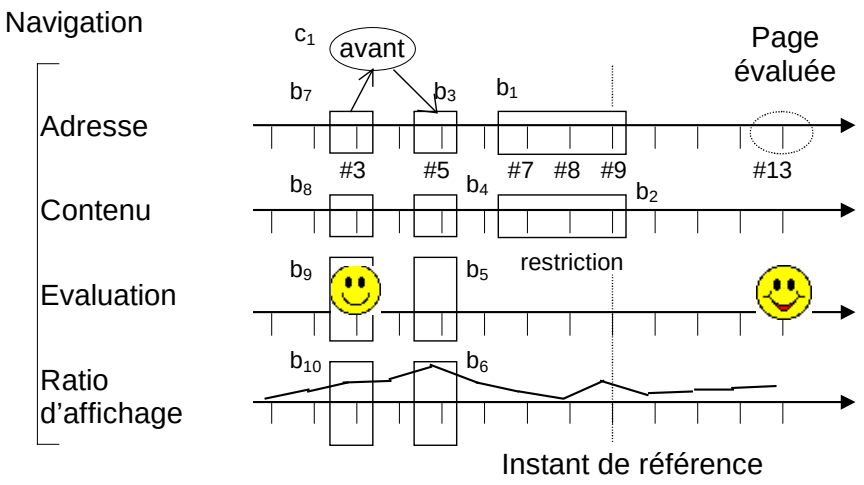

Figure V-10 : exemple d'un cas concret

Dans cette expression, la fonction f permet d'obtenir, suivant son numéro, le comportement élémentaire homologue déjà identifié dans une composante comportementale. De plus, nous introduisons une nouvelle fonction de localisation Ψ_B qui permet d'identifier la première page passée à partir de l'instant de référence, qui dépasse le seuil de similarité τ_{loc}, soit avec l'adresse, soit avec le contenu. Dans ce but, les fonctions de similarité $s_{Adresse}$ et $s_{Contenu}$, que nous décrivons dans la section suivante (cf. §2.1.4.1.1), sont utilisées. Notre objectif est de pouvoir identifier la première page qui présente une similarité très élevée (τ_{loc} proche de 1) avec soit le contenu, soit l'adresse d'une page. Durant la mise en correspondance, cette fonction est donc utilisée pour identifier dans un cas cible la page homologue d'une page sélectionnée dans un cas source.

Ψ_B : ComposanteComportementale, Variable, URL, EnsembleMots → TSchronique(u,T)

{ Ψ_B (c,v,adresse, contenu) = g(Γ(Chr(c,Adresse),1,Ref(c)), Γ(Chr(c,Contenu),1,Ref(c)),
Γ(Chr(c,Evaluation),1,Ref(c)), Γ(Chr(c,Affichage),1,Ref(c)),
adresse, contenu)

g(A.<i,a> , M.<i,m> , E.<i,e> , R.<i,r> , adresse, contenu) =
[<i,a>] si ($s_{Adresse}$ (a, adresse) > τ_{loc} ou $s_{Contenu}$(m, contenu) > τ_{loc}) et (e ∈ e_p ou a_{max} >= a > a_p)
g(A,M,E,R, adresse, contenu) sinon

g([],[],[],[],adresse, contenu)=[]

}

2.1.4 Conception des phases de raisonnement

Le raisonnement effectué par Broadway suit les quatre phases classiques en raisonnement à partir de cas : recherche, réutilisation, révision et apprentissage. Dans la spécification de ces phases, nous montrons comment nous avons utilisé les principes de gestion définis dans le modèle d'indexation par situations comportementales.

2.1.4.1 *Phase de recherche*

La phase de recherche repose sur l'utilisation de fonctions de similarité que nous introduisons en premier lieu, et d'une décomposition en étapes conformément au modèle d'indexation.

2.1.4.1.1 Fonctions de similarité

Pour effectuer les comparaisons entre cas, nous avons défini cinq fonctions élémentaires de similarité (une fonction pour chaque variable d'observation et une fonction pour les contraintes temporelles) ainsi qu'une fonction d'agrégation. Pour la variable Adresse, nous utilisons la structure hiérarchique sous-jacente dans l'organisation des données accessibles depuis un serveur HTTP. Les pages sont en effet généralement organisées en répertoires et nous effectuons une décomposition des adresses (cf. Figure V-11).

Chaque répertoire possède une sémantique et contient des pages qui sont reliées suivant le point de vue du concepteur. De plus, les répertoires les plus profonds contiennent des informations plus précises que les répertoires de surface. Nous utilisons alors la fonction élémentaire de similarité suivante qui prend en compte ces propriétés (REMIND, 1992) :

$$s_{\text{Adresse}}(x, y) = 1 - \frac{\text{h}(x, \text{MSCA}(x, y)) + \text{h}(y, \text{MSCA}(x, y))}{\text{h}(x, \text{root}) + \text{h}(y, \text{root})}$$

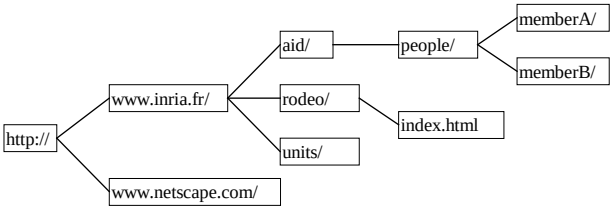

Figure V-11 : décomposition hiérarchique des adresses

Dans cette fonction, une adresse est considérée comme la référence d'un nœud d'une hiérarchie de symboles. MSCA donne le nœud commun le plus spécifique des deux adresses, et h donne le nombre de liens entre deux nœuds. Par exemple, nous obtenons les résultats suivants :

s_{Adresse}(http://www.inria.fr/aid/people/memberA, http://www.inria.fr/aid/people/memberB) = 0,75

s_{Adresse}(http://www.inria.fr/aid/people/memberA, http://www.inria.fr/rodeo/index.html) = 0,28

s_{Adresse}(http://www.inria.fr/rodeo, http://www.inria.fr/aid) = 0,5

Pour la variable Contenu, nous utilisons une fonction élémentaire de similarité prenant en compte le nombre de mots clefs communs entre deux pages :

$$s_{\text{Contenu}}(x, y) = \begin{cases} 0 \text{ si } x \cup y = \varnothing \\ \dfrac{\text{Card}(x \cap y)}{\text{Card}(x \cup y)} \text{ sinon} \end{cases}$$

Pour la variable Evaluation, nous utilisons les entiers associés à chaque niveau d'évaluation {-1,0,1,2} pour définir la fonction suivante :

$$s_{\text{Evaluation}}(x, y) = 1 - \frac{|x - y|}{3}$$

Pour la variable Affichage, nous considérons qu'un ratio d'affichage supérieur à a_{max} montre que l'utilisateur a certainement interrompu sa lecture pour effectuer une autre tâche. Dans ce cas, les ratios ne sont pas comparables, et nous donnons une similarité de 0,5. Sinon, nous définissons une fonction sur le domaine $[0 \ldots a_{max}]$:

$$s_{\text{Affichage}}(x,y)=\begin{cases}0{,}5 \text{ si } x>a_{\max} \text{ ou } y>a_{\max}\\ 1-\dfrac{|x-y|}{a_{\max}} \text{ sinon}\end{cases}$$

La dernière fonction élémentaire de similarité permet de comparer les contraintes temporelles. A partir des n contraintes exprimées dans une situation comportementale d'un cas source, $n(n+1)/2$ contraintes temporelles prises deux à deux sont dérivées (cet ensemble est noté y). Les comportements élémentaires homologues à ce cas source sont identifiés dans le cas cible. Sur l'ensemble y des contraintes initiales, seulement certaines contraintes restent vérifiées (cet ensemble est noté x). La similarité suivant les contraintes temporelles est alors donnée par la fonction suivante :

$$s_{\text{ct}}(x,y)=\frac{\text{Card}(x)}{\text{Card}(y)}$$

Par exemple, si la situation d'un cas source donne la séquence de pages sélectionnées (ABCDEF), nous obtenons les résultats suivants, avec q, la fonction retournant l'ensemble des contraintes vérifiées d'une séquence de pages :

s_{ct}(q(ABCDEF),q(ABCDEF))=1
s_{ct}(q (ABCDEF), q(FEDCBA))=0
s_{ct}(q (ABCDEF), q(ABEF))=6/15=0,4
s_{ct}(q(ABCDEF), q(ABEFCD))=11/15=0,73

Ces différentes fonctions élémentaires de similarité sont utilisées à plusieurs reprises dans les étapes de la recherche, et nous utilisons une fonction d'agrégation calculant une moyenne du vecteur v des différentes similarités élémentaires obtenues, pondérées par un vecteur de poids w :

$$g(v,w)=\frac{\sum_{i=1}^{n} v_i.w_i}{\sum_{i=1}^{n} w_i}$$

2.1.4.1.2 Etapes de recherche

La phase de recherche et, plus globalement, le raisonnement ne sont exécutés que lorsque la navigation courante a une longueur supérieure au minimum requis, noté $l_{\min}$. En effet, en dessous de cette longueur, nous estimons ne pas avoir assez d'information pour tenter de calculer des recommandations. L'objectif de la recherche est d'identifier au maximum les k meilleurs cas, suivant une recherche en deux phases. Premièrement, la phase de recherche des cas concrets est effectuée. Si cette recherche est un échec, c'est-à-dire qu'aucun cas n'est retrouvé, alors la deuxième phase qui recherche des cas potentiels est effectuée. Si la recherche ne permet pas d'identifier des cas, aucune recommandation ne sera fournie. Durant les étapes de recherche, nous mettons en évidence l'utilisation de seuils de similarité qui vont permettre d'assurer un minimum de similarité sur des parties mises en correspondance entre les cas sources et le cas cible. Il est en effet préférable d'arrêter la recherche plutôt que de continuer pour proposer des recommandations qui ne sont pas de qualité suffisante.

Pour la recherche des cas concrets, nous suivons les étapes de recherche suivantes :

1. *Filtrage sur la composante instantanée*. Cette première étape permet d'éliminer de la recherche les cas dont le contexte n'a pas au moins un nom de machine ou un mot clef en commun avec la navigation courante. En effet, ces cas ne peuvent être réutilisés pour le mécanisme de recommandation. Cette étape évite ainsi des comparaisons inutiles et augmente l'efficacité de la recherche.

