Action Concertée Incitative

MASSES DE DONNÉES

Descriptif complet du projet

I - FICHE D'IDENTITÉ DU PROJET

Nom du Projet : *(maximum 20 caractères)*

⟨ TRALALA /⟩

Titre du Projet : *(maximum 3 lignes)*

Langages pour la manipulation de documents XML : fondement et pratique
(**TRA**nsformation **LA**nguages for XML: **L**ogics and **A**pplications)

Type du Projet[1]**:**

Projet de recherche	Projet de recherche multi-thématiques	Projet de recherche avec infrastructure	Projet associé à l'imagerie spatiale	Autre
✓				

Durée du projet[2]**: 36 mois**

Description courte du Projet : *(une demi-page maximum)*

Notre projet se propose d'étudier les aspects de traitement, d'interrogation et de manipulation de grandes masses de données lorsque celles-ci sont disponibles au format XML. Nous nous intéressons plus précisément aux aspects langages de programmation et langages de requêtes. Notre ambition est de couvrir de manière intégrée un large spectre de problématiques: de celles liées aux **aspects langages** (expressivité, typage, étude de nouvelles primitives de programmation, logiques sous-jacentes pour l'interrogation, optimisation logique), jusqu'aux aspects traitant l'**accessibilité des données** (données en streaming, compression, définition de modèles d'accès en mémoire secondaire, utilisation de moteurs de persistance), en passant par les problématiques liées à **l'implantation** (compilation du filtrage, optimisation physique, vérification du sous-typage, modèles d'exécution pour le streaming).
Nous attaquerons ces problématiques en organisant notre recherche selon trois axes directeurs: langages de requêtes, traitement de données à la volée et typage de documents.

langages de requêtes: étude théorique de paradigmes d'interrogation sous l'angle de l'expressivité et de la complexité; définitions de langages de requêtes pour XML ayant les caractéristiques d'expressivité restreinte et de déclarativité typiques des langages du modèle relationnel ; implantation puis conception et validation de techniques d'optimisations adaptées à ces différents paradigmes.

traitement des données à la volée (streaming): identification d'une classe de requêtes qu'il est possible d'évaluer par « streaming », avec ou sans compression des données, et dans le premier cas de détermination de la granularité optimale de compression.

contraintes et typage de documents: vérification (efficace) de contraintes d'intégrités plus fines que celles des langages traditionnels (par exemple, l'*interleaving* des éléments XML; utilisation des types statiques pour la compilation efficace et l'optimisation de requêtes.

Outre une unité méthodologique, et une unité dans les objectifs, la coopération au sein du projet est accrue par la décision de choisir une cible logicielle unique à nos efforts d'implantation, le langage CDuce, qui est développé conjointement par le LIENS et le LRI, deux des sites participants à ce projet.

[1] Cocher la case correspondante au type du projet soumis.

Coordinateur du projet :

Nom	Prénom	Laboratoire (sigle éventuel et nom complet)
CASTAGNA	Giuseppe	LIENS, Laboratoire d'Informatique de l'École Normale Supérieure (UMR 8548)

Organisme de rattachement financier pour le présent projet :

École Normale Supérieure, Paris.

Équipes ou laboratoires partenaires (nom complet et éventuellement sigle)[3]

Projet GEMO, INRIA Futurs *en partenariat avec* le LIAFA, Laboratoire d'Informatique Algorithmique: Fondements et Applications (UMR 7089), équipe Vérification.
LIENS, Laboratoire d'Informatique de l'École Normale Supérieure (UMR 8548), équipe Langages.
LIF, Laboratoire d'Informatique Fondamentale de Marseille (UMR 6166), équipe Move.
Équipe MOSTRARE, INRIA Futurs, *en partenariat avec* le LIFL Laboratoire d'Informatique Fondamentale de Lille (UMR 8022), équipe Spécification, Tests et Contraintes.
LRI, Laboratoire de Recherche en Informatique, (UMR 8623), équipe Bases de Données.

[2]La durée d'un projet ne peut excéder 36 mois. Des demandes de projets d'une durée plus courte devront être particulièrement argumentées.
[3]Insérer autant de lignes que nécessaire.

Action Concertée Incitative

MASSES DE DONNÉES

Descriptif complet du projet

Informations de cadrage du projet :

Durée : 36 mois

Moyens demandés dans le cadre de l'ACI via le Fonds National de la Science (montants en Euros TTC) :

Équipement	42 500
Fonctionnement (sans CDD)	210 000
Dépenses de personnels (CDD)	217 900
Total	470 400

II - PRÉSENTATION DÉTAILLÉE DU PROJET

A - IDENTIFICATION DU COORDINATEUR ET DES AUTRES PARTENAIRES DU PROJET :

A1 - Coordinateur du Projet :

M. ou Mme. Prénom Nom[4]	M. Giuseppe Castagna
Fonction[4]	Chargé de Recherche CNRS
Laboratoire (Nom complet et sigle le cas échéant)[4]	Laboratoire d'Informatique de l'École Normale Supérieure (LIENS)
Adresse[4]	45, rue d'Ulm - 75005 Paris
Téléphone[4]	01 4432 2082
Fax	01 4432 2156
Mél[4]	Giuseppe.Castagna@ens.fr

[4] Champ obligatoire.

Action Concertée Incitative

MASSES DE DONNÉES

Descriptif complet du projet

A2- Équipes ou laboratoires partenaires du Projet:

Identification de l'équipe ou du laboratoire

Équipe ou Laboratoire	Projet GEMO, INRIA Futurs *en partenariat avec* le LIAFA, Laboratoire d'Informatique Algorithmique: Fondements et Applications (UMR 7089), équipe Vérification.
Adresse	INRIA, Parc-club Orsay université, ZAC des vignes, 4 rue J. Monod, 91893 Orsay Cedex

Organisme de rattachement financier de l'équipe pour le présent projet

INRIA

Responsable du projet au sein de l'équipe ou du laboratoire

M. ou Mme. Prénom Nom	M. Luc SEGOUFIN
Fonction	CR INRIA
Téléphone	01 7292 5930
Fax	01 7274 7335
Mél	luc.segoufin@inria.fr

Membres de l'équipe participant au projet (y compris le responsable)

Nom	Prénom	Poste statutaire	% du temps de recherche consacré au projet
Arion	Andrei	Doctorant	100
Manolescu	Ioana	CR INRIA	50
Muscholl	Anca	Professeur Paris 7 - LIAFA	50
Samuelides	Mathias	Doctorant	100
Segoufin	Luc	CR INRIA	50
Schwentick	Thomas	PR U. Marburg (All.)	partenaire étranger

Références :

1. Numerical Document Queries. H. Seidl, T. Schwentick, et A. Muscholl, Principles of Database Systems (PODS) 2003
2. Typing and querying XML documents: some complexity bounds. L. Segoufin, Principle of Databases Systems (PODS) 2003
3. Validating Streaming XML Documents. L. Segoufin et V. Vianu, Principle of Databases Systems (PODS) 2002
4. Andrei Arion, Angela Bonifati, Gianni Costa, Sandra D'Aguanno, Ioana Manolescu, Andrea Pugliese: XQueC: Pushing Queries to Compressed XML Data. VLDB 2003: 1065-1068
5. Albrecht Schmidt, Florian Waas, Martin L. Kersten, Michael J. Carey, Ioana Manolescu, Ralph Busse: XMark: A Benchmark for XML Data Management. VLDB 2002: 974-985

Action Concertée Incitative

MASSES DE DONNÉES

Descriptif complet du projet

Identification de l'équipe ou du laboratoire

Équipe ou Laboratoire	LIENS, Laboratoire d'Informatique de l'École Normale Supérieure (UMR 8548), équipe Langages.
Adresse	45, rue d'Ulm - 75005 Paris

Organisme de rattachement financier de l'équipe pour le présent projet

École Normale Supérieure de Paris

Responsable du projet au sein de l'équipe ou du laboratoire

M. ou Mme. Prénom Nom	M. Giuseppe Castagna
Fonction	CR CNRS
Téléphone	01 4432 2082
Fax	01 4432 2156
Mél	Giuseppe.Castagna@ens.fr

Membres de l'équipe participant au projet (y compris le responsable)

Nom	Prénom	Poste statutaire	% du temps de recherche consacré au projet
Castagna	Giuseppe	CR CNRS	30%
Frisch	Alain	Ingénieur des Télécoms	50%
Hosoya	Haruo	MdC (U. Tokyo)	*(partenaire étranger)*
Pierce	Benjamin	PR (U. of Pennsylvanie)	*(partenaire étranger)*

Références :

1. **Logiciel :** The CDuce Programming Language (`http://www.cduce.org`).
2. V. Benzaken, G. Castagna, and A. Frisch. CDuce: an XML-centric general-purpose language.In *ICFP '03, 8th ACM International Conference on Functional Programming*, pages 51–63, Uppsala, Sweden, 2003. ACM Press.
3. Alain Frisch, Giuseppe Castagna, and Véronique Benzaken. Semantic subtyping. In *IEEE Symposium on Logic In Computer Science (LICS)*, 2002.
4. A. Frisch and L. Cardelli. Greedy regular expression matching. Programming Language Technologies for XML (PLAN-X 2004, Venise).

