Un modèle pour l'interrogation visuelle des documents structurés.

Rami HARRATHI, Sylvie CALABRETTO
LIRIS UMR 5205, INSA de Lyon, Campus de la Doua, Bâtiment Blaise Pascal (501), 7, avenue Jean Capelle 69621 VILLEURBANNE CEDEX
prenom.nom@liris.cnrs.fr

RÉSUMÉ. *Dans cet article, nous présentons un modèle d'interrogation visuelle des documents structurés permettant de représenter graphiquement les requêtes utilisateurs sous forme de graphe, et d'effectuer ainsi des requêtes d'une rare complexité syntaxique et sémantique. Le modèle est validé par le développement d'un prototype XmlBrowser permettant d'explorer et d'interroger une collection de documents structurés (Xml).*

ABSTRACT. In this article, we present a visual querying model of the structured documents making it possible to represent graphically the users query in the form of graph, thus to carry out query of a rare syntactic and semantic complexity. The model is validated by the development of an XmlBrowser prototype for seamlessly browsing and querying a collection of structured document (Xml).

MOTS-CLÉS : XML, documents structurés, Recherche d'information, indexation, Langage de requête, visualisation.

KEYWORDS: *XML, structured document, information retrieval, indexing, query language, visualization.*

CORIA 2007, pp. 211–222
Saint-Étienne, 28–30 mars 2007

1. Introduction

Les documents structurés présentent l'avantage de posséder une structure qui facilite leur présentation, ainsi que leur interprétation et leur exploitation dans différents contextes. Ainsi il est devenu primordial de concevoir des méthodes d'indexation et de recherche permettant d'exploiter à la fois la structure et le contenu textuel de ces documents. Ces méthodes doivent permettre à l'utilisateur d'interroger des documents structurés en spécifiant des conditions sur le contenu textuel mais aussi sur la structure selon ses besoins.

Dans ce cadre et surtout avec l'avènement de XML (Bray et al., 2006), comme un format reconnu de représentation des documents structurés, de nombreux langages ont été développés pour interroger les documents structurés. Parmi ces langages, on peut citer UnQL (Buneman et al. 1996), Lorel (Abiteboul et al., 1997), XML-QL (Levy et al. 98), XQL (Robie et al., 1998), QUILT (Chamberlin et al., 2000), Elixir (Chinenyanga et al., 2001), XIRQL (Fuhr et al., 2001) et XQuery (Boag et al., 2006). La limite des langages proposés pour l'interrogation des documents structurés réside dans le fait qu'il s'agit de langages textuels inadaptés aux utilisateurs non informaticiens. Ces langages nécessitent de la part de l'utilisateur un apprentissage de la syntaxe formelle du langage.

Afin de faciliter la tâche des utilisateurs occasionnels, des travaux sont menés principalement sur deux plans, à savoir les langages naturels (Tannier, 2005) et les langages visuels (Aufaure et al., 1998). Les premiers représentent la solution la plus conviviale mais, face aux ambiguïtés sémantiques inhérentes aux langages parlés, ils sont encore loin d'être au point et efficacement exploitables. Les langages visuels quant à eux, caractérisés par l'incorporation d'éléments non textuels (formulaires, icônes, images, graphes, ...), représentent actuellement la meilleure alternative.

Dans ce cadre il y a un peu de travaux de recherche dans le domaine des langages d'interrogation visuelle et l'exploration graphique des documents structurés (Xml). Parmi ces langages, on peut citer XML-GL (Ceri et al., 1999) ; il s'agit d'un langage d'interrogation entièrement graphique pour la récupération et la manipulation de données XML. Mais il ne supporte pas l'exploration interactive et graphique des données XML.