2. *Filtrage sur la restriction.* L'objectif de cette étape est d'assurer un minimum de similarité des cas sources avec les toutes dernières pages que l'utilisateur a visitées dans la navigation courante. Chaque cas possède une restriction prédéfinie qui représente les p dernières pages sur les variables Adresse et Contenu. Ainsi, chaque cas définit une matrice de $2p$ valeurs qui est mise en correspondance avec les valeurs de la navigation courante. Cette comparaison utilise deux fonctions élémentaires de similarité, s_{Adresse} et s_{Contenu}, ainsi que la fonction d'agrégation g utilisant une matrice de poids *MP*. Cette étape ne retient finalement que les meilleurs cas (plus proches voisins avec un nombre maximum de cas k_r et un seuil de similarité τ_r).
3. *Filtrage sur les comportements élémentaires.* L'objectif de cette étape est d'assurer un minimum de similarité entre les pages pertinentes des cas sources et les pages visitées dans la navigation courante. Ainsi, pour chaque cas source issu de l'étape précédente, les contraintes de localisation sont appliquées pour identifier dans la navigation courante les pages homologues aux pages passées sélectionnées. Puis, la similarité de chaque page suivant les quatre variables (fonctions s_{Adresse}, s_{Contenu}, $s_{\text{Evaluation}}$, $s_{\text{Affichage}}$) est calculée puis agrégée avec la fonction g (vecteur de poids *VP*). Une page non identifiée dans le cas cible donne comme similarité 0. Ensuite, la similarité du cas cible avec un cas source est globalement donnée par le rapport entre la somme de ces similarités, et le nombre de pages sélectionnées dans le cas source. Cette similarité est alors combinée (fonction g avec le vecteur de poids *CP*) avec le ratio de complexité de la composante comportementale du cas source (ratio du nombre de pages sélectionnées sur le nombre maximum de pages sélectionnées dans les cas sources à comparer). Cette dernière opération a pour but de privilégier des cas constitués de nombreux comportements, ce qui dénote un cas plus spécifique pour une meilleure recommandation. Pour finir, uniquement les meilleurs cas sont retournés (plus proches voisins avec un nombre maximum de cas k_c et un seuil de similarité τ_c). Cet algorithme est décrit plus précisément dans l'annexe D, §2.2.
4. *Sélection sur les contraintes temporelles.* Finalement, les cas restants sont ordonnés suivant l'agrégation (avec la fonction de similarité g et le vecteur de poids *GP*) des similarités issues des étapes 2 et 3 et de la similarité des contraintes temporelles, calculée avec la fonction s_{ct}. Les meilleurs cas suivant ce critère sont alors retournés (plus proches voisins avec un nombre maximum de cas k et un seuil de similarité τ).

Pour la recherche de cas potentiels, nous suivons également ces quatre étapes. Toutefois, pour le filtrage sur la restriction, les navigations retenues par le filtrage sur la composante instantanée sont examinées pour chaque instant où le patron de cas potentiels est applicable. Une comparaison sur la restriction est effectuée comme pour les cas concrets. Durant la sélection des meilleurs cas par plus proches voisins, si deux cas issus d'une même navigation sont à comparer, seul le meilleur est retenu[52]. A la fin de cette étape, les cas potentiels identifiés sont effectivement explicités par instanciation du patron.

La phase de recherche retourne un ensemble de cas, et chaque cas donne une liste de pages recommandées. De plus, le nombre de remémorations de chaque cas identifié est incrémenté (attribut NbRemémorations).

2.1.4.2 *Phase de réutilisation*

L'objectif de la phase de réutilisation est de considérer l'ensemble de ces cas retrouvés pour former la liste des pages à recommander, tout en leur associant un facteur de pertinence. L'algorithme suivant est alors effectué :

[52] Cette contrainte évite ainsi de prendre par la suite en considération des cas qui diffèrent uniquement sur l'instant d'instanciation au sein d'une même navigation.

1. constitution de l'ensemble des pages à examiner en effectuant l'union des pages indiquées par les cas,
2. calcul du facteur de pertinence de chaque page,
3. sélection des meilleures pages, étant donné un degré minimum τ_r et un nombre maximum de recommandations *r*.

Le facteur de pertinence d'une page repose sur le calcul d'un vecteur de caractéristiques élémentaires de pertinence, noté *F*, et de leur agrégation suivant une moyenne pondérée avec le vecteur *FP*. Le calcul de chaque caractéristique donne un résultat entre 0 et 1, où 1 représente le cas le plus favorable. Différentes caractéristiques sont utilisées pour une page donnée :

1. l'évaluation globale de la page donnée par la moyenne des évaluations des cas sources proposant cette page,
2. le degré de confiance dans cette évaluation représenté par la moyenne des similarités des cas proposant cette page,
3. la représentativité de l'évaluation globale de la page donnée par le ratio du nombre de cas qui proposent cette page et du nombre de cas à comparer,
4. la moyenne de la qualité des cas proposant cette page (la qualité d'un cas est donnée par le ratio du nombre de confirmations sur le nombre de remémorations),
5. la présence d'annotations textuelles reliées à cette page (0 absence et 1 présence),
6. le temps de chargement moyen normalisé par le maximum des temps de chargement moyen des pages à comparer (utilisation des statistiques maintenues par le serveur de descriptions de pages).

Ainsi, Broadway met à profit dans la réutilisation quatre groupes de critères :

- les solutions proposées par les cas sources (critère 1),
- les informations de la mise en correspondance (critères 2 et 3),
- les statistiques sur la réutilisation des cas (critères 4),
- les informations sur les pages maintenues par les serveurs de descriptions de pages et d'annotations (critères 5 et 6).

Apres ce calcul, la liste des *pages recommandées suggérées* est affichée à l'utilisateur.

2.1.4.3 *Phase de révision*

La phase de révision est effectuée par l'utilisateur, lorsqu'à la suite de l'affichage des recommandations, il continue sa navigation, soit en sélectionnant une de ces pages, soit en choisissant un des hyperliens qui sont proposés dans les pages en cours d'affichage. Plus précisément, l'utilisateur est capable, grâce à la barre d'outils, d'évaluer les nouvelles pages visitées et en particulier celles qui lui ont été recommandées.

La particularité de Broadway est de permettre la révision simultanée de plusieurs raisonnements. En effet, pour une même navigation, de multiples raisonnements ont pu être débutés, soit pour des recommandations en continu, soit pour des recommandations à la demande. Tous ces raisonnements restent dans l'état de révision jusqu'à la fermeture de la navigation. A ce stade final, la solution révisée pour chaque cas cible est constituée des pages futures sélectionnées et notées de manière identique à la construction de la solution par le patron de cas potentiels. Cette solution donne la liste des *pages recommandées confirmées*[53].

[53] Le terme « recommandations confirmées » s'entend dans le sens d'une « solution confirmée » issue de la phase de révision du RàPC. Les recommandations confirmées peuvent contenir, soit des pages effectivement proposées suite à la réutilisation, soit de nouvelles pages évaluées par l'utilisateur.

2.1.4.4 *Phase d'apprentissage et maintenance de la base de cas*

Cette phase prend en entrée la navigation courante et l'ensemble des raisonnements en cours. L'apprentissage doit alors effectuer la synthèse de ces raisonnements pour mettre à jour le système et lui permettre d'améliorer son processus de recommandation. L'apprentissage se fait alors par trois mécanismes :

1. *Mise à jour des cas.* Les cas concrets retrouvés qui ont servi à la réutilisation sont examinés. Si un cas concret a proposé au moins une page qui se retrouve dans la liste des pages recommandées confirmées, son nombre de confirmations est incrémenté (attribut NbConfirmations). Ainsi, la qualité du cas augmente (nombre de confirmations sur le nombre de remémorations) et permettra de mieux classer les recommandations venant de ce cas, lors de la réutilisation pour des raisonnements futurs.
2. *Ajout des cas potentiels*. Les raisonnements pour la navigation courante sont examinés en séquence. Pour chaque raisonnement deux situations se présentent : soit uniquement la recherche des cas concrets a été effectuée, soit la recherche des cas potentiels a été effectuée à la suite de l'échec de la recherche des cas concrets.
   (a) Dans la première situation, la recherche des cas potentiels est effectuée. Si un cas potentiel ainsi retrouvé propose comme recommandation au moins une page qu'aucun cas concret ne proposait et qui fait partie des pages confirmées, il est ajouté comme cas concret. Cette opération a pour but de mettre à jour l'ensemble des cas concrets pour des recommandations que le système aurait pu donner si la recherche des cas potentiels avait été faite au début du raisonnement. Pour garder le gain de performance amené par la stratégie de recherche séquentielle, cette opération doit être faite hors ligne.
   (b) Dans la deuxième situation, si un cas potentiel propose au moins une recommandation qui a été confirmée, il est ajouté à la mémoire en tant que cas concret. Il s'agit dans cette situation de mémoriser les cas potentiels qui proposent des recommandations qui peuvent être utiles dans les raisonnements futurs, puisqu'elles sont confirmées par la navigation courante.

   Dans les deux cas, si un cas potentiel est mémorisé, sa navigation associée à son instant de référence est ajoutée à la liste L_C. Ainsi, la patron de cas potentiels ne pourra plus être instancié pour cet instant dans la navigation considérée.
3. *Ajout de la navigation courante* : si la navigation comporte au moins une évaluation implicite ou explicite (évaluation élément de e_n, ou ratio d'affichage supérieur à a_n) et possède une longueur suffisante l_{min}, elle est ajoutée dans la base des navigations. Grâce à la recherche de cas potentiels effectuée au besoin, cette technique permet d'effectuer l'apprentissage de comportements non encore identifiés (apprentissage paresseux). De plus, cet apprentissage permet d'éviter la mémorisation des cas cibles qui sont très nombreux dans Broadway puisqu'un raisonnement peut être lancé à chaque changement de page d'un utilisateur.

D'autre part, le Web est un espace dynamique dans lequel les pages Web peuvent rapidement changer de contenu ou devenir inaccessibles. C'est pourquoi Broadway est doté d'un module de maintenance des cas et des navigations, qui est exécuté régulièrement pour les effacer lorsqu'ils ne sont plus utilisables. Une page Web est dite *obsolète* si les échecs des chargements dépassent un certain taux f_{obs}, ou si la similarité de son propre contenu (fonction $s_{Contenu}$) entre deux visites tombe en dessous d'un seuil[54] τ_{obs}. Lorsque le serveur de descriptions de pages repère une page obsolète, il

[54] Nous ne gérons pas le cas que nous pensons improbable, dans lequel le contenu de la page évoluerait par petits changements et ne serait donc pas détecté.

envoie un événement qui sera alors traité par le serveur de recommandations pour éliminer de la mémoire les navigations (et les cas associés) dont le taux de pages obsolètes atteint un seuil fixé η.

2.2 Utilisation de la plate-forme CBR*Tools

Nous détaillons dans cette section la configuration des points d'ouverture de CBR*Tools, nécessaire à la réalisation de Broadway. Plus précisément, nous indiquons la nature de cette configuration pour chaque point : spécialisation (création d'une sous-classe), instanciation (choix et paramétrage d'une classe existante), ou utilisation d'une classe par défaut. Si un même point d'ouverture est utilisé de différentes manières, nous ne retenons que la plus spécifique suivant cet ordre. Nous utilisons alors la structuration en cas d'utilisation (présentée au chapitre III, §2.3) et la notion de stéréotype de la notation UML (cf. annexe A) pour montrer la nature de chaque configuration.