Action Concertée Incitative

MASSES DE DONNÉES

Descriptif complet du projet

Identification de l'équipe ou du laboratoire

Équipe ou Laboratoire	LIF, Laboratoire d'Informatique Fondamentale de Marseille (UMR 6166), équipe Move.
Adresse	CMI, 39 rue Joliot-Curie, 13454 Marseille Cedex 13

Organisme de rattachement financier de l'équipe pour le présent projet

Université de Provence

Responsable du projet au sein de l'équipe ou du laboratoire

M. ou Mme. Prénom Nom	M. Silvano dal Zilio
Fonction	CR CNRS
Téléphone	04 9111 3625
Fax	04 9111 3601
Mél	dalzilio@cmi.univ-mrs.fr

Membres de l'équipe participant au projet (y compris le responsable)

Nom	Prénom	Poste statutaire	% du temps de recherche consacré au projet
Dal Zilio	Silvano	CR CNRS	50
Lugiez	Denis	Professeur	30
Meysonnier	Charles	Doctorant	100

Références :

1. Silvano Dal Zilio, Denis Lugiez et Charles Meyssonnier. A Logic you Can Count On. In *31st Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL)*, ACM Press, 2004.
2. Silvano Dal Zilio et Denis Lugiez. XML schema, tree logic and sheaves automata. In *Rewriting Techniques and Applications (RTA)*, 2003.
3. Denis Lugiez. Counting and Equality Constraints for Multitree Automata. In *Foundations of Software Science and Computation Structures (FoSSaCS)*, 2003.
4. H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison et M. Tommasi. Tree automata techniques and applications. (livre à paraître). `http://www.grappa.univ-lille3.fr/tata`

Action Concertée Incitative

MASSES DE DONNÉES

Descriptif complet du projet

Identification de l'équipe ou du laboratoire

Équipe ou Laboratoire	MOSTRARE INRIA Futurs & LIFL, Laboratoire d'Informatique Fondamentale de Lille (UMR 8022), équipe Spécification, Tests et Contraintes.
Adresse	Université des Sciences et Technologies de Lille - Bâtiment M3, 59655 Villeneuve d'Ascq Cedex

Organisme de rattachement financier de l'équipe pour le présent projet

INRIA

Responsable du projet au sein de l'équipe ou du laboratoire

M. ou Mme. Prénom Nom	Mme. Anne-Cécile Caron
Fonction	Maître de conférences
Téléphone	03 2877 8573
Fax	03 2877 8537
Mél	caronc@lifl.fr

Membres de l'équipe participant au projet (y compris le responsable)

Nom	Prénom	Poste statutaire	% du temps de recherche consacré au projet
Niehren	Joachim	Ingénieur expert INRIA	50
Tison	Sophie	Professeure	50
Caron	Anne-Cécile	Maître de Conférences	50
Talbot	Jean-Marc	Maître de Conférences	50
Boneva	Iovka	Doctorante	75
Debarbieux	Denis	Doctorant	75

Références :

1. When Ambients Cannot be Opened. Iovka Boneva, et Jean-Marc Talbot. *Theoretical Computer Science*, 2004, à paraître.
2. Querying Unranked Trees with Stepwise Tree Automata, Julien Carme, Joachim Niehren, Marc Tommasi. *International Conference on Rewriting Techniques and Applications*, 2004, à paraître.
3. A New Algorithm for Normal Dominance Constraints, Manuel Bodirsky, Denys Duchier, Sebastian Miele, et Joachim Niehren, *ACM-SIAM Symposium on Discrete Algorithms* : 54-78, the ACM Press, 2004.
4. Path rewriting in semistructured data. D. Debarbieux, Y. Roos, S. Tison, Y. Andre, and A.C. Caron. *International Conference on Words* : 358-369, 2003.
5. Modèles de données semi-structurées et contraintes d'inclusion. A.C. Caron, D. Debarbieux, et Y. Roos, RSTI série RIA-ECA, volume 17. Extraction et Gestion des Connaissances, Hermès : 461-472, 2003.
6. Recognizable tree languages and non-linear morphisms. M.Dauchet, S.Tison, et M.Tommasi, *Theoretical Computer Science*, 281:219-234, 2002.
7. Grid structures and undecidable constraint theories. F. Seynhaeve, S. Tison, M. Tommasi, et R. Treinen. *Theoretical Computer Science*, 258:453-490, 2001.
8. The Constraint Language for Lambda Structures. Markus Egg, Alexander Koller, et Joachim Niehren. *Journal of Logic, Language, and Information*, 10:457-485, 2001.
9. Atomic Set Constraints with Projection. W. Charatonik, et J.-M. Talbot, *International Conference on Rewriting Techniques and Applications*, 311–325, 2002.
10. Generalized definite set constraints. P. Devienne J.-M. Talbot, et S. Tison. *Constraints, an International Journal*, 5(1-2):161–202, 2000.
11. On Rewrite Constraints and Context Unification, Joachim Niehren, Ralf Treinen, et Sophie Tison. *Information Processing Letters*. 74(1-2): 35-40, 2000.
12. Tree automata techniques and applications. H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison, et M. Tommasi. Livre en ligne: http://www.grappa.univ-lille3.fr/tata.

Action Concertée Incitative

MASSES DE DONNÉES

Descriptif complet du projet

Identification de l'équipe ou du laboratoire

Équipe ou Laboratoire	LRI, Laboratoire de Recherche en Informatique, (UMR 8623), équipe Bases de Données.
Adresse	L.R.I, Bât 490, Université de Paris-Sud, 91405 Orsay Cedex

Organisme de rattachement financier de l'équipe pour le présent projet

Université Paris-Sud

Responsable du projet au sein de l'équipe ou du laboratoire

M. ou Mme. Prénom Nom	Véronique Benzaken
Fonction	Professeur
Téléphone	01 6915 6628
Fax	01 6915 6586
Mél	veronique.benzaken@lri.fr

Membres de l'équipe participant au projet (y compris le responsable)

Nom	Prénom	Poste statutaire	% du temps de recherche consacré au projet
Benzaken	Véronique	Professeur	30%
Bidoit	Nicole	Professeur	30%
Burelle	Marwan	Doctorant	30%
Miachon	Cédric	Doctorant	75%
Objois	Matthieu	Doctorant	40%
Thion	Virginie	Doctorant	25%

Références :

1. Semantic Subtyping, A. Frisch, G. Castagna, V. Benzaken, IEEE Symposium on Logic in Computer Science (LICS) 2002
2. CDuce: an Xml-centric general-purpose language, ACM International Conference on Functional Programming (ICFP) 2003
3. Benchmarking Queries Over Trees: Learning the hard truth the hard way, F. Wattez, S. Cluet, V. Benzaken, C. Fiegel et G. Ferran, Intl Conference on Managment of Data (Sigmod) 2000
4. Static Management of Integrity Constraints in Object-Oriented Database Systems: Design and Implementation, V. Benzaken and X. Schaefer, Intl Conf on Extending Database Theory (EDBT), 1998
5. Persistent Object Systems Proceedings of the sixth International Workshop on Persistent Object Systems, M.P. Atkinson, V. Benzaken et D. Maier. Workshop in Computing Series, Springer Verlag 1994.
6. Fixpoint calculus for querying semistructured data, N. Bidoit, M. Ykhlef, Int. Workshop on World Wide Web and Databases (WebDB) 1998
7. A first step toward implementing dynamic algebraic dependencies, N. Bidoit, S. De Amo, Theoretical Computer Science, 190, (1998) 115-149
8. Implicit Temporal Query Languages: towards completeness, N. Bidoit, S. de Amo, Foundation of Software Technology and Theoretical Computer Science (FSTTCS) 1999