Le système LORE inclut un *guide de données (*DataGuides) (Roy et al., 1998) ; il s'agit d'une interface graphique pour l'exploration des données XML. DataGuides présente une description compacte des données XML ainsi le squelette de base de la requête utilisateur est hérité de la structure de la base de données. Cette dépendance oblige l'utilisateur à connaître la structure de la base de données et limite l'expressivité des requêtes utilisateurs. Dans (Kevin et al., 2000) les auteurs présentent le système BBQ (*B*lended *B*rowsing and *Q*uerying) de formulation interactive des requêtes. Comme DataGuides, dans BBG l'utilisateur est sensé connaître la structure de la base de données.

Récemment Holger (Holger et al., 2005) a proposé un environnement graphique pour visualiser et interroger une collection de documents en se basant sur une représentation graphique de la requête sous forme d'un arbre. La limite principale de l'approche proposée est que toutes les relations structurelles ne sont pas supportées, seules les relations child et descendant sont supportées.

Dans cet article nous proposons un modèle d'interrogation permettant d'interroger visuellement les documents structurés. L'approche proposée se base sur un modèle de graphe d'interrogation permettant d'effectuer des requêtes d'une rare complexité syntaxique et sémantique. Nous présentons cet article de la manière suivante : dans la section 2 nous décrivons le modèle d'indexation utilisé, dans la section 3 nous présentons la syntaxe et la sémantique du graphe d'interrogation proposé pour la recherche d'information structurée. Enfin, dans la section 4, nous décrivons notre prototype.

2. Indexation de la structure des documents

Dans notre approche, nous adoptons le modèle DOM (Apparao et al., 1998) où la structure logique d'un document est modélisée par un arbre de noeuds. Les nœuds de cet arbre sont typés (éléments, attributs, texte). Nous proposons de raffiner l'approche de Dietz's (Dietz., 1982), où chaque noeud est associé à son numéro dans l'arbre en pré-ordre et en post-ordre. Ainsi nous utilisons deux identificateurs pour un noeud d'arbre XML début et fin. Les valeurs de début et fin sont assignées aux noeuds comme suit :

- *début* : l'ordre d'apparition d'un noeud dans la lecture séquentielle du document XML
- *fin* : l'ordre de disparition d'un noeud dans la lecture séquentielle du document XML

Les valeurs de début et fin assignées aux noeuds d'un document XML sont calculées en utilisant les méthodes fournies par le parseur SAX. Chaque noeud de l'arbre XML est codé par un intervalle [début, fin]. La relation entre deux noeuds u et v est résolue aisément comme suit :

- Ancêtre -Descendant, u est un ancêtre de v si seulement si l'intervalle de u contient l'intervalle de v.
- Précèdent -suivant, u précède v si seulement si l'intervalle de u précède l'intervalle de v.

La figure ci-dessous donne un exemple d'assignement pour un document XML.

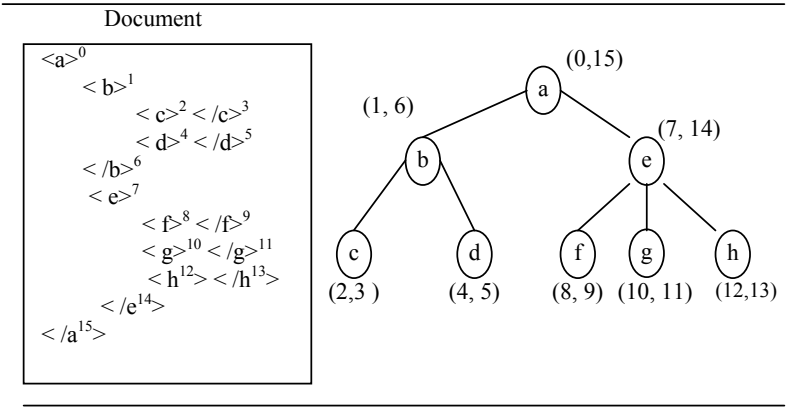

Figure 1. *L'assignement des valeurs de début et fin aux noeuds d'un document XML.*