2.2.1 Gestion du raisonnement

La réalisation du cycle de raisonnement pour Broadway est effectuée par la spécialisation de cinq points d'ouverture (notés en gras dans la figure Figure V-12) : la phase de recherche (Retrieve), la phase de réutilisation (Reuse), la phase d'apprentissage (Retain), l'objet de sauvegarde du raisonnement (Reasoning) et la fabrique (TimeExtendedReasonerFactory). Pour Broadway, le raisonnement doit être stoppé à la phase de révision puis, lors de la fermeture d'une navigation, l'ensemble des raisonnements liés à cette navigation est pris en compte par l'apprentissage. Dans ce but, le cycle du raisonnement est premièrement configuré pour être suspendu à la phase de révision (paramètre à donner à l'instance de la classe Reasoning). Deuxièmement, la classe Reasoning est specialisée pour pouvoir relier entre eux, les états des raisonnements effectués pour une même navigation. La gestion de ce chaînage est déléguée au client du système de raisonnement qui lui seul est en mesure de contrôler les raisonnements en fonction des navigations effectuées par les utilisateurs de Broadway. Lorsqu'une navigation est fermée, le système finalise le dernier raisonnement qui prend ainsi en compte les états des raisonnements qui lui sont reliés durant la phase d'apprentissage.

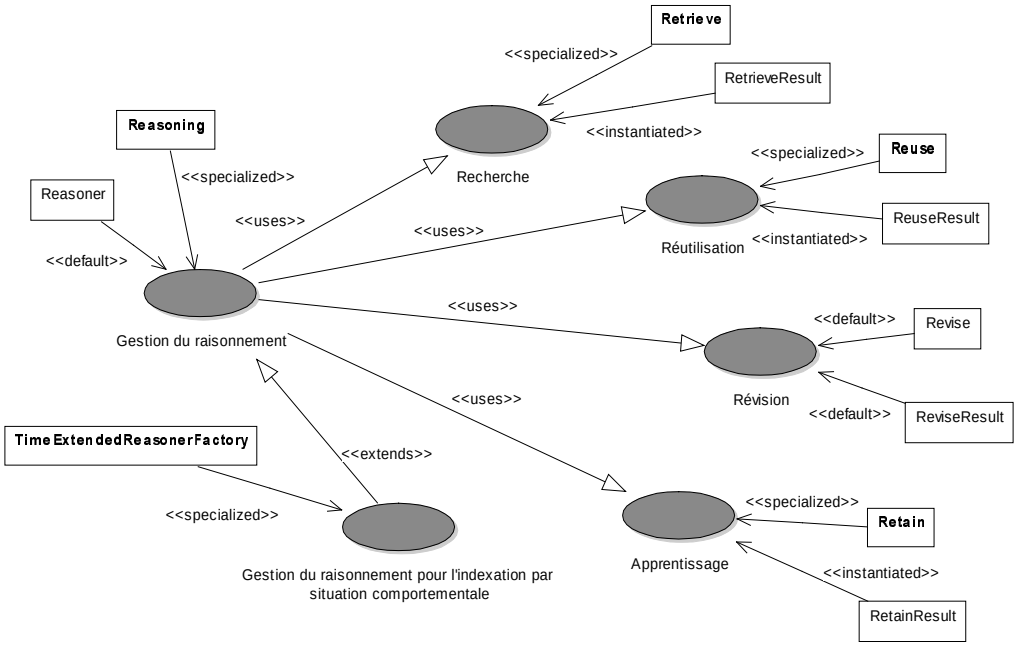

Figure V-12 : utilisation des points d'ouverture pour le raisonnement

Pour le raisonnement lui-même, la phase de recherche est tout d'abord configurée pour retourner les résultats sous la forme d'une liste de cas associés à leur similarité. La phase de réutilisation met alors en œuvre un nouveau schéma de réutilisation de recommandations que nous détaillons dans le paragraphe suivant. La phase de révision n'est pas exécutée puisqu'elle n'est pas automatisée. Cette phase correspond en effet à l'observation de la navigation effective de l'utilisateur. A la fin d'une navigation, les raisonnements sont repris pour effectuer la phase d'apprentissage qui implante de manière spécifique les directives définies (cf. §2.1.4.4).

La phase de réutilisation repose sur un nouveau schéma de réutilisation de recommandations. Ce schéma peut être configuré pour manipuler des recommandations sous la forme de pages Web pour Broadway et fournit également un cadre plus général pour la réutilisation de recommandations. Nous illustrons ainsi notre approche de capitalisation de composants pour le RàPC soutenue par CBR*Tools. Cette approche pragmatique est centrée sur la généralisation de solutions qui sont conçues pour une application particulière. Ainsi, nous obtenons non seulement une modélisation plus flexible pour l'application directement visée, mais aussi des composants réutilisables dans d'autres contextes.

Le schéma de réutilisation de recommandations suppose qu'un cas source donne comme solution une liste d'éléments évalués (recommandations). Le raisonnement doit alors former la liste des éléments évalués correspondant au cas cible courant. Ce schéma suppose également qu'un ensemble de cas a été retrouvé à la suite de la phase de recherche. La réutilisation consiste alors à réunir l'ensemble des recommandations données par les cas sources pour former la liste finale des éléments évalués qui constitue la solution au problème courant. La modélisation de cette stratégie est donnée dans la Figure V-13. De manière similaire aux autres schémas présentés, ce schéma est constitué d'un protocole ouvert (classe RecommendationsAdaptationPattern) basé sur le patron de conception *Patron de méthode*. Puis, ce protocole délègue l'analyse des éléments évalués à des objets différents suivant le patron *Stratégie* (classe RecommendationFeature). Chaque objet d'analyse retourne l'évaluation d'un élément suivant une caractéristique déterminée.

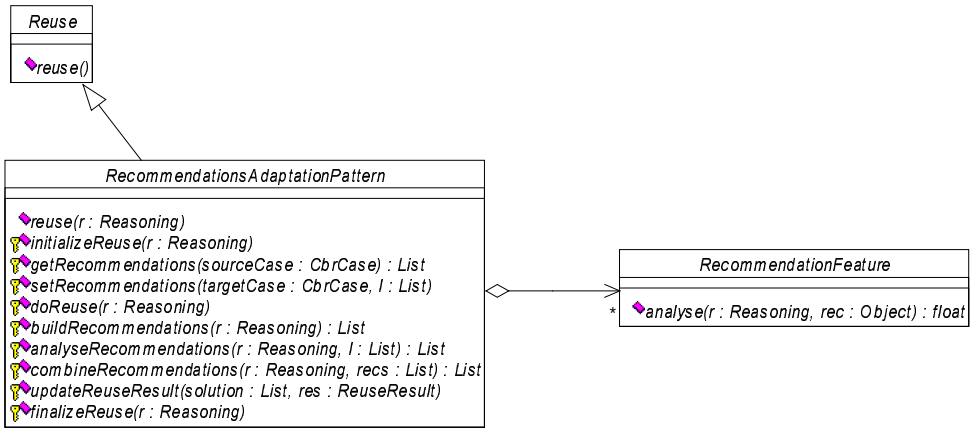

Figure V-13 : schéma de réutilisation de recommandations

Le protocole de réutilisation se décompose alors en trois étapes importantes (cf. Figure V-14) :

- constitution de la liste des éléments à étudier (méthode buildRecommendations) en réunissant les éléments donnés par les cas sources (méthode getRecommendations),
- analyse de chaque élément au regard d'une liste de caractéristiques (méthode analyseRecommendations) et agrégation des évaluations pour obtenir une évaluation globale,

- combinaison (sélection, tri) de la liste des éléments (méthode combineRecommendations) suivant les évaluations.

Cette modélisation explicite donc deux points d'ouverture spécifiques qui permettent d'une part de modifier ce protocole (constitution de la liste des éléments par union ou par intersection, analyse en combinant les caractéristiques suivant différentes fonctions d'agrégation, utilisation de différents critères de tri et de sélection pour former la liste finale) et d'autre part d'ajouter ou de retirer dynamiquement des critères d'analyse. Ainsi, nous pouvons spécialiser ce schéma dans le cadre de Broadway suivant l'algorithme défini (cf. §2.1.4.2) avec une modélisation plus stable et réutilisable.

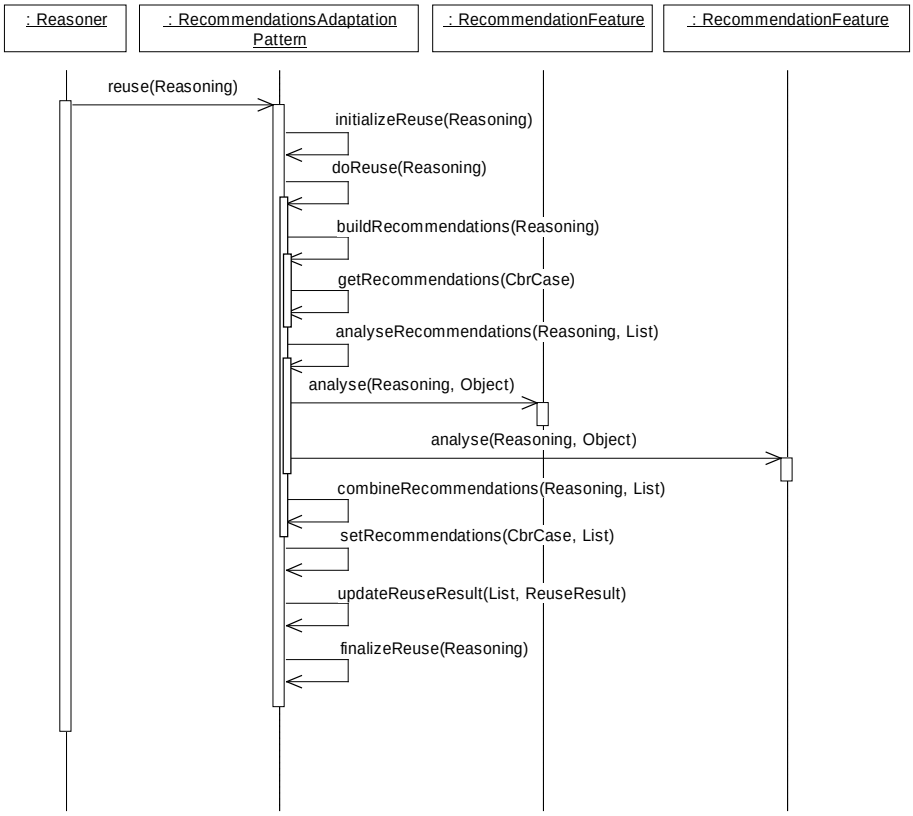

Figure V-14 : protocole du schéma de réutilisation de recommandations

2.2.2 Représentation des cas

La représentation des données d'observation, sur lesquelles repose la description des cas, nécessite uniquement la spécialisation de la représentation d'un enregistrement (cf. Figure V-15). Dans Broadway en effet, quatre variables ont été retenues (adresse, contenu, évaluation et ratio d'affichage). La spécialisation du point d'ouverture Record permet donc de déclarer ces variables. Les types standards des chroniques et des curseurs, permettant de manipuler des chroniques échantillonnées, sont alors directement utilisés.

