

Résumé de la thèse

Sébastien Ferré

7 août 2003

Titre : Systèmes d'information logiques : un paradigme logico-contextuel pour interroger, naviguer et apprendre.

La plupart des systèmes d'organisation et de recherche d'information peuvent être classés en deux grandes catégories, dont les exemples paradigmatiques sont les systèmes de fichiers et les bases de données. Les systèmes de fichiers s'appuient sur une structure hiérarchique, qui sert aussi bien à l'organisation des données qu'à leur recherche par navigation. Cette structure est insatisfaisante car elle rend difficile certaines recherches, à cause de l'ordre qu'elle impose sur les critères de rangement ; et elle rend difficile le rangement d'un fichier dans plusieurs répertoires. De plus, l'organisation des données doit être faite manuellement, ce qui la rend fastidieuse. Dans les bases de données, cette organisation est transparente, et un langage d'interrogation expressif permet des recherches plus complexes. En revanche, ce langage peut être rebutant à utiliser, et aucun mécanisme de navigation n'assiste les utilisateurs dans leurs recherches. On retrouve cette opposition dans le domaine des moteurs de recherche sur Internet, par exemple entre Yahoo et Google. Finalement, le problème est que ces deux paradigmes de recherche d'information, que sont la navigation et l'interrogation, ne sont jamais complètement intégrés. Ils sont parfois associés (ex., commande `find` dans le système de fichiers UNIX), mais sans permettre d'alterner librement ces deux paradigmes dans une même recherche.

Pour réaliser cette combinaison, nous nous sommes fondés sur l'Analyse de Concepts (AC), qui permet de construire automatiquement une structure de navigation à partir d'une simple description des objets. Cette structure forme un *treillis de concepts*, aussi appelé treillis de Galois, où les concepts jouent à la fois le rôle de répertoire et de requête : l'*extension* du concept représente à la fois le contenu d'un répertoire et les réponses à une requête, tandis que son *intension* représente le chemin du répertoire et la requête. Ce couplage entre extension et intension, entre répertoire et requête, apporte de nombreux avantages : données centrées sur les objets, structure d'organisation et de navigation automatique et non-hiérarchique, alternance libre d'interrogation et de navigation dans une même recherche. Plus précisément, une requête définit une sorte de répertoire virtuel contenant les réponses à la requête, et les liens de navigation sont de nouveaux éléments de requête permettant de raffiner la requête courante et donc d'accéder à des sous-répertoires. Ces liens sont calculés automatiquement pour être pertinent au contexte de la recherche, et pour permettre l'accès à tous les objets par la seule navigation (complétude).

Dans l'AC, les descriptions d'objets sont limitées à des ensembles d'attributs. Comme il nous semble important de permettre à la fois des langages de requête expressifs et une approche générique vis-à-vis des nombreux domaines d'application possibles (ex., systèmes de fichiers, génie logiciel, systèmes d'information grand public), nous avons généralisé l'AC en remplaçant les ensembles d'attributs par les formules d'une logique presque arbitraire. Il suffit que cette logique forme un treillis pour la relation de déduction, pour conserver tous les résultats importants de l'AC. Cette Analyse de Concepts Logique (ACL) offrent donc les avantages déjà mentionnés, plus l'expressivité et la généralité d'une logique pour la description des objets, la formulation des requêtes et la représentation des liens de navigation. Ce sont ces propriétés de (1) données centrées sur les objets, (2) de combinaison d'interrogation et de navigation, (3) de représentation logique et (4) de généralité, qui caractérisent ce que nous appelons les Systèmes d'Information Logiques (SIL).

La définition de logiques demandant des compétences de logicien et de programmeur, nous avons développé une technique de modularisation des logiques permettant à des développeurs d'applications de définir des logiques *ad-hoc* par simple assemblage de composants prédéfinis, les *foncteurs logiques*. Chaque foncteur logique est défini par un langage, une sémantique et un jeu d'opérations logiques, telles que la déduction et la conjonction. Un type peut être affecté à chaque foncteur, définissant ses contraintes d'application, comme cela se fait pour les fonctions dans les langages de programmation typés. Nous avons implémenté une bonne dizaine de foncteurs, parmi lesquels certains représentent des domaines concrets (ex., entiers, chaînes de caractères, dates), et d'autres constituent des opérations sur les logiques (ex., somme, produit, clôture propositionnelle). Un foncteur logique particulier implémente une version générique de la logique «All I Know», qui permet d'appliquer l'hypothèse du monde clos à toute logique, et ce de façon monotone.

De par sa nature intensionnelle, la navigation est déjà une forme d'extraction de connaissances. En effet, les réponses à une requête ne sont pas immédiatement les objets satisfaisant cette requête, mais des compléments à cette requête, c-à-d. des formules logiques reflétant l'existence de certains types d'objets plutôt que d'autres. Nous avons généralisé le principe de cette navigation à la recherche de certaines formes de règles d'association. La navigation peut aussi être vue comme une forme suggestive de l'interrogation : le SIL assiste l'utilisateur en suggérant des modifications pertinentes de la requête. Parallèlement, nous avons esquissé une forme suggestive à la définition et à la mise-à-jour des descriptions d'objets. Dans l'exemple de la messagerie électronique, le SIL peut ainsi suggérer des propriétés à affecter aux nouveaux messages en fonction des anciens messages, et donc assister l'utilisateur dans leur filtrage et leur classification (ex., spams).

Nous avons développé un prototype pour implémenter et expérimenter toutes les idées développées au cours de cette thèse. Il est générique dans la mesure où il doit être instancié à une logique particulière pour former une application. Cette logique peut être assemblée à partir des foncteurs logiques. Cette généralité nous a permis de spécialiser facilement notre prototype de SIL à diverses applications, telles que la gestion d'une bibliographie, d'une messagerie électronique, ou d'un ensemble de recettes décrites par leurs ingrédients.