Ainsi les relations structurelles entre les noeuds (ancêtre –descendant, suivant - précèdent) sont déterminées aisément en examinant les valeurs de début et fin. Soit T l'ensemble des noeuds d'un arbre XML et Q ($Q_{début}$, Q_{fin}) un nœud de cet arbre, les relations structurelles sont résolues comme suit :

- Ancêtre (Q) = $\{N \in T \ / \ N_{début} < Q_{début} \text{ et } Q_{fin} > N_{fin}\}$
- Descendant (Q) = $\{N \in T \ / \ N_{début} > Q_{début} \text{ et } Q_{fin} > N_{fin}\}$
- Précédent (Q) = $\{N \in T \ / \ Q_{début} > N_{fin}\}$
- Suivant (Q) = $\{N \in T \ / \ Q_{fin} < N_{début}\}$

3. Graphe d'interrogation des documents structurés

Le modèle de représentation de la requête que nous présentons a pour but principal de fournir un modèle permettant l'interrogation des documents structurés. Ce modèle d'interrogation se base sur une représentation graphique permettant la manipulation directe, la souplesse et l'assistance dans la formulation des requêtes.

Une requête est un graphe orienté G représenté par un couple (N, R), où : N est l'ensemble de sommets représentant des noeuds et R est l'ensemble des arcs représentant les relations de structure entre les nœuds. Dans une requête on distingue deux types de contraintes :

- les contraintes de description : les contraintes sur le nœud lui-même (type, nom…).

- les contraintes de relation : les contraintes sur les relations de structure qui peuvent exister entre les nœuds.

Dans la suite nous présentons la syntaxe et la sémantique du graphe d'interrogation.

Définition 1 (Nœud et Document structuré). Nous proposons la définition suivante d'un noeud en nous inspirant de l'approche de Grust (Grust et al., 2002). Un noeud de la structure (l'arbre XML) est décrit par un n-uplet < *début, fin, parent, type, nom, valeur* > où :

- *début*, *fin, parent* sont les identificateurs du nœud
- *type* : le type du nœud (élément, attribut, texte)
- *nom* : nom du noeud (nom de la balise ou de l'attribut, dans le cas d'un noeud de type texte le nom vaut nul)
- *valeur* : la valeur du noeud (la valeur de l'attribut ou le contenu textuel, dans le cas d'un noeud de type élément la valeur vaut nul)

Ainsi un document structuré est défini comme un arbre des nœuds.

Définition 2 (Noeud Cible). Un noeud cible est un nœud d'arbre représentant le résultat de la requête. C'est le nœud qui va être retourné et affiché à l'utilisateur.

Définition 3 (Contraintes élémentaires de description). Une contrainte élémentaire de description est une contrainte sur le descripteur du nœud. On distingue trois contraintes élémentaires de description :

- une contrainte portant sur le type du noeud C_{Type} : elle exprime le type du nœud (élément, attribut, texte).
- une contrainte portant sur le nom du noeud C_{Nom} : elle exprime le nom du nœud (nom de la balise ou de l'attribut).
- une contrainte portant sur la valeur du noeud C_{Valeur} : elle exprime la valeur du nœud (la valeur d'un nœud de type texte (le contenu textuel) ou la valeur d'un noeud de type attribut).

Etant donné un nœud d'arbre n, ces contraintes élémentaires sont formalisées comme suit :

- C_{Type} = *(Type (n)= t) (le nœud n est de type de t,* $t \in$ *{élément, attribut, texte})*
- C_{Nom} = *(Nom (n)= m) (le nœud n a le nom m)*
- C_{Valeur} = *(Valeur (n)=v) (le nœud n a la valeur v)*

Définition 4 (Contraintes de description). Une contrainte de description C_d est une conjonction de contraintes élémentaires de description.