Action Concertée Incitative

MASSES DE DONNÉES

Descriptif complet du projet

B - DESCRIPTION DU PROJET

Sites et participants

1. **Projet GEMO**, INRIA Futurs & **LIAFA**, équipe Vérification:

   Luc SEGOUFIN[5] (CR INRIA), Ioana MANOLESCU (CR INRIA), Anca MUSCHOLL (Pr), Andrei ARION (Doc), Mathias SAMUELIDES (Doc)

2. **LIENS**, Laboratoire d'Informatique de l'ENS, équipe Langages:

   Giuseppe CASTAGNA[5] (CR CNRS), Alain FRISCH (Ingénieur corps des Télécoms)

3. **LIF**, Laboratoire d'Informatique Fondamentale de Marseille, équipe Move:

   Silvano DAL ZILIO[5] (CR CNRS), Denis LUGIEZ (Pr), Charles MEYSONNIER (Doc)

4. **Equipe MOSTRARE**, INRIA Futurs & **LIFL**, Laboratoire d'Informatique Fondamentale de Lille, équipe STC:

   Anne-Cécile CARON[5] (MdC), Sophie TISON (Pr), Joachim NIEHREN (Ing.Exp. INRIA), Jean-Marc TALBOT (MdC), Iovka BONEVA (Doc), Denis DEBARBIEUX (Doc)

5. **LRI**, Laboratoire de Recherche en Informatique, équipe Bases de Données:

   Véronique BENZAKEN[5] (Pr), Nicole BIDOIT (Pr), Marwan BURELLE (Doc), Cédric MIACHON (Doc), Matthieu OBJOIS (Doc), Virginie THION (Doc)

Partenaires étrangers

Haruo Hosoya (Univ. Tokyo – Japon)

Benjamin C. Pierce (Univ. Pennsylvania – USA)

Thomas Schwentick (Univ. Marburg – Allemagne)

Résumé des objectifs

Le constat qui motive cette ACI est l'explosion du nombre d'applications qui produisent, consomment et manipulent de grandes « *masses de données* ». Dans de nombreux cas, il s'agit de données brutes, ou issues de la fusion de sources d'informations hétérogènes, et qui ne possèdent donc pas de critères de classification communs. Ces caractéristiques expliquent pourquoi il est nécessaire dans ce cas d'utiliser un modèle de représentation des données plus flexible que le modèle relationnel classique, modèle couramment qualifié de semi structuré, dont XML est un des exemples les plus connus.

Notre projet se propose d'étudier les aspects de traitement, d'interrogation et de manipulation de grandes masses de données lorsque celles-ci sont disponibles au format XML. Nous nous intéressons plus précisément aux aspects langages de programmation et langages de requêtes. Notre ambition est de couvrir de manière intégrée un large spectre de problématiques: de celles liées aux **aspects langages** (expressivité, typage, étude de nouvelles primitives de programmation, logiques sous-jacentes pour l'interrogation, optimisation logique), jusqu'aux aspects traitant l'**accessibilité des données** (données en streaming, compression, définition de modèles d'accès en mémoire secondaire, utilisation de moteurs de persistance), en passant par les problématiques liées à **l'implantation** (compilation du filtrage, optimisation physique, vérification du sous-typage, modèles d'exécution pour le streaming).

Nous attaquerons ces problématiques en organisant notre recherche selon trois axes directeurs: langages de requêtes, traitement de données à la volée et typage de documents.

[5]Responsable de site

langages de requêtes: une des particularités du modèle relationnel [AHV95] est de proposer formellement la définition de langages de requêtes fondés soit sur l'algèbre relationnelle, soit sur des calculs relationnels. L'algèbre relationnelle consiste en un ensemble d'opérateurs génériques (projection, sélection, produit cartésien, ...) qui manipulent des relations, tandis que les calculs relationnels sont fondés sur la logique du premier ordre. Deux caractéristiques importantes de ces langages sont que: (i) ils sont déclaratifs, dans le sens où on décrit le résultat et non la manière de l'obtenir; (ii) leur expressivité intrinsèque est limitée, en particulier ils ne sont pas Turing-complets. La « simplicité » de ces langages permet d'obtenir de bonne performances, performances qui peuvent être encore améliorées en exploitant des propriétés algébriques des opérateurs (optimisation logique) et également en se basant sur des techniques de gestion de la mémoire secondaire (optimisation physique). Notre but est d'arriver, ou tout au moins de s'approcher, à la définition d'un modèle pour XML ayant ces mêmes caractéristiques. Nos objectifs se déclinent de la manière suivante : étude théorique de paradigmes d'interrogation sous l'angle de l'expressivité et de la complexité; définitions de langages de requêtes pour XML ayant les caractéristiques d'expressivité restreinte et de déclarativité typiques des langages du modèle relationnel ; implantation puis conception et validation de techniques d'optimisations adaptées à ces différents paradigmes.

traitement des données à la volée (streaming): cette technique permet de travailler sur des flots de données, sans avoir à charger entièrement les documents en mémoire, ce qui n'est pas toujours possible en général. On peut aussi envisager de travailler à la volée sur des données compressées. C'est donc un sujet essentiel dans le contexte des grandes masses de données. Un des verrous de cet axe de recherches est d'identifier une bonne classe de requêtes qu'il est possible d'évaluer par « streaming », avec ou sans compression des données, et dans le prémier cas de déterminer la granularité optimale de compression.

contraintes et typage de documents: les systèmes de types servent en premier lieu à la validation des documents et à la vérification de contraintes d'intégrité, mais, comme c'est le cas avec les langages de programmation conventionnels, les types sont aussi une source importante d'optimisation, ce qui fait de l'étude du typage un des objectifs évident de notre projet.

Une seconde motivation pour étudier le typage est que nous sommes intéressés par des contraintes d'intégrité satisfaites indépendamment de l'ordre d'apparition des champs dans un document, une situation à l'opposé de ce que l'on trouve avec les « systèmes de types classiques » pour XML, tel que les DTD par exemple. Ce choix est naturel lorsqu'on travaille avec des données obtenues à partir de la fusion de bases de données relationnelles (un cas important de documents de grande taille), puisque l'ordre des champs est alors sans intérêt.

Les équipes impliquées dans notre projet ont déjà, chacune de leur côté, travaillé sur le thème de la manipulation des documents XML. Ce n'est pas l'unique raison de notre regroupement. En effet, la cohésion de notre groupe est assurée par l'utilisation d'une même approche fondamentale, la théorie des automates d'arbres et les logiques associées, et par un intérêt commun pour les langages de requêtes et la validation de documents: typage, contraintes d'intégrité, techniques de compilation et d'optimisation.

Outre une unité méthodologique, et une unité dans les objectifs, la coopération au sein du projet est accrue par la décision de choisir une cible logicielle unique à nos efforts d'implantation, le langage CDuce, qui est développé conjointement par le LIENS et le LRI, deux des sites participants à ce projet.

Différences par rapport à la proposition 2003

Outre les modifications naturelles des perspectives et des objectifs, dues à l'évolution de nos recherches dans le temps qui s'est écoulé, cette proposition présente une nouveauté majeure. Par rapport à la proposition de 2003, nous avons renforcé l'axe langage de requêtes. En effet, la proposition précédente était essentiellement ciblée sur les aspects logiques, syntaxiques et d'optimisation logique du langage de requêtes. Nous avons ajouté à cette proposition la définition de techniques d'accès aux documents XML en mémoire secondaire, la définition et l'implantation de techniques et d'algorithmes d'optimisation physique des requêtes ce dans le but final d'obtenir un optimiseur de requêtes générique pouvant être paramétré en fonction des besoins et caratéristiques des applications visées.

D'un point de vue de l'organisation nous avons procédé à un rééquilibrage des tâches, une réorganisation des ressources et une uniformisation des coûts, avec un soin particulier porté à la transparence et à la lisibilité de ces derniers (voir la section *Critères pour la détermination des moyens financièrs* dans la partie C).

B1 – Objectifs et contexte:

Le constat qui motive cette ACI est l'explosion du nombre d'applications qui produisent, consomment et manipulent de grandes « *masses de données* ». Dans de nombreux cas, il s'agit de données brutes, partiellement spécifiées ou entachées d'erreurs, ou encore de données obtenues par la fusion de sources d'informations hétérogènes, qui ne possèdent pas de critères de classification communs. Ces caractéristiques expliquent pourquoi l'utilisation du modèle relationnel classique se révèle parfois impropre à la manipulation de ce type de données, voire à leur stockage.