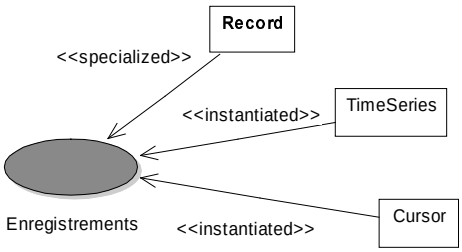

Figure V-15 : utilisation des points d'ouverture pour les données d'observation

Au-dessus de ces données d'observation, il s'agit de représenter les indices d'une situation comportementale (cf. Figure V-16). Pour la composante instantanée, le point d'ouverture CompoundIndice est utilisé pour modéliser les indices du contexte d'un enregistrement (liste de sites traversés et ensemble de mots clefs). Comme aucun autre indice instantané n'est utilisé, la structuration par défaut d'une situation instantanée qui ne comprend que le contexte, est utilisée. Pour la composante comportementale, le point d'ouverture BehaviouralSituation est spécialisé pour simplifier les opérations de manipulation des indices comportementaux (consultation du nombre de pages sélectionnées par exemple) étant donné la sémantique particulières des variables.

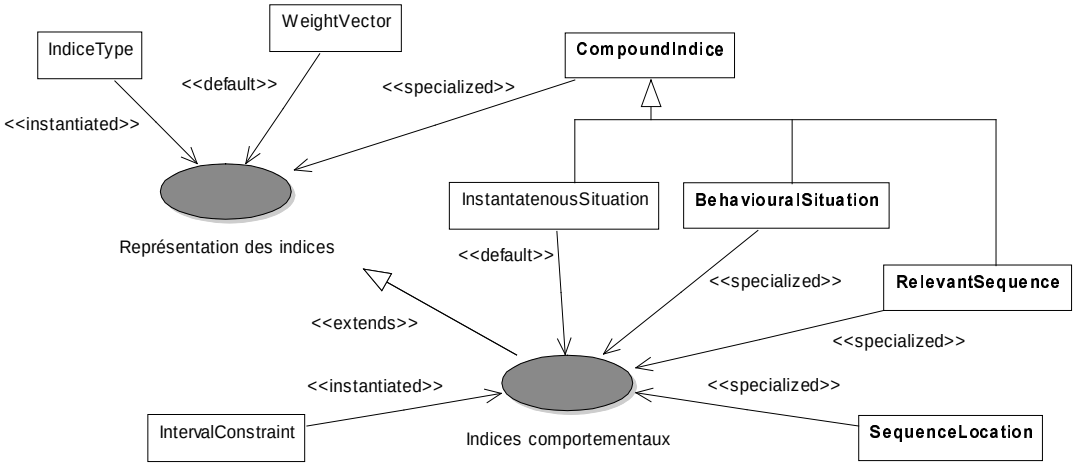

Figure V-16 : utilisation des points d'ouverture pour les indices

Le point d'ouverture RelevantSequence est également spécialisé pour déclarer les types de comportements élémentaires formés, suivant les variables, d'indices sur des entiers (évaluation), des réels (ratio d'affichage), des ensembles de mots (contenu) et des URI (adresse). Enfin, la gestion des situations comportementales nécessite également la réalisation d'une contrainte de localisation spécifique pour identifier les pages passées pertinentes. Le point d'ouverture SequenceLocation est alors spécialisé.

Les cas concrets sont alors représentés grâce à la spécialisation du point d'ouverture TimeExtendedCase qui permet de définir la partie solution (liste de pages recommandées) ainsi que des informations de gestion spécifiques à Broadway (nom de l'utilisateur, date de création, nombre de remémorations, nombre de confirmations) (cf. Figure V-17). Le patron de cas potentiels est également réalisé par la spécialisation du point d'ouverture PotentialCaseTemplate qui utilise une énumération spécifique permettant de n'instancier le patron qu'aux instants pertinents d'un enregistrement (cf. §2.1.4.1.2).

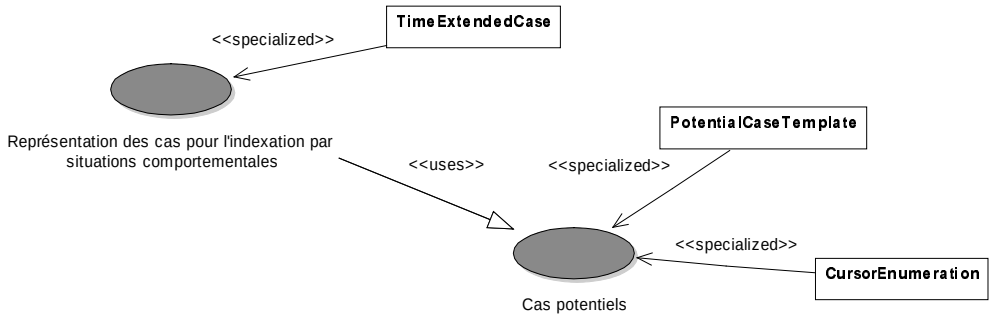

Figure V-17 : utilisation des points d'ouverture pour les cas

2.2.3 Organisation de la mémoire

L'organisation de la mémoire se décompose d'une part en l'organisation de l'indexation et l'organisation des cas. Pour l'organisation de l'indexation (cf. Figure V-19), une unique stratégie d'indexation est déclarée au niveau du point d'ouverture IndexBase qui est alors spécialisé. Cette stratégie utilise un certain nombre d'index simples en les combinant (cf. Figure V-18).

Tout d'abord, l'index s (la racine de l'arbre de composition) permet de déclencher la recherche des cas concrets, et si elle échoue, la phase de recherche de cas potentiels est effectuée. La composition des index pour la recherche des cas concrets permet de combiner en séquence les index simples c1, c2, c3 et c4. Tout d'abord, l'index c1 retourne les cas concrets filtrés à partir du contexte du cas cible. Ensuite, l'index c2 filtre ces cas sur la restriction, et l'index c3 filtre les cas restants sur les comportements élémentaires. Enfin, l'index c4 sélectionne les meilleurs cas en prenant en compte les contraintes temporelles et les similarités intermédiaires données par les index c2 et c3.

La recherche de cas potentiels est effectuée de manière similaire avec les index p1, p2, p3 et p4. Les navigations identifiées par l'index p1 sont parcourues par l'index p2 qui instancie et sélectionne les cas potentiels sur leur restriction. Les cas potentiels retournés, complètement instanciés, sont alors filtrés et sélectionnés (index p3 et p4) de manière identique aux cas concrets.

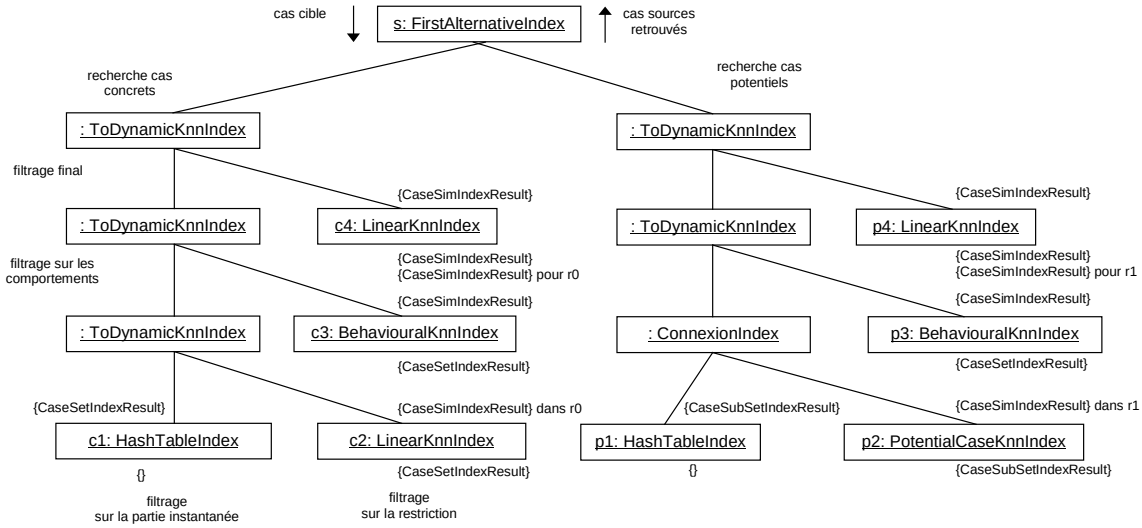

Figure V-18 : utilisation des index lors d'une recherche des cas sources

La réalisation de cette stratégie utilise tout d'abord l'index standard de classe FirstAlternativeIndex pour réunir l'indexation des cas concrets et des cas potentiels. Puis il est nécessaire de spécialiser des index à base de table de hachage (HashTableIndex) pour définir les notions spécifiques de clefs et

d'éléments (index c1 et p1). Les index c2 et p2 utilisent un filtre d'indice qui construit dynamiquement la représentation de la restriction utilisée pour les comparaisons. Les index c3 et p3 sont standards et permettent de construire les composantes comportementales homologues de chaque cas à comparer. Enfin, les index c4 et p4 sont des versions spécialisées de l'index par plus proches voisins pour prendre en compte les résultats intermédiaires de mise en correspondance.

Les similarités utilisées par ces index doivent présenter une structure identique aux indices (spécialisation du point d'ouverture Similarity). Ces mesures utilisent des fonctions élémentaires standards de similarité sauf pour l'adresse de pages Web et le ratio d'affichage (spécialisation du point d'ouverture ValueSimilarityFct pour réaliser les fonctions s_{Adresse} et $s_{\text{Affichage}}$).