Par exemple si l'utilisateur désire obtenir un élément Article, il peut exprimer son besoin en donnant le nom (nom de la balise) et le type du nœud (élément), la requête est traduite en une contrainte de description :

R1 : C_d (n)= (Type (n) = élément) ^ (Nom (n)=Article)

Définition 5 (Contraintes de relation). Une contrainte de relation C_R exprime la relation qui existe entre deux noeuds. Etant donnés deux nœuds d'arbre n_1 et n_2, cette contrainte est de la forme suivante :

- *R* (n_1) = n_2 (le noeud n_2 appartient à l'ensemble des noeuds en résultat de l'application de la relation *R* sur le nœud n_1).
- *R* ∈ {child, descendant, parent, ancestor, following, preceding, following-sibling, preceding-sibling}.

Par exemple si l'utilisateur désire obtenir un élément Article suivi d'un élément Section, sa requête se traduit en une contrainte de relation entre deux noeuds n_1 et n_2, qui ont respectivement les contraintes de description $C_d(n_1)$ et $C_d(n_2)$ suivantes :

R2 : C_R = (Suivant (n_1) = n_2)
C_d (n_1) = (Type (n_1) = élément) ^ (Nom (n_1)=Article)
C_d (n_2) = (Type (n_2) = élément) ^ (Nom (n_2)=Section)

R3 : C_R = (Fils (n_1) = n_2)
C_d (n_1)= (Type (n_1) = élément) ^ (Nom (n_1)=Titre)
C_d (n_2)= (Type (n_2) = texte) ^ (Valeur (n_2)= Recherche d'information)

R4 : C_R = (Fils (n_1) = n_2)
C_d (n_1)= (Type (n_1) = élément) ^ (Nom (n_1)=Article)
C_d (n_2)= (Type (n_2) = attribut) ^ (Nom (n_2)= Auteur) ^ (Valeur (n_2)= Rami).

La requête R3 signifie que l'utilisateur désire obtenir un élément Titre sur 'Recherche d'information'. La requête R4 signifie que l'utilisateur désire obtenir tous les éléments Article ayant un attribut Auteur dont la valeur est 'Rami'.

Définition 6 (Graphe d'interrogation). Un graphe d'interrogation G est une conjonction de contraintes de relation.

Par exemple si l'utilisateur désire obtenir un élément Article contenant deux éléments Section et Titre sur ' document structuré ' tel que l'élément Section est suivi par l'élément Titre, la requête se traduit en un graphe d'interrogation (Figure 2).

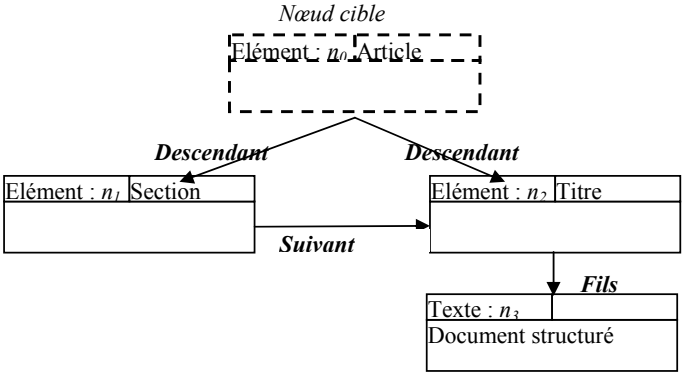

Figure 2. *Exemple de graphe d'interrogation.*

Ce graphe d'interrogation est formalisé comme suit :

– $G = (Descendant\ (n_0) = n_1)\ \wedge\ (Descendant\ (n_0) = n_2)\ \wedge\ (Suivant\ (n_1) = n_2)\ \wedge\ (Fils\ (n_2) = n_3)$

Afin de valider notre approche un prototype a été développé. Dans la section suivante nous présentons les différentes caractéristiques de ce prototype appelé XmlBrowser.

4. Présentation du prototype XmlBrowser

4.1. *Architecture générale du prototype*

Le prototype XmlBrowser constitue un outil permettant d'indexer, d'interroger et d'explorer facilement les différentes structures logiques d'un corpus de documents structurés au format XML. L'outil proposé offre une interface graphique permettant de saisir des requêtes utilisateurs.