Un constat supplémentaire est que, parmi les applications émergentes qui manipulent de grands volumes de données, beaucoup manipulent des données irrégulières et complexes. Au problème de la taille des données, s'ajoute le problème de la complexité de description (du « schéma ») de ces données. Dans ce contexte, il est nécessaire d'utiliser un modèle de représentation des données plus flexible, couramment qualifié de modèle de *données semi structurées*, dont XML est un des exemples les plus importants.

Notre projet se propose d'étudier les aspects de traitement, d'interrogation et de manipulation des grandes masses de données, lorsque celles-ci sont disponibles au format XML. Nous nous intéressons plus précisément aux aspects langages de programmation et langages de requête. Notre ambition est de couvrir un large spectre de l'univers « langages pour XML », en explorant un nombre important de facettes différentes qui vont des aspects logiques aux types, du *design* de langages à leur implantation, jusqu'aux differents aspects des données qu'ils manipulent: compression, persitance, optimisation physique.

1 Masses de données semi structurées

Les données semi structurées [ABS99] peuvent se voir comme une relaxation du modèle relationnel classique, un des fondements des bases de données traditionnelles, dans lequel on autorise une structure moins rigide et homogène des « champs de données ». Ce modèle s'est révélé très utile dans la représentation de familles de documents variés: multimédia, hypertexte, données scientifiques, ...

Le langage XML, acronyme de *eXtensible Markup Language* [XML98], est un format textuel qui permet de créer des documents contenant des données semi structurées. Une de ses caractéristiques est d'être auto-descriptif: les données ont une structure d'arbre contenant des « balises » qui informent sur la structure et la sémantique des documents. Ces informations peuvent être exploitées à l'aide de technologies Bases de Données.

Conçu à l'origine comme un dérivé simple et flexible de la norme SGML, le format XML joue un rôle croissant dans l'échange d'informations sur le Web et s'est imposé comme un pilier de l'initiative du *Web Sémantique* [BL98]. Autre exemple de « masses de données XML », l'explosion programmée d'Internet comme support de calculs globaux [GRI]. Mais l'utilisation du format XML n'est pas restreinte au seul domaine des « applications Internet, » et on peut trouver d'autres exemples qui génèrent et manipulent de grands volumes de données (semi structurées). Par exemple dans le domaine des Systèmes d'Information Géographique [GIS]: cadastre, outils de planification urbaine, applications géologiques, ou bien encore en bioinformatique, avec le séquençage des différents génomes ou les efforts de modélisation en biologie moléculaire [GO].

Un développement durable de ce type d'applications dépend donc de la qualité des outils que nous pourrons produire pour manipuler les données semi structurées. On retrouve dans cette catégorie les exemples classiques d'outils de l'informatique théorique: compilation, filtrage, systèmes de types, et ainsi de suite. Cependant XML fournit un nouveau domaine d'étude, pour lequel il n'est pas possible de réutiliser telles quelles les technologies déjà existantes. Il nous faut donc développer de nouveaux fondements théoriques, si possible en s'inspirant de méthodes utilisées dans d'autres domaines de l'informatique.

2 État de l'art

Les recherches dans le domaine des langages pour la manipulation de documents XML sont loin de la maturité, ce qui est accrédité par le nombre croissant des travaux scientifiques dans ce domaine et par la grande perméabilité qui existe encore entre travail académique et prototypes industriels. La situation est encore plus sommaire lorsqu'il s'agit de prendre en compte les problèmes liés aux grands volumes de données. Cet état de fait est en partie lié au caractère très général et flexible de XML, qui a été créé dans l'esprit de s'accommoder d'une multitude d'utilisations distinctes.

Le développement autour des données semi structurées et XML peut être grossièrement regroupé en deux catégories :

1. les efforts de standardisations afin d'obtenir des outils de requêtes et de validation des documents. Les recherches qui nous intéressent découlent de modèles théoriques basés sur des automates d'arbres et diverses formes de logiques,

2. les implantations de systèmes et de prototypes, qui tentent de répondre aux problèmes d'optimisation et d'indexation.

2.1 Validité de documents: DTD, schémas et algèbres d'arbres

XML emprunte en grande partie sa simplicité et sa syntaxe à SGML. En particulier, un document XML se présente à première vue comme une succession de balises imbriquées, ou plus abstraitement comme un arbre étiqueté. En cela XML ressemble beaucoup à HTML (un autre dérivé de SGML) mais il diffère de celui-ci par une caracteristique fondamentale, qui est que les balises n'ont pas de sémantique définie a priori, tandis qu'en HTML chaque balise a un sens très precis. Cette possibilité qui est à l'origine même de la flexibilité de XML doit toutefois être structurée . Ainsi, afin d'imposer un peu plus de structure, on associe couramment à un document XML un « schéma », l'équivalent d'un type.

Le premier mécanisme de validation de documents, qui a été défini en même temps que la spécification de XML, est le formalisme des DTD, ou *Document Type Definition*. Les DTD sont basées sur un langage qui s'apparente aux expressions régulières, elles définissent l'ensemble des balises qu'il est possible de trouver dans un document valide et établissent des contraintes sur l'ordre d'apparition de ces balises.

Les types définis par les DTD sont parfois trop rigides pour être utilisables dans la pratique. Par exemple, on obtient souvent un document invalide après avoir permuté l'ordre des balises d'un document valide. Un nouveau standard, *XML-Schema* [XS00], a été proposé pour palier à ces inconvénients. Il existe en fait toute une famille de langages de schémas pour XML [CM01, KMS], XML-Schema étant pour le moment la seule extension des DTD à être retenue par le W3C.

Les automates d'arbres, et les logiques associées, fournissent un bon outil théorique pour comprendre ces « systèmes de types » pour XML. En effet, la représentation des données semi structurées se base couramment sur des arbres étiquetés, ou des arbres avec pointeurs. Il est donc normal de penser à utiliser les automates d'arbres pour étudier ces données, et d'appliquer la correspondance classique qui relie automate, logique et langage de requêtes. On peut faire ici un parallèle avec les liens qui relient les expressions régulières aux automates à états finis ou encore les grammaires hors contextes aux automates à piles. Cette connexion a été suivie par de nombreux chercheurs. Citons par exemple les travaux de M. Murata sur les « Hedge automata » [CM01, Mur01], à la base du langage de schéma RELAX, ou des travaux effectués par les équipes de notre projet [DL03, MSS03, Seg03].

Projets similaires, nationaux et internationaux.
Notre projet regroupe certainement l'ensemble des équipes nationales qui travaillent dans le domaine des systèmes de types et des fondements logiques de XML. La seule exception, à notre connaissance, est la collaboration de J. Vouillon (PPS) avec les membres du projet Cristal de l'INRIA pour permettre l'interaction des langages OCaml et XDuce (voir section suivante). Nous avons également une bonne implication au niveau international, à travers les collaborations développées par chacune des équipes: projet Procope avec l'université de Marburg, collaboration avec V. Vianu (UCSD), collaboration avec B. Pierce (UPenn), collaboration avec H. Hosoya (U. Tokyo), collaboration avec M. Benedikt des Bell-labs (Lucent).

2.2 Langages de programmation

Les premiers travaux concernant les langages pour la manipulation de documents XML, de loin les plus nombreux, sont issus de la communauté « bases de données ». Ils concernent surtout les langages d'interrogation et de manipulation des documents hypertexte (moteurs de recherches, data mining) et les outils d'intégration de données. Certains des langages développés pour les documents hypertexte sont historiquement à la base de technologies d'extraction et de transformations de documents XML actuels [XPa, XQu01, XSL99].

Une série de travaux plus récents est issue de la communauté « langages de programmation ». Dans ce contexte, l'objectif général est de fournir de nouveaux opérateurs pour les langages de programmation manipulant des données XML. On peut isoler trois approches différentes:

Approche librairies externes. La première approche consiste simplement à ajouter les documents XML comme type de base de langage de programmation classique [JAXb, Lis, DOM] par le biais de librairies externes. Les résultats obtenus sont loin d'être totalement satisfaisants. Dans le cas de DOM (Document Object Model) par exemple, une API pour XML très répandue, il est impossible de garantir statiquement que le resultat d'une transformation d'un document valide (par rapport à une certaine DTD ou à une autre notion de types) soit lui aussi valide.

Approche langages. La seconde approche, plus fondamentale, consiste à définir un nouveau langage statiquement typé, dédié à la transformation de documents XML. Deux principaux courants de recherches apparaissent selon la manière dont est traité le typage :

- une *approche immersive*, où les types XML sont plongés dans le système de type du langage hôte: Relaxer [MA00], JAXB [JAXa] basés sur Java; HaXML [WR99] basé sur Haskell; Xtatic [GP03] et Xen[BMS03] basés sur C#.