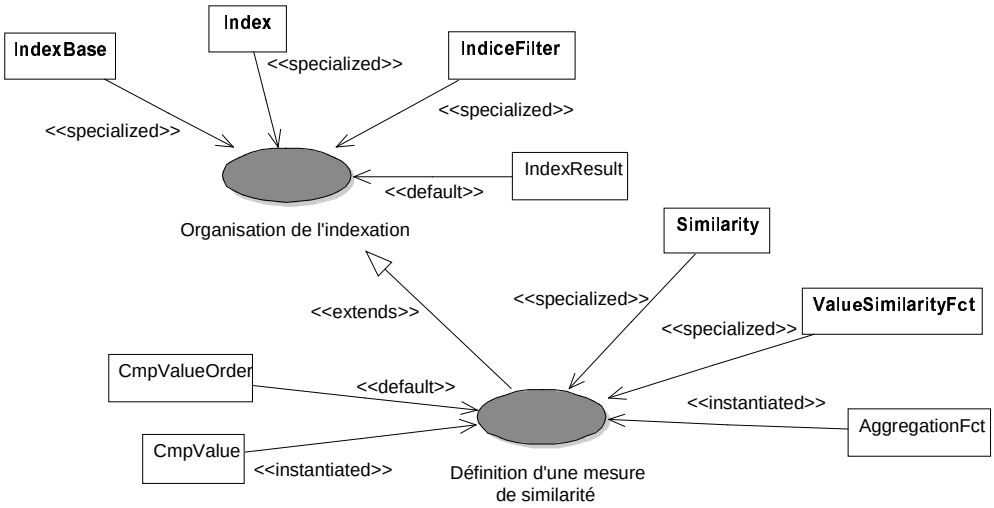

Figure V-19 : utilisation des points d'ouverture pour l'organisation de l'indexation

Enfin, pour l'organisation des cas (cf. Figure V-20), le point d'ouverture CaseBase est instancié pour utiliser la base de cas composée, formée d'une première base de cas pour les cas potentiels et d'une seconde pour les cas concrets (cf. chapitre IV, §2.2). Le point d'ouverture TemplateBase est finalement spécialisé dans l'unique but de déclarer le patron de cas potentiels utilisé par Broadway.

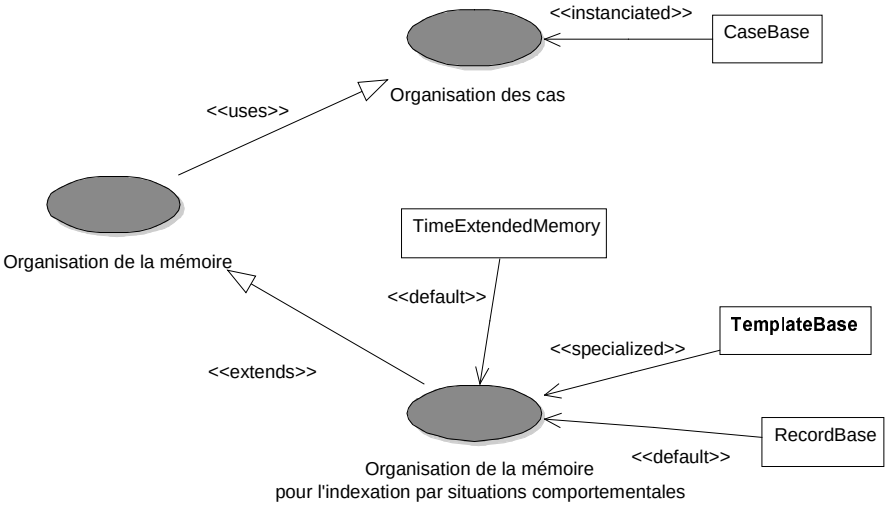

Figure V-20 : utilisation des points d'ouverture pour l'organisation des cas et de la mémoire

2.3 Bilan

La modélisation et la réalisation de Broadway nous a permis d'évaluer concrètement nos propositions du modèle d'indexation par situations comportementales, et de la plate-forme à objets CBR*Tools.

2.3.1 Modèle d'indexation par situations comportementales

L'utilisation de notre modèle de représentation dans le cadre de Broadway montre premièrement la généricité des concepts introduits. Deuxièmement, nous montrons l'apport de notre modèle pour la mise en évidence de connaissances.

2.3.1.1 *Généricité du modèle de représentation*

Pour les données d'observation, nous avons représenté les navigations des utilisateurs (enregistrements) qui sont construites avec quatre variables : adresse, contenu, évaluation et ratio d'affichage. L'évolution de ces variables est représentée par des chroniques échantillonnées pour lesquelles l'unité de temps est le changement d'une page. Enfin, une navigation possède un contexte donnant les mots clefs et les sites traversés. Nous donnons dans le Tableau V-2 le récapitulatif de ces choix de représentation tout en rappelant l'utilisation du modèle dans le cadre de la nutrition des plantes.

Concepts du modèle	*Exemple pour la nutrition des plantes*	*Utilisation dans Broadway*
Variables	rayonnement solaire, température, humidité, écart du drainage, irrigation	adresse, contenu, évaluation, ratio d'affichage
Chronique	échantillonnées	échantillonnées
Enregistrement	expérimentation	navigation
Contexte	consigne (objectif)	mots clefs et sites Web (synthèse de la navigation)

Tableau V-2 : utilisation des concepts pour la représentation des données d'observation

Concepts du modèle	*Exemple pour la nutrition des plantes*	*Utilisation dans Broadway*
composante instantanée	contexte	contexte
comportement élémentaire	séquence de valeurs (évolution)	caractéristique des pages visitées
contrainte de localisation	Ψ_P (par position), Ψ_V (par valeur)	Ψ_B (par valeur sur l'adresse ou le contenu), Ψ_R(par référence)
contrainte temporelle	relation d'occurrence des événements et des séquences de valeurs	précédence des pages visitées
patron de cas potentiels	deux patrons suivant les conditions climatiques	un seul patron sélectionnant des pages passées comme comportements et des pages futures comme solution

Tableau V-3 : utilisation des concepts pour la représentation des cas

Pour la représentation des cas et des situations comportementales (cf. Tableau V-3), nous avons mis en pratique le concept de situation instantanée qui représente le contexte de la navigation. Pour la composante comportementale, nous avons pu sélectionner les pages passées caractérisant le

comportement des utilisateurs avec des comportements élémentaires. Le formalisme que nous avons utilisé nous a également permis de définir une nouvelle contrainte de localisation (Ψ_B) pour identifier de manière spécifique des pages homologues dans un cas cible. Les contraintes temporelles ont été utilisées pour ordonner les pages sélectionnées. Enfin, nous avons défini un patron de cas potentiels pour l'extraction automatique d'expériences pertinentes passées pour guider le raisonnement.

2.3.1.2 *Mise en évidence de cas : vers la découverte des connaissances*

La réalisation concrète de Broadway nous a permis de valider les mécanismes proposés pour les deux premières étapes du processus de découverte des connaissances (cf. chapitre II, §1.1.2.2 et §3) :

1. *Extraction de cas potentiels à partir des données brutes*. L'extraction d'enseignements est effectuée grâce à l'instanciation d'un *patron de cas potentiels*. Ce patron permet d'extraire à partir des navigations, des enseignements liés à un groupe d'utilisateurs. Cette extraction utilise deux hypothèses principales pour caractériser les comportements des utilisateurs : importance des toutes dernières pages visitées, et importance des pages passées évaluées explicitement ou implicitement sur un critère lié au temps d'affichage. Le patron permet alors de structurer en comportements élémentaires et en contraintes temporelles les indices effectifs des cas potentiels identifiés.
2. *Mise à jour des cas concrets*. Un fois identifiés, les cas potentiels jugés utiles par les décisions d'apprentissage sont stockés en mémoire sous la forme de cas concrets. Cette mémorisation n'est effectuée que lorsque les cas concrets déjà identifiés ne sont pas suffisants. Les cas concrets sont alors mis à jour suite aux différents raisonnements qui les réutilisent : nombre de remémorations, nombre de confirmations. Ainsi, des informations liées à la qualité de chaque cas sont calculées et sont utilisées pour guider les réutilisations.

Nous n'avons pas abordé l'interprétation des cas par analyse des indices. Toutefois, nous pouvons aisément supposer que l'étude des cas concrets est facilitée par leur structure explicite associant une liste de pages passées sélectionnées, à un ensemble de pages recommandées. Ainsi, l'interprétation des cas montrant un fort degré de qualité (nombre de confirmations sur le nombre de remémorations) pourrait mener à l'identification de navigations typiques pour mieux comprendre le processus de navigation.

Toutefois les mécanismes proposés impliquent la création de nombreux cas concrets. Ces créations sont rendues possibles car chaque cas concret demande peu de mémoire pour sa représentation, étant donné que les informations coûteuses sont partagées dans les navigations. De plus, comme nous l'avons montré, la recherche s'effectue par filtrages successifs pour maintenir l'efficacité du système. Enfin, ces mécanismes ont été conçus dans la perspective de l'analyse des cas concrets pour créer des cas abstraits. Ces derniers permettraient de généraliser et de remplacer des cas concrets pour réduire la taille de la base de cas. Grâce à notre modèle d'indexation, la création de ces cas abstraits bénéficierait du travail réalisé dans l'identification et la mise à jour des cas concrets.

2.3.2 Plate-forme à objets CBR*Tools

Nous évaluons l'utilisation de CBR*Tools suivant deux critères. Dans le premier critère, nous estimons le degré d'expertise nécessaire de la plate-forme pour réaliser Broadway. De manière générale, une plate-forme demandant peu d'expertise est plus facile à utiliser. Dans le second critère, nous estimons plus concrètement la réutilisation de la modélisation et du code pour Broadway. L'objectif d'une plate-forme est de maximiser cette réutilisation et de limiter la création de nouvelles classes.

2.3.2.1 *Evaluation du degré d'expertise nécessaire de CBR*Tools pour Broadway*

L'estimation du degré requis d'expertise, relatif au fonctionnement de notre plate-forme pour la réalisation de Broadway, repose sur la complexité des connaissances nécessaires à chaque type d'utilisation d'un point d'ouverture :

- une spécialisation demande en général une compréhension approfondie du point d'ouverture,
- une instanciation demande uniquement la connaissance des paramètres et du comportement global du composant réalisant un point d'ouverture,
- une utilisation par défaut reste transparente pour l'utilisateur.

Nous rappelons que, si un point d'ouverture est utilisé suivant différentes méthodes, nous ne retenons que celle qui demande le plus d'expertise. Cette estimation montre que la réalisation de Broadway nécessite une connaissance approfondie d'uniquement 50% des points d'ouverture de la plate-forme (cf. Tableau V-4).

Type d'utilisation	*Axe « gestion du raisonnement »*	*Axe « Représentation des cas »*	*Axe « Organisation de la mémoire »*	*Total*
Spécialisation	5	8	6	*19 (50%)*
Instanciation	3	4	3	*10 (26%)*
Par défaut	3	2	4	*9 (24%)*
Total	*11*	*14*	*13*	*38 (100%)*

Tableau V-4 : utilisation des points d'ouverture de CBR*Tools

Parmi les spécialisations effectuées, cinq d'entre elles ne sont pas considérées obligatoires (cf. chapitre IV, §3.2.2.1) mais ont été nécessaires car il n'existait pas de valeurs par défaut ou de composants utilisables par instanciation :

- chaînage des états des raisonnements associés à une navigation (Reasoning),
- contrainte de localisation spécifique Ψ_B (SequenceLocation),
- parcours des enregistrements adapté au patron de cas potentiels (CursorEnumeration),
- configuration des index (index),
- ajout de fonctions élémentaires de similarité (ValueSimilarityFct).