Le prototype est réalisé entièrement en langage java (1.3) en utilisant des API telles que l'API SAX de Xerces pour parser les documents Xml et l'API JGraph pour implémenter l'éditeur des graphes de requêtes. L'architecture du prototype XmlBrowser est la suivante :

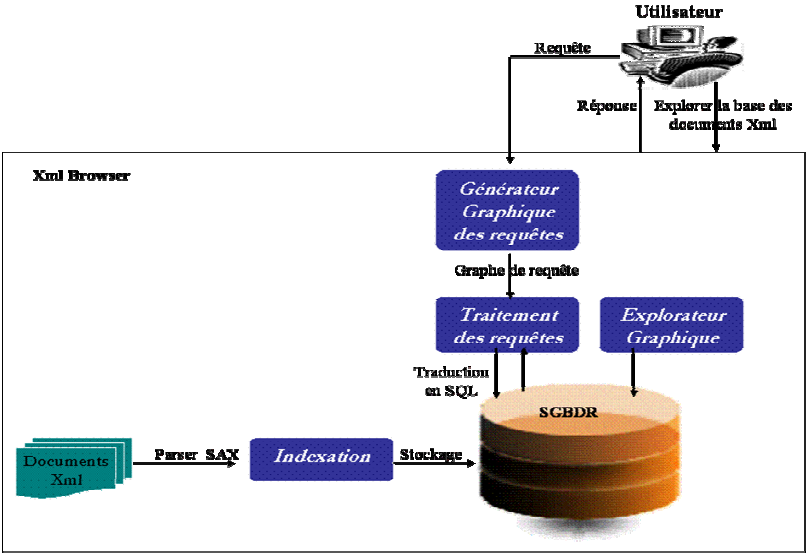

Figure 3. *Architecture générale du prototype Xml Browser.*

Pour le stockage des index nous avons utilisé une base de données relationnelle en profitant de la maturité des bases de données au niveau stockage et langage d'interrogation. Notre base contient trois tables : la première table pour stocker la structure, la deuxième pour le contenu des attributs et la troisième pour le contenu textuel dans documents XML. L'architecture comprend aussi :

- Un module d'indexation, qui parse un corpus de documents XML fourni par l'utilisateur.
- Un module de génération des requêtes, interface graphique, permet à l'utilisateur de saisir sa requête sous forme d'un graphe.
- Un module de traitement des requêtes, permet de traduire les graphes d'interrogation en requête SQL. Ce module renvoie à l'utilisateur une liste d'éléments répondant à sa requête.
- Un module d'exploration graphique, présente les différentes structures des documents XML dans une hiérarchie.

4.2. *Présentation de l'interface de XmlBrowser*

Parmi les fonctionnalités de notre prototype, nous trouvons l'indexation d'un corpus de documents XML fourni par l'utilisateur. Les documents XML indexés

sont visualisés dans une hiérarchie, permettant ainsi à l'utilisateur d'explorer aisément la base de documents (Figure 4).