- une *approche orientée XML*, où des langages totalement nouveaux sont définis à partir de systèmes de types spécifiques à XML.

Les précurseurs dans l'approche orientée XML sont les langages XDuce [XDu], XQuery [XQu01, FSW00a] et XMλ [MS99], qui sont tous des langages fonctionnels avec des opérations de filtrage (*pattern-matching*) spécialisées.

Notre projet s'articule autour du langage de programmation CDuce [CDu], une extension du langage XDuce de Pierce et Hosoya, qui est développé conjointement par l'équipe Langages du LIENS et l'équipe Bases de Données du LRI. Comme son aîné, CDuce est basé sur le paradigme fonctionnel, un modèle de programmation qui s'est révélé particulièrement bien adapté à la manipulation des structures de données arborescentes. D'ailleurs, la plupart des langages de manipulation de documents XML, tel que XSLT et XQuery, sont des langages fonctionnels.

Approche programmation logique. Finalement, on peut isoler une approche plus « bases de données traditionnelles », parfois inspirée du style de la programmation logique, dont un exemple est donné par le langage de requêtes TQL, récemment proposé par Cardelli et Ghelli et qui se base sur une logique modale d'arbres [CG01]. Dans ce cadre, l'idée est de définir un équivalent de Datalog pour les données semi structurées. Il existe d'autres « approches logiques » aux problèmes qui nous intéressent, comme par exemple [BY98, FSW00b, HVJT01].

Conclusion. La première de ces trois approches possède l'avantage d'être plus simple à implanter et de pouvoir immédiatement bénéficier de tout le développement existant pour le langage hôte. Toutefois, dans la pratique, elle tend à introduire de la structure inutile, qui entame fortement la lisibilité et la facilité d'écriture et de réutilisation du code. C'est pourquoi la deuxième approche nous paraît plus intéressante, l'idée directrice étant que les langages qui manipulent des documents XML doivent prendre sérieusement en compte les types XML: DTD, XML-Schema, RELAX-NG, etc.

Projets similaires, nationaux et internationaux.
Les projets les plus proches du nôtre dans ce domaine sont sans aucun doute les projets de développement des langages XDuce (U. Tokyo), Xtatic (U. Pennsylvania, USA) et TQL (U. Pisa, Italie et Microsoft Research, UK), avec qui nous collaborons régulièrement. En particulier, les responsables de XDuce et Xtatic, H. Hosoya et B. Pierce, sont deux des membres étrangers associés à notre projet. Toujours en relation, mais plus éloignés, sont les travaux sur Xen (Microsoft Research), J-wig (U. Aharus, Danemark). Pour terminer, XQuery mérite une attention particulière car, en plus d'être assez proche de notre domaine, il est également le langage de réference pour les requêtes XML, ce qui le rend incontournable. C'est pourquoi nous entretenons des contacts réguliers (visites de longue durée) avec Jerôme Siméon et Phil Wadler.

2.3 Langages et optimisation de requêtes

Dans le contexte de données au format XML, la frontière entre langages de programmation, langages de transformations et langages de requêtes n'est pas aisée à tracer et comme le fait remarquer Victor Vianu [Via01], il n'existe pas à ce jour de standard définitif. Ces langages peuvent être classés en trois catégories :

- **langages d'adressage** dans cette catégorie les requêtes naviguent dans les documents et ne font qu'extraire de l'information comme par exemple: *Pour toute personne dans une liste, donner les textes correspondant à tous ses éléments* `<prenom>`. Le standard pour de tels langages est XPath [XPa] qui sert par ailleurs de brique de base pour d'autres langages dce requêtes tels que XQuery.

- **langages de transformations** dans cette catégorie, non seulement les requêtes extraient de l'information (souvent en utilisant XPath) mais également la restructurent comme par exemple : *Mettre chaque element* `<person>` *d'une liste satisfaisant une condition donnée sous un nouveau tag* `<result>`. Le standard pour cette catégorie est XSLT [XSL99].

- **langages de requêtes** Ces derniers sont en général un mélanges des deux premières catégories. Le standard potentiel pour cette dernière catégorie est XQuery.

Après une comparaison de différentes propositions [FSW+99], le W3C a mis en évidences plusieurs recommandations pour la définition de tels langages [CFMR03]. En particulier, les requêtes doivent pouvoir être exécutées sur

un unique documents comme sur plusieurs documents, elles doivent pouvoir retourner tout ou partie du document, les conditions de filtrage doivent pouvoir porter tant sur le contenu que sur la structure. Plus précisément sans toutefois être exhaustifs, les points suivants sont prioritaires :

- importance de la déclarativité. Une requête doit être décomposée en trois parties: un *pattern* permettant de lier des variables à des portions du document, un *filtre* qui correspond à une condition de sélection, et une partie *construction* qui permet de restructurer le résultat.
- présence des quantificateurs universels et existentiels, jointures entre différents documents, requêtes imbriquées, tri.

De telles recommandations ne sont guère surprenantes et correspondent à l'expérience de la communauté bases de données en ces matières. D'un point de vue historique, depuis la définition de SQL en passant par celle d'OQL [BDK92] jusqu'à la définition de langages de requêtes pour XML, les mêmes principes ont guidé et guident les concepteurs.

Les résultats existants en matière d'optimisation de requêtes sur des données (XML) persistantes peuvent être classifiés en fonction du modèle sous-jacent de stockage et de traitement de requêtes. Des travaux comme [FK99, SGT+99] se sont appuyés sur des systèmes de gestion de bases de données relationnelles pour stocker des documents XML, et ont traité des requêtes XML en les traduisant en SQL. En conséquence, tous les aspects reliés au placement des données, au stockage, et à l'optimisation de requêtes ont été délégués au système relationnel. Cette approche a ses limites, puisque des langages de requêtes XML plus récents (tels que XQuery et CDuce, ciblés par notre projet) sont bien plus complexes, et très différents de SQL: par exemple, ces langages sont basés sur un modèle de données arborescent, pas sur des tuples, et favorisent des listes imbriquées comme brique de base de l'interrogation, et non pas des multi-ensembles plats, non-ordonnés, comme c'est le cas en SQL.

Pour échapper à ces limitations, plusieurs systèmes de stockage persistant et de traitement de requêtes XML *natifs* ont été développés récemment, parmi lesquels le système Timber (de l'Université de Michigan) est le plus évolué. Ces systèmes proposent des approches d'optimisation ciblées sur un sous-ensemble du langage de requêtes XQuery.

Projets similaires, nationaux et internationaux.
Les projets similaires au nôtre pour la définition de langages de requêtes sont: XML-QL [DFF+98], XQL [XQL99], Quilt [CFR00] et sa variante Kweelt [Kwe01]. Ces langages présentent surtout un intérêt historique, car leur étude a pris fin avec l'élaboration de XQuery, qui s'en est clairement inspiré (les auteurs de ces langages font tous partie du groupe de standardisation de XQuery).
Un langage similaire mais plus simple, X-OQL [ABCK02] a été développé à l'INRIA, sans relation avec XQuery, et est encore utilisé par la compagnie Xyleme.
Enfin, XSQuirrel [FS03] est un projet plus récent, mais qui se limite à un langage d'adressage.
Pour ce qui concerne la partie optimisation de requêtes sur XML nous avons déjà cité le système Timber. Le travail que nous proposons est plus ambitieux, car plus générique, de deux points de vue. D'abord, nous étudierons le problème de l'optimisation de requêtes XML en restant aussi générique que possible en ce qui concerne le langage d'interrogation; des langages différents seront mieux adaptés à des contextes différents, d'où le besoin de généricité. De même, il n'existe pas actuellement de consensus sur le meilleur format de stockage de documents XML; en conséquence, notre travail sur l'optimisation se basera sur un modèle générique de stockage, ce qui permettra (*i*) d'exploiter la totalité des modules de stockage et d'indexation disponibles, et (*ii*) de proposer, dans une étape ultérieure, les modules de stockage et d'indexation permettant le traitement efficace d'un ensemble de requêtes donné.

3 Verrous scientifiques

On peut identifier les verrous scientifiques de notre domaine de recherches avec les limitations des outils actuellement disponibles. À chacune de ces limitations correspond un thème de notre projet.