2.3.2.2 *Evaluation de l'aide apportée par CBR*Tools pour la réalisation de Broadway*

Sur le plan de la réalisation, nous estimons l'apport de CBR*Tools en termes de modélisation et de réutilisation de code. Dans ce but, nous partageons les classes nécessaires à Broadway en quatre types (cf. Tableau V-5) :

1. les classes créées uniquement par spécialisation d'interfaces de CBR*Tools,
2. les classes créées par spécialisation de classes abstraites ou concrètes de CBR*Tools,
3. les classes existantes de CBR*Tools utilisées uniquement par instanciation, et
4. les classes créées spécifiquement qui n'ont pas de lien de spécialisation avec des classes existantes.

Le nombre de spécialisations (1+2) reste toujours supérieur ou égal au nombre de points d'ouverture spécialisés, car un même point d'ouverture peut être spécialisé à plusieurs reprises. Par exemple dans Broadway, la configuration des index nécessite trois spécialisations (configuration des deux tables de hachage et de l'algorithme par plus proches voisins pour le filtrage final). Le nombre de classes différentes utilisées par instanciation directe est également supérieur ou égal au nombre de

points d'ouverture instanciés. En effet, un même point d'ouverture peut être instancié pour des raisons différentes en sélectionnant plusieurs sous-classes. De plus, ce groupe de classes recouvre aussi : les classes des points d'ouverture spécialisés mais aussi instanciés (6 classes différentes sont également instanciées pour le point d'ouverture Index), les classes annexes utilisées pour les paramètres des points d'ouverture, et les classes des points d'ouverture utilisés par défaut (si la valeur par défaut n'est pas null). Enfin, dans Broadway, seules les classes nécessaires à la représentation du schéma de réutilisation sont nouvelles.

Types de classes	***Axe « gestion du raisonnement »***	***Axe « Représentation des cas »***	***Axe « Organisation de la mémoire »***	***Total***
Spécialisation d'interfaces	2	3	3	*8 (11%)*
Spécialisation de classes abstraites ou concrètes	3	11	8	*22 (30%)*
Instanciation	5	13	17	*35 (48%)*
Création	8	0	0	*8 (11%)*
Total	*18*	*27*	*28*	*73 (100%)*

Tableau V-5 : types de classes utilisées dans Broadway

En définitive, CBR*Tools facilite la modélisation d'une application comme Broadway grâce à son architecture dans laquelle les comportements ont déjà été analysés et assignés à différentes classes (89% des classes sont issues de spécialisation ou d'instanciation de classes existantes). De plus, des parties algorithmiques plus lourdes sont réutilisées, principalement dans les index et les similarités (78% des classes réutilisent du code à travers des spécialisations de classes concrètes ou abstraites, et des instanciations).

3 Proposition d'un nouveau type d'assistants à la navigation

Avec notre système Broadway, nous introduisons un nouveau type d'assistants à la navigation. Nous proposons en effet la réutilisation de *comportements observés par variables* pour calculer les recommandations. Les assistants de navigation existants ne permettent pas une observation aussi détaillée et flexible. De plus, le cadre générique formé par notre modèle d'indexation nous permet d'aborder des problèmes d'assistance dans des contextes différents. Dans cette section, nous détaillons tout d'abord une évaluation expérimentale de notre prototype Broadway, menée avec des utilisateurs dans un contexte précis de recherche d'informations. Puis, nous présentons l'originalité de Broadway par rapport aux approches existantes.

3.1 Evaluation expérimentale de Broadway

Une expérimentation avec des utilisateurs réels a été menée par des étudiants en DESS d'ergonomie pour évaluer si Broadway assiste effectivement les utilisateurs dans leurs tâches de recherche d'informations sur le Web (Hébraud *et al.*, 1998). Nous introduisons tout d'abord les objectifs et la procédure d'expérimentation puis nous présentons les résultats de cette expérimentation. Enfin, nous donnons la configuration et l'adaptation de Broadway qui a été nécessaire pour cette expérimentation.

3.1.1 Objectifs et procédure d'expérimentation

La procédure d'expérimentation a été déterminée en toute autonomie par des élèves ergonomes qui ont reçu au préalable une formation d'une journée à l'utilisation et aux concepts de Broadway. Pour cette expérimentation, dix étudiants en psychologie ont été choisis pour jouer le rôle d'utilisateurs de Broadway qui veulent rechercher des informations sur le Web. Deux groupes dont les connaissances du Web et des navigateurs sont équivalentes, ont alors été formés : le premier groupe aura accès aux recommandations de Broadway contrairement au deuxième. Toutefois, tous les sujets devront systématiquement évaluer les pages visitées avec la barre d'outils de Broadway[55]. Tous les sujets auront alors le même objectif de recherche :

« *Vous devez rechercher des informations sur la vie de Patrick Sommier. Vous savez qu'il s'agit d'un artiste contemporain qui travaille probablement dans le monde du théâtre* »

Cet objectif a été choisi avec soin : aucun des sujets n'avait de connaissance préalable sur cet artiste et sa biographie n'était pas directement accessible depuis les cinq premiers liens donnés par le moteur de recherche Altavista pour le mot clef « spectacles ». Ces cinq liens constitueront les points de départ des navigations des sujets. Des tests préliminaires ont montré que l'accès à l'information demandée était accessible sans assistance depuis ces liens en 30 minutes environ. Ainsi, le dépassement de ce délai sera considéré comme un échec de la recherche. Une base de cas a également été construite au préalable avec des exemples de navigations réussies. L'objectif principal de cette expérimentation est de comparer la qualité des navigations entre le groupe utilisant Broadway et celui sans. Plus précisément, cinq hypothèses sont à évaluer (Hébraud *et al.*, 1998) :

1. Les sujets de la condition « Avec Broadway » ont un taux de réussite à la tâche supérieur à ceux de la condition « Sans Broadway » (indicateur : nombre de réussites).
2. Les sujets de la condition « Avec Broadway » retrouve l'information plus rapidement que ceux de la condition « Sans Broadway » (indicateur : temps total).
3. Les sujets de la condition « Avec Broadway » visitent moins de pages Web que ceux de la condition « Sans Broadway » (indicateur : nombre de pages visitées).
4. Les évaluations portées sur les pages Web sont homogènes pour les deux groupes (indicateur : évaluations affectées à une même page par tous les sujets).
5. Les sujets de la condition « Avec Broadway » sont globalement plus satisfaits que ceux de la condition « Sans Broadway » (indicateur : somme des évaluations sur le nombre de pages évaluées).

3.1.2 Résultats de l'expérimentation

Les résultats de l'expérimentation sur la vérification des trois premières hypothèses sont donnés dans le Tableau V-6. Nous constatons que, dans le contexte choisi, l'utilisation de Broadway a effectivement facilité la tâche de recherche d'information : il y a plus de succès, la durée des navigations est réduite et leur longueur également.

Pour l'hypothèse 4, l'expérimentation a permis de montrer que les évaluations ne sont pas cohérentes dans certains cas. En effet, bien que l'objectif de recherche ait été précisément défini, il semble que certains sujets aient évalué une page en se basant sur le thème général abordé, alors que d'autres ont évalué positivement uniquement les pages ayant directement un rapport avec la requête. Toutefois, il y a un consensus sur l'évaluation la plus positive. Pour pallier ce problème, nous pensons étudier certaines modifications pour une version future de Broadway : réduction du nombre de degrés d'évaluation, définition plus claire de la sémantique des évaluations, ou normalisation des évaluations

[55] L'évaluation systématique n'est pas requise dans une utilisation courante de Broadway.

en tenant compte des pages évaluées le plus positivement. Enfin, l'expérimentation n'a pu valider l'hypothèse 5 au vu du nombre restreint d'utilisateurs, bien que les résultats bruts montrent une satisfaction globale plus grande pour les sujets ayant bénéficié de Broadway.

Critères d'évaluation	***Avec Broadway (4 sujets)***	***Sans Broadway (6 sujets)***
Nombre de succès	75% (3/4)	33% (2/6)
Durée des navigations réussies (moyenne)	18 min	24 min
Longueur des navigations réussies (moyenne)	19 pages	39 pages

Tableau V-6 : synthèse des résultats pour les trois premières hypothèses

Cette expérimentation a permis une première évaluation de Broadway par des utilisateurs qui n'étaient pas liés au développement du système. Les résultats confirment la validité de notre approche dans les conditions restreintes de l'expérimentation, mais d'autres tests restent nécessaires pour les généraliser.

3.1.3 Configuration de Broadway utilisée

Durant l'expérimentation, les sujets localisés à l'université de Provence à Aix-en-Provence ont accédé pendant plusieurs jours au système Broadway situé à l'INRIA Sophia Antipolis au travers du réseau Internet. Même si deux groupes avaient été créés et que seulement l'un d'entre eux eut accès aux recommandations, Broadway fut nécessaire dans les deux cas pour observer les comportements des sujets et pour leur permettre d'évaluer les pages. Pour ne pas engendrer des conflits entrent les deux groupes au niveau des données enregistrées et de la base de cas générée, deux systèmes Broadway avaient été lancés simultanément. Au moment de l'expérimentation, le prototype de Broadway n'intégrait pas l'ensemble des mécanismes que nous avons décrits dans ce chapitre. Notamment, le serveur de recommandations utilisait uniquement des cas potentiels, et les fonctions d'analyse de la base de cas pour gérer l'obsolescence des navigations et des cas n'avaient pas été implantées. L'analyse des pages Web était également plus simple et ne disposait pas des fonctions de filtrage des mots non significatifs et de la réduction des mots à leur racine.

Pour réaliser l'expérimentation, les ergonomes chargés de l'étude avaient exprimé le besoin de pouvoir analyser finement les navigations des sujets mais aussi l'activité du serveur de recommandations et les choix de recommandations effectués. Aucun module n'avait jusqu'alors été prévu pour ce type d'analyse, cependant l'architecture ouverte de Broadway nous a facilement permis d'ajouter de tels modules. Deux modules ont alors été conçus et implantés : un *traceur* et un *enregistreur*. Ces deux modules sont des processus distants qui se connectent tout simplement au répartiteur d'événements lors du lancement de Broadway.

Le traceur permet de sauvegarder dans un fichier les événements au fur et à mesure qu'ils se produisent, dans un format brut qui peut être analysé directement sous le tableur Excel par exemple. Chaque ligne du fichier donne le nom du module émetteur, la date, le nom de l'utilisateur, la classe de l'événement et les informations caractérisant le contenu de l'événement. L'extrait présenté dans la Figure V-21 permet de retracer l'activité de l'utilisateur et du système de la façon suivante. Tout d'abord, la barre d'outils d'un utilisateur nommée baseErgo commence par envoyer une évaluation. Puis, suite à la sélection d'un hyperlien dans le navigateur, le proxy charge une nouvelle page.