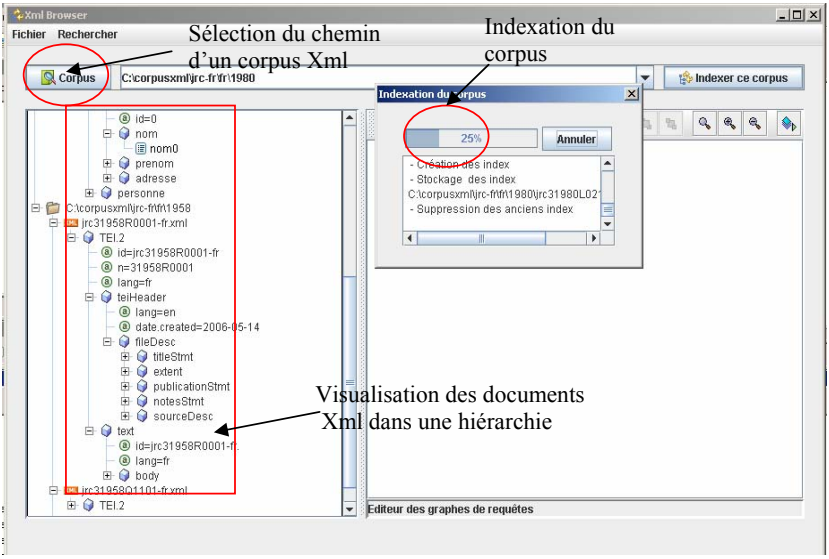

Figure 4. *Indexation et Exploration d'un corpus de documents Xml.*

XmlBrowser offre aussi une interface graphique permettant à l'utilisateur de saisir sa requête sous forme d'un graphe (Figure 5). Les types des nœuds de ce graphe sont représentés par des icônes et l'utilisateur peut éditer les descripteurs de chaque nœud (type, nom, valeur) en spécifiant leurs valeurs. L'éditeur des noeuds offre un dictionnaire des noms (nom des attributs, les balises) afin de contrôler la saisie des descripteurs. L'utilisateur peut aussi spécifier les relations qui peuvent exister entre les nœuds de façon interactive en éditant les relations de structure disponibles.

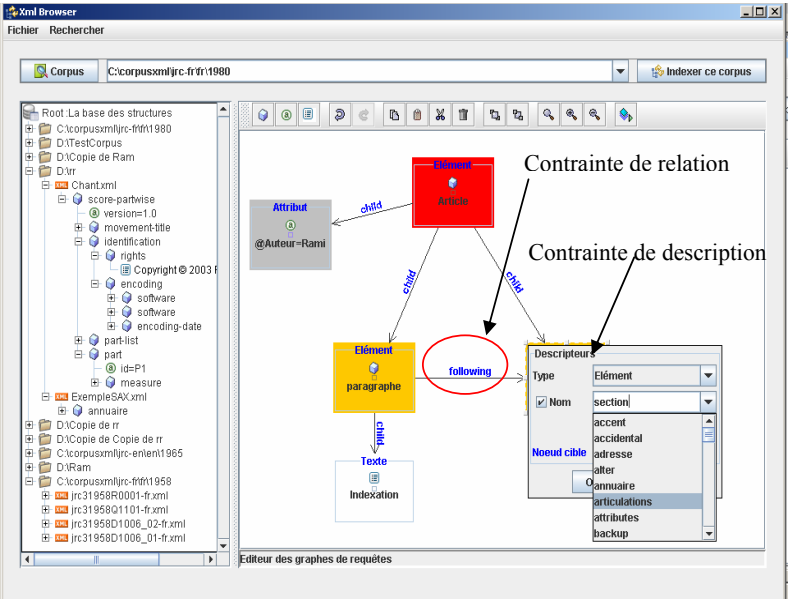

Figure 5. *Création d'un graphe d'interrogation.*

5. Conclusion et perspectives

Dans cet article nous avons proposé un modèle flexible d'interrogation permettant une interrogation visuelle des documents structurés plus expressive que les langages textuels. Le modèle d'interrogation se base sur une représentation graphique de la requête sous forme d'un graphe permettant ainsi de créer des requêtes d'une rare complexité syntaxique et sémantique.

En termes de perspectives à notre travail nous comptons intégrer une approche orientée recherche d'information afin d'ordonnancer les documents selon leur pertinence par rapport à la requête/au besoin de l'utilisateur.

6. Bibliographie

Abiteboul, S., Quass, D., Mc Hugh, J., Widom, J., Wiener, J-L. , "The Lorel query language for semi-structured data". *International Journal on Digital Libraries*, 1(1), 68-88, 1997.