3.1 Un langage de requêtes expressif et optimisable

Les outils disponibles actuellement sont d'un certain point de vue trop expressifs. Ainsi, il est facile de montrer que les langages XSLT et XQuery, par exemple, sont Turing complets. Cette puissance d'expression a son revers, puisque beaucoup de problèmes d'optimisation et de validation sont indécidables. Les outils d'optimisation étant particulièrement importants lorsqu'on traite des grandes masses de données, on voudrait pouvoir se rabattre vers un noyau de requêtes plus simples, ayant toutes les bonnes propriétés qu'on retrouve dans les langages de requêtes pour les bases de données traditionnelles: on veut un noyau simple (facilement utilisable), efficace (pouvant s'évaluer rapidement) et dont on comprend les avantages et les limitations (à l'aide d'outils logiques).

Enfin, la plupart des optimiseurs de requêtes pour XML sont liés soit à un langage particulier soit à un type de stockage particulier. Or la nature très diverse des applications gérant des données XML entraine des besoins différents tant en terme de langages de requêtes qu'en terme de techniques de stockage . Nous proposons de définir et d'implanter un optimiseur de requêtes générique tant au plan logique (paramètrage du langage utilisé) qu'au plan physique (règlage fin et prise en compte des différentes techniques de stockage).

Toutes les équipes impliquées dans ce projet ont participé à des recherches sur les fondements de langages de requêtes pour XML, citons en particulier [BY98, Seg03] ainsi qu'à leur optimisation [SWK+02, ABM94, WCBF00, BDH92]

3.2 Dompter l'utilisation de la mémoire

Une autre limitation qui frappe les « outils XML » existants concerne les problèmes de gestion de la mémoire. Deux solutions s'imposent si on cherche à travailler sur de grands volumes de données, tout en limitant l'utilisation des ressources mémoires: (i) compresser les données [LS00] et (ii) faire de l'évaluation de requêtes à la volée, c'est-à-dire sans avoir à charger entièrement un document en mémoire. Des travaux sur la caractérisation des classes de requêtes que l'on peut évaluer à la volée ont été mené au sein de l'équipe Gemo [SV02]. On peut même envisager de combiner ces deux approches et travailler à la volée sur des données compressées.

Le format XML est un format textuel qui contient beaucoup de régularités et d'informations redondantes. La compression des documents XML est donc une tâche aisée, qui donne de bons résultats en pratique: des taux de compressions supérieur à 80% ont été obtenus par diverses universités. Cependant, la tâche devient beaucoup plus difficile dès lors qu'on veut pouvoir évaluer les requêtes sans décompresser les documents, ou en ne les décompressant que de manière limitée.

La recherche sur le streaming des données XML est très active, et il reste encore beaucoup de points à éclaircir, en particulier sur les fondements théoriques. Il n'existe par contre que quelques approches qui permettent de travailler sur des données compressées, et ce domaine est riche en problèmes à résoudre. Un des verrous de cet axe de recherches est d'identifier de bonnes classes de requêtes qu'il est possible d'évaluer à la volée, avec ou sans compression des données.

3.3 De meilleurs types pour les documents

Comme c'est le cas avec les langages de programmation conventionnels, les types sont une source importante d'optimisation, ce qui fait de l'étude du typage un des objectifs évidents de notre projet. Cependant, le système des DTD, le formalisme de validation des documents le plus ancien et aussi le plus accepté, est extrêmement frustre. Un exemple de limitation des systèmes de types dérivant des DTD est donné par le fait qu'ils imposent un ordre sur les balises des documents, même si on ne le désire pas. Or, lorsqu'on utilise XML pour coder le résultat de la fusion de données relationnelles par exemple (un cas important de documents de grande taille), l'ordre des balises n'est pas spécifié. Des travaux sur des systèmes de types pour XML qui ne contraignent pas l'ordre des balises ont été menés au sein de l'équipe Move du LIF [LD02, DL03, DLM04].

XML-Schema, un langage de schémas plus récent que les DTD, permet de définir des types et de prendre en compte des contraintes plus riches. En particulier des contraintes d'ordre (ou de non ordre) sur les balises, ou des contraintes de référence. Cependant la plupart des langages de transformation sont non typés, alors même que, comme le reflète le processus actuel de standardisation, les documents XML sont intrinsèquement typés. Nous pensons qu'un langage de programmation pour XML doit prendre en compte ces types XML (DTD, XML-Schema).

Un des verrous de notre projet est donc d'étudier des systèmes de types, et des relations de sous-typage, plus avancés que ceux actuellement disponibles.

B2 – Description du projet:

Les équipes impliquées dans notre projet ont déjà, chacune de leur côté, travaillé sur le thème de la manipulation des documents XML. Ce projet s'inscrit dans la continuité directe de ces travaux et chacun des sites partenaires contribue de manière spécifique à ce projet au travers de ses compétences et des recherches déjà menées. Cependant il ne s'agit pas de l'unique raison de notre regroupement. En effet, la cohésion de notre groupe est assurée par l'utilisation d'une même approche fondamentale, la théorie des automates d'arbres et les logiques associées, et par un intérêt commun pour les langages de requêtes et la validation de documents: typage, contraintes d'intégrité, ... C'est à l'intersection du travail réalisé par chacune des équipes que nous avons identifié trois axes de recherches spécifiques, liés au traitement des grandes masses de données, et sur lesquels nous nous proposons de travailler. Nos objectifs cherchent à répondre aux questions soulevées à l'annexe B1:

- Peut-on identifier un sous-ensemble de requêtes, sous la forme d'un langage, qui soit à la fois simple (facilement utilisable), efficace (pouvant s'évaluer rapidement, avec des automates d'arbres par exemple) et dont on comprend les avantages et les limitations (à l'aide d'outils logiques)?

- comment étendre les résultats et les outils théoriques connus au cas où les documents doivent être traités à la volée? Comme on s'intéresse au cas où les données sont compressées, peut-on traiter les documents à la volée directement sous leur forme compressée, et quel fragment des requêtes peut-on espérer implanter de cette manière?

- comment prendre partie du typage des documents pour optimiser les requêtes? On peut imaginer utiliser des informations de types explicite, à partir d'un schéma, ou implicite, à partir de contraintes d'intégrité extraites des données qui sont traitées. Comment étendre les résultats et les outils théoriques connus au cas où l'ordre d'apparition des balises dans les documents n'est pas important?

Ces trois objectifs ne constituent pas trois buts totalement séparés. Par exemple, les optimisations du langage de requêtes dépendent d'informations de types, et les choix effectués dans l'élaboration du système de types dépendent du langage choisit.

Ensuite, il y a une unité dans les outils fondamentaux que nous utilisons, qui sont les automates d'arbres et les logiques associées, comme l'atteste les publications de chaque équipe [CDJ+99, DL03, LD02, Seg03, MSS03, JMTT00]. Bien qu'on puisse affirmer que l'application des automates d'arbres à XML est une approche reconnue et assez bien établie, nos propositions permettent d'isoler des questions qui ne sont pas encore résolues sur le plan théorique.

Finalement, il y a une unité dans le contexte applicatif, avec le choix du langage CDuce. Ainsi, les sites participant se sont accordés sur le principe de proposer des solutions ayant comme « cible » une extension du langage CDuce.

1 Étendre le langage CDuce

Responsable: Giuseppe Castagna.

Le cadre pratique de notre étude est le langage CDuce. Outre la proximité scientifique, plusieurs raisons plaident pour le choix de ce langage CDuce. Tout d'abord, CDuce est un langage généraliste et d'ordre supérieur, contrairement aux autres langages de sa catégorie, tel XDuce [XDu] et XQuery [XQu01]. Ensuite, CDuce possède l'algèbre de types la plus riche parmi les langages de transformation pour XML [BCF03]: types produit, flèche, union, intersection, différence, expression régulières de types, et types récursifs. CDuce possède aussi un mécanisme de filtrage de motifs très puissant et offre la possibilité de définir des fonctions surchargées. Finalement, les fondements formels du système de types de CDuce permettront de bâtir les développements futurs prévus dans le cadre de notre projet sur un solide socle théorique.

Conformément à l'esprit des ACI, ce projet vise essentiellement à développer une recherche amont et s'inscrit en complémentarité de projets à vocation plus pratique. Citons par exemple un projet RNTL entre le LIENS, le LRI et la société Brixlogic, pour le développement d'interfaces graphiques pour CDuce appliqué au domaine des applications financières. Dans ce type de projets, plus applicatifs, on se sert de CDuce comme d'une base, alors que, dans le projet que nous portons, le langage CDuce est un objet d'études à part entière. C'est donc une occasion de développer et d'étendre les fondements du langage CDuce, plutôt que de simplement « utiliser » le langage.