L'ancienne page dans le navigateur est alors effacée et remplacée par une nouvelle : l'observateur (*matcher*) envoie alors deux événements pour l'indiquer. Le serveur de recommandations (*recommender*) activé par l'affichage d'une nouvelle page commence son raisonnement puis envoie un événement indiquant les résultats. Enfin, l'utilisateur sélectionne une des recommandations dans le contrôleur (*manager*).

```
toolbar;13/03/1998 15:22:09;baseErgo;broadway.events.PageEvaluationEvent;
        url=http://www.grandtheatre.qc.ca/pour_enfants.html, eval=1
proxy;13/03/1998 15:27:51;baseErgo;broadway.events.PageEvent;
        url=http://www.grandtheatre.qc.ca/programmation_chrono.html, state=LOADED
watcher;13/03/1998 15:24:16;baseErgo;broadway.events.PageEvent;
        url=http://www.grandtheatre.qc.ca/pour_enfants.html, state=STOP
watcher;13/03/1998 15:24:22;baseErgo;broadway.events.PageEvent;
        url=http://www.grandtheatre.qc.ca/programmation_chrono.html, state=START
recommender;13/03/1998 15:28:00;baseErgo;broadway.events.AdviceStartEvent;state=START
recommender;13/03/1998 15:28:03;baseErgo;broadway.events.AdviceResultEvent;
        state=RESULT, duration=2667, nb=2,
        url0=http://www.brookes.ac.uk/schools/sol/home/index.htm, rate0=0.44375,
        url1=http://www.brookes.ac.uk/~p0030818/cal/1674.html, rate1=0.44375
manager;13/03/1998 15:25:04;baseErgo;broadway.events.AdviceSelectionEvent;
        url=http://www.brookes.ac.uk/schools/sol/home/index.htm
```

Figure V-21 : extrait de l'enregistrement des événements[56]

```
Navigation from baseErgo

size: 20
start date: 13/03/1998 15:20:23
end date: 13/03/1998 15:34:09

1. URL = http://www.pariserve.tm.fr/sortir/dinersp.htm

        ○ keywords = DINERS, SPECTACLES
        ○ size = 13714
        ○ display = 1920
        ○ ratio = 0.14000292
        ○ eval = 0
        ○ selected = –
        ○ advice :

[....]

5. URL = http://www.grandtheatre.qc.ca/programmation_chrono.html

        ○ keywords = PROGRAMMATION, CHRONOLOGIQUE
        ○ size = 31111
        ○ display = 22360
        ○ ratio = 0.71871686
        ○ eval = 0
        ○ selected = http://www.brookes.ac.uk/schools/sol/home/index.htm
        ○ advice :
            □ 0.44375 http://www.brookes.ac.uk/schools/sol/home/index.htm
            □ 0.44375 http://www.brookes.ac.uk/~p0030818/cal/1674.html
```

Figure V-22 : extrait du fichier généré par l'enregistreur pour les événements de la Figure V-21

Contrairement au traceur, l'enregistreur effectue un traitement des événements en les regroupant par utilisateur et en les présentant de manière plus claire. L'enregistreur permet en effet de générer la synthèse d'une navigation dans un fichier HTML lorsqu'une navigation est fermée (cf. Figure V-22). Ainsi, la taille de la navigation, ses dates de début et de fin, et toutes les pages visitées sont facilement

[56] Les informations sont ici formatées avec des retours à la ligne pour plus de clarté, mais en réalité chaque événement n'est représenté que par une seule ligne.

accessibles. De plus, pour chaque page, différentes informations sont disponibles : URL, mots clefs, taille en octets, temps d'affichage en millisecondes, ratio d'affichage, évaluation (de –1 à 2), recommandation sélectionnée et liste des recommandations obtenues.

Ces deux modules sont complémentaires : alors que l'enregistreur est dépendant de la création d'une navigation et propose une vue synthétique mais réduite d'une navigation, le traceur enregistre tous les événements même lorsqu'une navigation n'a pas été démarrée. Ces deux modules ont été nécessaires au dépouillement des données[57]. Pour l'expérimentation, tous les fichiers créés par le traceur et l'enregistreur ont été placés automatiquement dans l'espace public du proxy utilisé dans Broadway et qui opère dans ce cas comme un serveur HTTP classique. Ainsi, les personnes chargées du dépouillement ont pu librement accéder aux données distantes avec leur navigateur Web.

La réalisation de ces deux modules montre enfin la stabilité de l'architecture ouverte choisie au regard de spécifications qui évoluent. Ces évolutions sont naturelles dans un contexte réel d'expérimentation et correspondent exactement aux contraintes que nous voulions prendre en compte.

3.2 Comparaison avec les systèmes existants d'aide à la navigation

Nous classons les différents assistants existants de navigation en trois grands groupes suivant les techniques utilisées :

- raisonnement à partir de cas : Radix (Corvaisier *et al.,* 1997), Hypercase (Micarelli & Sciarrone, 1996) et Hospitext (Elkassar & Charlet, 1997).
- autres techniques d'apprentissage automatique : WebWatcher (Armstrong *et al.,* 1997), PersonnalWebWatcher (Mladenic, 1996), Letizia (Lieberman, 1995), WebMate (Chen & Sycara, 1998).
- statistiques et comparaisons : approche de Yan *et al*. (Yan *et al.,* 1996), WBI (Barrett *et al.,* 1997), FindMe (Burke *et al.,* 1997) et Footprints (Wexelblat & Maes, 1997).

Ces assistants sont récents (les premières publications datent de 1995) et la plupart ont été réalisés de manière concourante à nos travaux. Nous montrons que Broadway met en œuvre une approche originale permettant à la fois la prise en compte de comportements structurés par variables dans la situation de recommandation, et l'apprentissage à partir d'un groupe d'utilisateurs, tout en proposant une architecture ouverte et adaptée au Web. La synthèse de cette comparaison est donnée dans le Tableau V-7.

3.2.1 Situation de recommandation

Un assistant de navigation observe l'utilisateur et peut générer des recommandations, à un instant précis. Trois approches sont possibles pour constituer la situation servant à déterminer les recommandations : la page Web courante et/ou l'intérêt exprimé par l'utilisateur (*contexte local*), un résumé automatique de la navigation et de l'intérêt de l'utilisateur (*profil utilisateur*), ou un ensemble d'observations issues des navigations (*comportements*).

Six assistants utilisent un contexte local : WebWatcher, PersonnalWebWatcher, Footprints, FindMe, WBI et Hospitext. WebWatcher et PersonnalWebWatcher utilisent la page courante ainsi qu'un ensemble de mots clefs donné par l'utilisateur au début de sa recherche. WebWatcher met en valeur les hyperliens recommandés de la page courante. Footprints permet de visualiser un graphe

[57] Le traceur intervient en fait en tant que sauvegarde des événements émis. Cette précaution a été très utile dans la mesure où, suite à des erreurs de manipulation qui ont conduit à ne pas démarrer les navigations à observer, l'enregistreur n'a pu générer les fichiers des navigations.

dans lequel chaque nœud représente une page et les nœuds sont reliés entre eux par des liens représentant les déplacements passés des utilisateurs. De plus, les liens sont dessinés avec différentes couleurs pour montrer leur fréquence d'utilisation. Ainsi, un utilisateur peut visualiser ce graphe pour repérer sa position courante et choisir de suivre un de ces liens en cliquant directement sur le graphe. L'idée soutenue par ce système est qu'un lien est intéressant si un grand nombre d'utilisateurs l'ont suivi. FindMe est un type d'approche permettant de guider l'utilisateur dans la recherche d'éléments précis. Cette technique a été appliquée pour la sélection de voitures, de vidéos, de restaurants ou d'appartements. Lorsqu'un utilisateur débute sa navigation, il indique ses premières contraintes et le système peut alors lui proposer une solution initiale. Puis, l'utilisateur navigue par modifications successives de ces contraintes par rapport à l'élément courant sélectionné. Par exemple, l'utilisateur peut indiquer qu'il cherche un appartement moins cher, plus grand, plus beau… Le système génère alors une nouvelle proposition en effectuant une similarité entre la proposition courante et les éléments existants, tout en prenant en compte la modification demandée. WBI utilise un ensemble de séquences types issues des navigations d'un utilisateur. Il propose alors la page finale d'une séquence dès que l'utilisateur visualise la première page. Enfin, l'objectif d'Hospitext est d'assister différents types d'utilisateurs durant leur consultation de dossiers médicaux de patients qui sont disponibles dans un hypermédia. A partir d'un objectif de navigation, il propose une séquence de documents à visiter.

Deux assistants, Letizia et WebMate, construisent un profil utilisateur à partir des navigations observées et des interactions avec le système. Letizia est un agent qui assiste un utilisateur en navigant en parallèle et de manière autonome depuis la page courante en cours de visualisation. Letizia utilise le temps passé par l'utilisateur à lire la page pour explorer les environs et anticiper ses besoins. Cet assistant construit une description instantanée de l'intérêt de l'utilisateur à partir de ses actions durant la navigation courante (profil utilisateur formé d'un vecteur de mots clefs). WebMate est un agent qui apprend le profil d'un utilisateur suivant un ensemble de pages appréciées lors de ses navigations sur le Web. Un profil est dans ce système constitué d'un ensemble de vecteurs de mots clefs pondérés. Letizia et WebMate sont alors capables de filtrer et de recommander des pages qui dépassent un seuil de similarité avec le profil.

Enfin, trois derniers assistants prennent en compte un ensemble d'événements issus de la navigation de l'utilisateur : Radix, Hypercase et l'approche de Yan *et al*. Radix, que nous avons déjà présenté (cf. chapitre II, §1.1.1.3.5), est basé sur une description structurée des sessions de navigation et utilise une fonction de similarité prenant en compte la succession ordonnée des actions de l'utilisateur (changement de pages) pour calculer les recommandations. Hypercase utilise également l'ensemble des pages déjà visitées comme indice de son raisonnement pour déterminer les recommandations. Enfin, l'approche de Yan *et al*. propose l'utilisation de navigations typiques construites statistiquement. Ainsi, la navigation d'un utilisateur (ensemble de pages visitées) est tout d'abord analysée et classée dans l'une de ces navigations typiques. Puis, le serveur propose les pages de cette navigation qui n'ont pas encore été visitées. Cependant, comme dans Hypercase, l'ordre d'accès aux pages n'est pas pris en compte[58].