Apparao,V., Byrne, S., Champion,M., Isaacs, S., Jacobs, I., Le Hors,A., Nicol,G., Robie,J., Sutor,R., Wilson,C., Wood,L., "Document Object Model (DOM) ". W3C recommendation, Technical Report REC-DOM-Level-1-19981001, Octobre 1998

Aufaure, M., Bonhomme, C., Lbath, A. "LVIS: un langage visuel d'interrogation de bases de données spatiales". BDA 1998

Boag, S., Fernández, MF., Florescu, D., Robie, J., Siméon,J., Watson Research, T.J., "XQuery 1.0: An XML Query Language". W3C recommendation, Technical Report PR-xquery-20061121 , WWW Consortium, Novembre 2006

Bray, T., Paoli J., Sperberg-McQueen, C. M., Maler E., Yergeau F., "Extensible Markup Language (XML) 1.0 ";Fourth Edition W3C recommendation, Technical Report REC-xml-20060816 , WWW Consortium, September 2006

Buneman, P., Davidson, S., Hillebrad, G., Suciu, D., "A query language and optimisation techniques for unstructured data". *ACM-SIGMOD record*, pp. 505-516, Montréal, 1996.

Ceri, S., Comai, S., Damiani, E., Fraternalli, P., Paraboschi, S. et Tanca, L. : "XML-GL :A graphical language for querying and restructuring WWW Data". *In Proc. Of the 8th Int. WWW Conference*, WWW8, Toronto, Canada, May 1999.

Chamberlin, D., Robie, J., Florescu, D., "Quilt: An XML query language for heterogeneous data sources*". In Proc. 3rd Int. Workshop on WWW and databases*. p.1-25. Dallas, 2000.

Chinenyanga, T. T., Kushmerick, N. , "Expressive Retrieval from XML Documents". *In Proc. Of ACM SIGIR 2001*, pp. 163-171, New-Orlean, USA, 2001.

Dietz, PaulF. "Maintaining order in a linked list". *In Proceedings of the Fourteenth Annual ACM Symposium on Theory of Computing*, pages 122–127, San Francisco, California, May 1982.

Fuhr, N., Grossjohann, K. "XIRQL: A query Language for Information Retrieval in XML Documents". *In Proc. of the 24th annual ACM SIGIR*, New Orléans, p.172-180, 2001.

Grust, T, "Accelerating XPath Location Steps". In M. J. Franklin, B. Moon, and A.Ailamaki, editors*, Proceedings of the 2002 ACM SIGMOD International Conference on Management of Data,* Madison, Wisconsin, USA, pages 109-120. ACM, 2002.

Holger, M., Klaus, S., Felix, W., Simone, L., François B., "Visual exploration and retrieval of XML document collections with the generic system X^{2}". *Int. J. on Digital Libraries* 5(1): 3-17 (2005)

Kevin, DMunroe. Yannis, P., "BBQ: A Visual Interface for Integrated Browsing and Querying of XML*". VDB 2000:* 277-296

Levy, A., Fernandez, M., Suciu, D., Florescu, D., Deutsch, "A. XML-QL: A query language for XML". *W3C technical report, Number NOTE- xml-ql-19980819*, 1998.

Robie, J., Lapp, J., Schach, D., "XML Query Language (XQL*)". Proc. of W3C QL'98 (Query Languages 98) ,* Massachusetts, 1998.

Roy, G., Jennifer, W., "Interactive query and search in semistructured databases". In webDB'98, *Proc. Int. Workshop on the Web and Databases*, 1998.

Tannier, A., “From natural language to NEXI, an interface for INEX 2005 queries”. *Advances in XML Information Retrieval: Fourth Workshop of the INitiative for the Evaluation of XML Retrieval (INEX 2005)*, Schloss Dagstuhl, Germany, November 28-30, 2005. © Springer-Verlag, Lecture Notes in Computer Science (LNCS 3977), pages 373-387, 2006.