Ainsi nous envisageons d'étendre CDuce avec le plus grand nombre possible de fonctionnalités que nous décrivons ici de suite (requêtes, interfaçage avec moteurs de persistance, gestion du streaming avec ou sans compression, typage de l'interleaving, optimisation). Il y a toutefois un certain nombre d'extensions que nous envisageons de développer

dans le cadre de ce projet qui sont plus spécifiques à CDuce. Parmi celles-ci, l'extension de CDuce par le polymorphisme, l'extension de l'algèbre de filtrage de CDuce pour permettre de filtrage "en profondeur" sur les arbres XML, l'interfaçage avec d'autres langages de programmation (en particulier OCaml) et l'"error mining" (la capacité de détecter des erreurs de programmation qui formellement n'en sont pas, telle la définition de requêtes dont le résultat est toujours vide). Ces extensions trouvent toute leur place dans ce projet car les autres volets de ce projet bénéficieront de ces avancées. Ainsi, par exemple, l'extension de l'algèbre de filtrage pourra être intégrée dans un langage de requêtes basée sur CDuce tandis que le polymorphisme pourra être étudié en combinaison avec le typage de l'interleaving. De manière similaire les opérateurs algébriques au sein du langage de requêtes doivent être génériques : une forme de polymorphisme doit être introduite et des mécanismes d'inférence de type devront également être définis afin de permettre d'inférer le type du résultat des requêtes, et l'étude sur le polymorphisme de CDuce pourra être utilisé comme base de départ d'une telle étude

2 Langage de requêtes

Responsable: Véronique Benzaken.

Une des particularités du modèle relationnel [AHV95] est de proposer formellement la définition de langages de requêtes fondés soit sur l'algèbre relationnelle, soit sur des calculs relationnels. L'algèbre relationnelle consiste en un ensemble d'opérateurs génériques (projection, sélection, produit cartésien, ...) qui manipulent des relations, tandis que les calculs relationnels sont fondés sur la logique du premier ordre. Ces langages théoriques constituent le coeur du standard SQL. Deux caractéristiques importantes de ces langages sont que: (i) ils sont déclaratifs, dans le sens où on décrit le résultat et non la manière de l'obtenir; (ii) leur expressivité intrinsèque est limitée, en particulier ils ne sont pas Turing-complets. La « simplicité » de ces langages permet d'obtenir de bonne performances, performances qui peuvent être encore améliorées en exploitant des propriétés algébriques des opérateurs (optimisation logique) et également en se basant sur des techniques de gestion de la mémoire secondaire (optimisation physique).

À ce jour, dans le contexte des données semi structurées, il n'existe aucun équivalent à ces langages de requêtes (aussi bien au sens algébrique, qu'au sens calcul) qui soit pleinement satisfaisant en terme de déclarativité et d'expressivité.

Nos objectifs se déclinent de la manière suivante : étude théorique de paradigmes d'interrogation sous l'angle de l'expressivité et de la complexité; puis conception et validation de techniques d'optimisations adaptées à ces différents paradigmes.

Expressivité, Complexité De façon générale (et générique) nous nous proposons d'étudier les problèmes d'expressivité sous un angle fondamental. L'idée est de reproduire le triangle relation – SQL – algèbre des bases de données relationnelles en remplaçant les relations par des documents XML, les algèbres par des automates d'arbres (Walk Tree Automata ou autres) et les calculs par une variante de la Logique Monadique du Second Ordre (MSO) et/ou des logiques modales.

Le coeur des langages de manipulation de données semi structurées repose sur le mécanisme de filtrage de motifs. Ce mécanisme repose lui-même (algorithmiquement) sur la notion d'automates d'arbres.

La logique monadique du second ordre (MSO) sur les arbres a été amplement étudiée. Une formule MSO avec n variables définit une requête pour un ensemble de n-tuples de noeuds dans un arbre. MSO est très expressive, même si elle n'est pas Turing complète. En vertu du théorème de Thatcher et Wright (1968) MSO peut exprimer toutes les requêtes régulières. Malheureusement, le problème de l'évaluation d'une requête MSO est de complexité non-élémentaire, et en conséquence ne peut pas être résolu efficacement.

Les requêtes monadiques régulières ont suscité beaucoup d'intérêt depuis le développement du Web [GK02b, BFG01]. Elles peuvent être formulées également par des automates d'arbres [NS02, NS98, FGK03, CNT04], des programmes Datalog monadiques [GK02a], et en logique MSO avec une seule variable. Des requêtes monadiques représentées par automates ou programmes Datalog monadiques peuvent être évaluées en temps linéaire – dans le produit de la taille de l'arbre et de la requête. Elles offrent une bonne efficacité mais une puissance expressive moindre, à cause de la limitation aux requêtes monadiques.

Dans un premier temps, nous souhaitons étudier les langages de requêtes n-aires [BS02]. Ces requêtes sont essentielles pour l'extraction d'information, par exemple, si on veut extraire tous les tuples produit-prix dans une page Web commerciale. Nous définirons des requêtes n-aires régulières par des automates d'arbres, programmes Datalog, des formules dans des logiques modales, et par des transducteurs. Le but est de trouver des langages de requêtes, qui sont à la fois expressifs, efficaces, et faciles à générer automatiquement (à partir des exemples, par spécification interactive ou apprentissage automatique). Enfin nous étudierons des requêtes numériques dans des arbres. Nous privilégierons dans un premier temps le cas particulier des requêtes sur des arbres non-ordonnés, utilisant l'approche de la logique modale, comme par exemple dans la logique des ambients [CGA+02, CG01, BT04].

Nous allons classifier systématiquement les problèmes algorithmiques soulevés par l'évaluation de ces langages de requêtes: la vérification de modèle, la satisfaisabilité, l'inclusion de deux requêtes, et aussi l'apprentissage de requête. Les sites Gemo, LIF, LIFL et LRI sont impliqués dans cette étude.

Optimisation. XML permet la représentation de données semi structurées sous une forme textuelle. L'optimisation de requête comporte deux aspects étroitement liés : l'optimisation logique et l'optimisation physique. La première est liée à la nature même du langage et vise à exploiter des réécritures afin d'obtenir une requête équivalente mais dont les performances seront meilleures, la seconde a pour objet d'exploiter la façon dont les données sont organisées sur disque afin de générer des plans d'exécution les plus efficaces possible.

Ainsi, en matière d'optimisation logique, afin d'améliorer les performances des requêtes, il faut adopter une représentation (intermédiaire) concrète adéquate. Les informations apportées par le schéma de la base (son type) peuvent guider cette décision. De même, la connaissance du type des données permet d'optimiser les requêtes avant leur exécution (par exemple, si l'on sait que telle balise est uniquement présente dans le sous-arbre droit du document il n'est pas nécessaire de parcourir le document intégralement). Ceci explique pourquoi nous cherchons aussi à améliorer les notions de typage dans ce projet (voir section 4). En particulier, avec la notion de sous typage sémantique définie dans CDuce [FCB02], nous disposons d'une algèbre adaptée aux problématiques classiques d'optimisation logique de requêtes et ce dans le cadre de bases de données semi structurées. La base sémantique constitue un terrain idéal pour l'étude tant de l'implantation que de l'optimisation. Il est donc très intéressant de considérer des extensions bases de données, telles que la définition d'un langage de requêtes et surtout de son optimisation.

Afin d'étudier la nature même des optimisations physiques à entreprendre il nous faut réaliser l'intégration d'un système de stockage persistent de données XML dans le moteur CDuce. Le système candidat est XQueC (GeX), développé au sein du projet Gemo. Ce système fournit un stockage persistent pour des documents XML, organisé selon les chemins présents dans l'arbre. Le système de stockage à concevoir devra permettre de charger les documents une seule fois pour des interrogations multiples, ainsi qu'un accès sélectif aux données (et donc une consommation mémoire réduite) lors du traitement des requêtes. En s'appuyant sur ce stockage nous proposons de définir et mettre en œuvre des techniques d'optimisation de requêtes, dans le sens algébrique employé classiquement dans les systèmes de bases de données.

Enfin, la plupart des optimiseurs de requêtes pour XML sont liés soit à un langage particulier soit à un type de stockage particulier. Or la nature très diverse des applications gérant des données XML entraîne des besoins différents tant en terme de langages de requêtes qu'en terme de techniques de stockage . Nous proposons de définir et d'implanter un optimiseur de requêtes générique tant au plan logique (paramétrage du langage utilisé) qu'au plan physique (réglage fin et prise en compte des différentes techniques de stockage).