Broadway se situe dans cette troisième classe d'assistants et propose plus particulièrement la prise en compte de *comportements observés par variables*. Broadway est en effet basé sur l'observation des comportements structurés en variables : adresse, contenu des pages, évaluation explicite et évaluation implicite. Notre approche est originale car elle ne se limite pas à ces quatre variables, et grâce à notre modèle d'indexation, elle offre un cadre générique pour la prise en compte de tout type de variables. Par exemple, il est possible d'ajouter simplement dans le prototype actuel des variables portant sur

[58] Les auteurs reconnaissent l'importance de prendre en compte cet ordre et l'envisagent dans leurs travaux futurs.

des saisies de formulaire, ou sur l'envoi de requêtes à des moteurs de recherche. De plus, notre approche est également applicable dans d'autres contextes d'assistance à la navigation dans lesquels les variables prennent des sémantiques spécifiques :

- aide à la navigation sur un site de commerce électronique : descriptif textuel d'un article, sélection d'un article dans son panier d'achat,
- aide à la navigation dans une hiérarchie thématique (par exemple Yahoo) : descriptif textuel d'un thème, type de lien hypertexte sélectionné (catégorie transversale, sous-catégorie ou super-catégorie), sélection d'un site.

Toutefois, la prise en compte de ces variables est plus coûteuse qu'une approche par contexte local, par profil utilisateur ou par comportements. Nous mettons alors en œuvre notre modèle d'indexation qui permet de gérer efficacement les situations comportementales (filtrage progressif lors de la recherche et partage des données brutes entre les cas concrets).

Assistants de navigation	***Situation de recommandation***	***Apprentissage***	***Architecture***		
			indépendance navigateur	*assistance multi-sites*	*ouverture*
FindMe (Burke *et al.*, 1997)	contexte local	pas d'apprentissage	✓	-	✓
Footprints (Wexelblat & Maes, 1997)	contexte local	groupe	✓	-	-
Hospitext (Elkassar & Charlet, 1997)	contexte local	groupe (RàPC)	✓	-	-
WBI (Barrett *et al.*, 1997)	contexte local	unique utilisateur	✓	✓ (proxy)	✓
WebWatcher (Armstrong *et al.*, 1997)	contexte local	groupe	✓	✓ (redirection des liens)	✓
PersonalWebWatcher (Mladenic, 1996)	contexte local	unique utilisateur	✓	✓ (proxy)	✓
Letizia (Lieberman, 1995)	profil utilisateur	unique utilisateur	✓	✓	-
WebMate (Chen & Sycara, 1998)	profil utilisateur	unique utilisateur	✓	✓ (proxy)	-
Hypercase (Micarelli & Sciarrone, 1996)	comportements	pas d'apprentissage (RàPC)	✓	-	-
Radix (Corvaisier *et al.*, 1997)	comportements	groupe (RàPC)	-	✓	-
Yan *et al* (Yan *et al.*, 1996)	comportements	groupe	✓	-	-
Broadway	**comportements observés par variables**	**groupe (RàPC)**	✓	✓ (proxy)	✓

Tableau V-7 : comparatif des assistants de navigation existants avec Broadway

3.2.2 Apprentissage

Un assistant de navigation met à jour ses connaissances grâce à un mécanisme d'apprentissage. Trois types d'apprentissage existent : le cas limite où il n'y a *pas d'apprentissage*, l'apprentissage centré sur un *unique utilisateur* et l'apprentissage à partir d'un *groupe* d'utilisateurs.

Hypercase est un assistant qui utilise des techniques de raisonnement à partir de cas. Cependant, cet assistant ne peut pas apprendre et s'enrichir à partir des expériences des utilisateurs, puisqu'il utilise uniquement un ensemble de cas prédéfinis construits par des experts. FindMe ne met pas non plus à jour ses connaissances suivant les comportements des utilisateurs, seuls de nouveaux éléments (voitures, vidéos, appartements…) peuvent être ajoutés dans sa base.

Letizia, WebMate, PersonnalWebWatcher et WBI ne prennent en compte qu'un seul utilisateur. Letizia et WebMate apprennent le profil de leur unique utilisateur. Contrairement à WebMate, Letizia n'apprend pas à améliorer ses recommandations au cours des différentes navigations, puisque le profil n'est pas sauvé ni mis à jour. PersonnalWebWatcher a pour but explicite d'assister un unique utilisateur. WBI enregistre les navigations d'un utilisateur et les examine pour en extraire des séquences qui se produisent fréquemment. Ce fonctionnement permet alors de mettre en évidence des raccourcis dans les navigations d'un utilisateur.

Enfin, les autres approches mettent à profit les navigations d'un groupe d'utilisateurs pour améliorer leur processus de recommandation. Radix enregistre de nouveaux cas dans sa base. Hospitext nécessite le recueil de navigations commentées par les utilisateurs pour connaître le type des informations recherchées à chaque choix d'un lien. Puis, ce système est doté d'un module d'analyse pour extraire automatiquement, à partir de ces observations, des navigations abstraites liant des types de documents et des types d'éléments structurels, plutôt que des documents en absolu. Les navigations sont ainsi réutilisables dans différents contextes. WebWatcher suit un groupe d'utilisateurs au cours de leurs navigations et tente d'apprendre la fonction suivante avec différentes méthodes :

$$\text{QualitéLien} : \text{Page} \times \text{Intérêt} \times \text{Lien} \rightarrow [0 \ldots 1]$$

Footprints analyse les logs HTTP d'un serveur pour former le graphe de synthèse des navigations effectuées par les utilisateurs. Enfin, l'approche de Yan *et al*. analyse également les logs d'un serveur HTTP pour construire statistiquement des navigations typiques (*clustering*).

Comme ces derniers travaux, Broadway met à jour ses connaissances à partir des comportements d'un groupe d'utilisateurs (enregistrement et réutilisation de cas provenant d'utilisateurs différents, cf. §2.1.4.4). Les cas représentent des connaissances issues d'utilisateurs différents et sont réutilisés durant le raisonnement.

3.2.3 Architecture : indépendance des navigateurs, assistance multi-sites et ouverture

Sur le plan de l'architecture des assistants, trois critères sont importants dans le cadre de nos contraintes d'application : *l'indépendance* vis-à-vis des navigateurs utilisés, *l'assistance multi-sites* sur le Web et *l'ouverture* du système pour modifier son comportement.

Dans Radix, l'observation des actions de l'utilisateur, comme la sélection d'un signet, la spécification manuelle d'une adresse, l'utilisation des boutons de navigation (« précédant » et « suivant »), est nécessaire pour représenter une navigation et ses composants. C'est pourquoi ce système utilise un navigateur avec des fonctions personnalisées : son utilisation est alors restreinte à une plate-forme logicielle spécifique. Les autres assistants présentés peuvent être utilisés avec les navigateurs classiques.

Les systèmes existants visent, soit une assistance générale du côté de l'utilisateur, soit une assistance limitée à un serveur spécifique. Radix, Letizia, WebWatcher, PersonalWebWatcher, WBI et WebMate font partie de la première catégorie et permettent une assistance multi-sites. Ces assistants reposent alors généralement sur une architecture de serveur proxy (PersonalWebWatcher, WBI et WebMate). WebWatcher est implanté sous la forme d'un serveur HTTP qui analyse et modifie tous les liens des pages visitées pour les rediriger vers le serveur lui-même. Ce procédé permet de suivre les utilisateurs durant leurs navigations tout en présentant un fonctionnement similaire à un proxy (mais plus coûteux). D'un autre côté, Hypercase, Hospitext, FindMe, Footprints et l'approche de Yan *et al*. sont liés à un hypermédia particulier. Hypercase nécessite en effet la définition préalable de chaque page, ce qui n'est pas adapté au Web qui est par nature dynamique et sans bornes. Hospitext utilise des connaissances du domaine et plus particulièrement une taxonomie des pages pour effectuer son apprentissage. Cette approche est difficile à mettre en pratique sur le Web en général puisqu'une telle hiérarchie n'existe pas[59]. FindMe est une approche servant à concevoir un serveur dans un domaine bien spécifique (recherche de vidéos par exemple). Footprints et l'approche de Yan *et al*. utilisent les logs d'un serveur, et sont donc fortement liés à celui-ci.

Enfin, certains assistants mettent en avant une architecture permettant de changer et d'évaluer différentes stratégies d'assistance ou de fonctionnement. WebWatcher et PersonalWebWatcher peuvent utiliser différentes techniques d'apprentissage. FindMe définit une approche applicable dans différents contextes. WBI offre également une architecture modulaire dans laquelle différents agents peuvent collaborer.

Broadway est indépendant du navigateur utilisé, et peut assister les utilisateurs sur n'importe quel site du Web grâce à une architecture de proxy. Il est enfin très ouvert : il repose sur une architecture répartie intégrant un ensemble extensible de modules et il utilise notre plate-forme CBR*Tools pour réaliser le serveur de recommandations.

4 Conclusion

Broadway est un système d'assistance à la navigation sur le Web, motivé par la difficulté de la recherche d'informations sur ce vaste hypermédia. Broadway est tout d'abord une application concrète qui nous a permis d'évaluer notre modèle d'indexation par situations comportementales ainsi que notre plate-forme à objets CBR*Tools. Nous montrons ainsi que notre modèle d'indexation permet la représentation des navigations ainsi que l'extraction des cas potentiels et leur transformation en cas concrets. Sur le plan de la réalisation, CBR*Tools nous a offert un cadre cohérent dans lequel nous avons spécialisé les objets gérant notre modèle d'indexation puis assemblé les phases du raisonnent pour former le système complet.

Grâce à Broadway, nous introduisons également un nouveau type d'assistants à la navigation. Nous proposons en effet la réutilisation de *comportements observés par variables* pour calculer les recommandations. Les assistants de navigation existants utilisent, soit le contexte local (dernière page visitée et parfois les objectifs de recherche), soit un profil utilisateur (résumé de la navigation en termes de mots clefs), soit des comportements simples (séquence ou ensemble de pages). Dans Broadway, quatre variables sont utilisées : adresse, contenu, évaluation explicite et évaluation implicite. Cette observation par variables, combinée avec notre modèle d'indexation, permet la gestion de comportements plus détaillés dans un cadre flexible et générique. L'évaluation

[59] Ce type d'approche pourrait être viable sur le Web si l'utilisation des métadonnées (informations liées aux pages) était normalisée et généralisée.

expérimentale de Broadway réalisée avec des utilisateurs réels montre la faisabilité et l'intérêt d'une telle approche.

Broadway est enfin un premier prototype démontrant notre approche. D'autres systèmes similaires mais basés sur des variables différentes, sont en cours d'étude avec notre participation : assistance d'un grand groupe d'utilisateurs sur un hypermédia délimité[60] et assistance dans une tâche de recherche combinant des requêtes et des navigations[61].

[60] Cf. contrat de recherche avec le CTI-CNET (centre de recherche de France Télécom) de 1998 à 2000.
[61] Cf. collaboration entre l'INRIA et le centre de recherche de Xerox en 1998.