Parmi les résultats attendus dans cet axe de recherches, nous chercherons à fournir une étude comparative des langages XDuce, XQuery et CDuce selon deux perspectives. Tout d'abord nous étudierons la précision du typage des requêtes. Ensuite, nous nous proposons d'identifier un ensemble de scénarios fréquents d'utilisation de données XML, et d'établir pour chacun d'entre-eux, le langage et le modèle d'évaluation à même d'offrir les meilleures performances. Par exemple, dans un contexte traditionnel de gestion de données persistantes, un stockage fragmenté du document fournit un accès sélectif et donc efficace aux données alors que dans un contexte de traitement de données en mémoire centrale, il convient d'appliquer des techniques basées sur des automates d'arbres sur le document dans son intégralité. Cette étude se concrétisera en l'implantation d'un optimiseur de requêtes générique tant en terme de langages supportés qu'en terme de modèles d'accès et de stockage aux données. Les performances des plans générés par un tel optimiseur seront validées au moyen des bancs de test proposés par le groupe XML-Query du W3C et XMark [SWK+02]. Les sites Gemo et LRI sont impliqués sur cet axe de recherche.

3 Efficacité: traitement à la volée et compression

Responsables: Anca Muscholl, Ioana Manolescu.

Un impératif lorsqu'on travaille au traitement de grandes masses de données est de limiter au maximum l'utilisation des ressources mémoires. Dans ce cadre, deux approches sont intéressantes: (i) compresser les données et (ii) faire de l'évaluation de requêtes à la volée, c'est-à-dire sans avoir à charger entièrement un document en mémoire. On peut même envisager de combiner ces deux approches et travailler à la volée sur des données compressées.

Streaming. L'idée est d'évaluer des requêtes en ne faisant qu'une seule passe sur le document sans qu'il soit nécessaire de stocker celui-ci ou en ne stockant que des informations partielles sur une mémoire de taille limitée. De nombreuses équipes arrivent à évaluer ainsi des fragments de XPath à l'aide d'automates à pile [GMOS03, GS03]. Il est bien évident que toute requête ne peut s'évaluer ainsi et il n'existe à ce jour quasiment pas de travail théorique montrant les limitations de l'approche [SV02]. Nous comptons (i) mettre au point un modèle permettant de mieux cerner

ces limitations, (ii) identifier un langage de requête permettant une évaluation à la volée et (iii) tester sa pertinence sur CDuce.

Streaming et documents compressés. Le format XML est un format textuel qui contient beaucoup de régularités et d'informations redondantes. La compression des documents XML est donc une tâche aisée, qui donne des bons résultats en pratique. Cependant, la tâche devient plus compliquée dès lors qu'on veut pouvoir évaluer les requêtes sans décompresser les documents, ou en ne les décompressant que de manière partielle.

On trouve deux techniques possibles dans ce cas.

Une première approche consiste à séparer la structure d'arbre des documents, des données contenues dans les nœuds, et de ne compresser que ces dernières, sans changer le "squelette" du document. Cette approche a été implantée dans l'outil XMill [LS00], qui présentait toutefois le désavantage de nécessiter la décompression de tout le document pour permettre son interrogation. Des projets plus récents [TH02, MPC03, ABC$^+$03, ABC$^+$04, CN04] ont résolu à des degrés variables ce problème, en choisissant des compromis entre l'expressivité du langage de requêtes supporté sur les données compressées, et le taux de compression. En particulier, dans nos travaux sur le projet XQueC [ABC$^+$03, ABC$^+$04], nous avons privilégié la capacité d'interrogation, utilisant un sous-ensemble important de XQuery; XQueC est un système de gestion de données XML compressées plus adapté à l'interrogation, en échange, son taux de compression est plus faible que dans XMill, par exemple.

Une deuxième approche consiste à compresser la structure d'arbre, en identifiant et partageant des sous-structures similaires ou identiques, dans l'esprit des arbres de décisions binaires (BDD) utilisés en vérification [BGK03, FGK03]. On montre que, dans ce cas, le langage de requête XPath peut s'évaluer avec une décompression limitée sur une structure d'arbre compressée. Cependant cette approche présente le désavantage de devoir compresser les données en fonction d'un ensemble de requêtes fixé à l'avance.

Nous proposons de poursuivre nos travaux sur la compression de documents XML dans deux directions.

Premièrement, permettre l'interrogation des données XML compressées à la volée, afin de profiter du double avantage de réduction de ressources nécessaires pour traitement des requêtes. Il est à noter que tous les travaux précédents sur la gestion des données XML compressées reposent sur le stockage persistent, a priori, des documents, et ne s'intéressent pas au streaming.

Deuxièmement, étudier différentes granularités de compression des données XML, et leur adéquation à l'interrogation de données. En effet, dans le projet XQueC (et autres projets similaires), la compression est appliquée à une granularité très fine, sur chaque valeur (feuille) du document XML; en ce qui concerne le traitement des requêtes structurelles, XQueC fonctionne comme un processeur de requêtes XML sans compression. Nous proposons d'étudier une méthode permettant de choisir la granularité optimale de compression, qui se situe, selon les documents et les requêtes considérées, entre le niveau fin de XQueC, et le niveau de tout le document. Ce choix de granularité peut conduire à des meilleures performances, que les documents soient stockés de manière persistante, ou traités à la volée.

4 Contraintes et Typage de documents

Responsable: Anne-Cécile Caron.

Comme le reflète le processus actuel de standardisation de XML, les documents sont intrinsèquement typés, ceci alors même que, actuellement, la plupart des langages de transformation de documents sont non typés. Dans le cas des langages typés, il est intéressant de noter que les systèmes de types sont basés sur la technologie des automates d'arbres de manière sous-jacente: automates de Glushkov (1-non-ambigus) pour XMλ [MS99], langages réguliers d'arbres avec XDuce, *hedge automata* dans Relaxer [MA00], ...

Dans ce projet, un de nos objectifs est de définir de nouvelles notions de validité pour les documents XML, plus fines que les DTD, avec pour soucis de faciliter les optimisations et de pouvoir typer les requêtes (voir section 2). Nous avons déjà proposé des travaux concernant le typage des requêtes dans les parties précédentes, mais le problème du typage apparaît à ce point central que nous lui réservons une section particulière afin de pouvoir aborder des travaux supplémentaires prévus dans le cadre du projet.

Typage et optimisations. L'intérêt de typer les documents XML est comparable à l'intérêt de typer les programmes (et les données) dans les langages de programmation conventionnels. Plus particulièrement, dans le contexte de notre projet, les informations de types sont une source utile pour l'optimisation des programmes. Ainsi, les informations données par le schéma d'un ensemble de documents, leurs types, peuvent guider le choix d'un bon format de représentation intermédiaire des données. De même, la connaissance du type des données permet d'optimiser les requêtes avant leur exécution: on peut prévoir statiquement quels tests sont inutiles ou bien quelles sous parties des documents ne seront jamais explorées.

Séquences ordonnées et non-ordonnées. Outre des notions de types inspirés par les DTD, ou d'autres langages de schémas existants, nous sommes aussi intéressés par des contraintes d'intégrité satisfaites indépendamment de l'ordre d'apparition des champs dans un document. Ce choix est naturel lorsqu'on travaille avec des données obtenus à partir de la fusion de bases de données relationnelles (un cas important de documents de grande taille), puisque l'ordre des champs est alors sans intérêts. C'est aussi une hypothèse intéressante dans le cadre du langage de requêtes exposé à la section 2. En effet, dans le cas de données « non-ordonnées », on peut espérer réutiliser des techniques d'optimisation développées dans le cadre de langages pour les bases de données relationnelles. Cependant, les classes d'automates d'arbres utilisés dans le cadre de la manipulation de document XML ne sont pas totalement satisfaisantes en présence de ces hypothèses. En effet, il faut pouvoir raisonner sur des arbres non ordonnés et de degrés non bornés. Dans ce projet, parmi d'autres modèles d'automates, nous travaillerons sur une nouvelle classe d'automates d'arbres développée par l'équipe du LIF [LD02, DL03, DLM04] qui permet de traiter ce cas précis.

Contraintes de référence. Si le typage permet de garantir statiquement que les résultats d'une transformation sont bien conformes à ce qui est attendu, ils ne permettent pas de garantir qu'une transformation ne va jamais violer certaines contraintes d'intégrité.

Les contraintes d'intégrité jouent un rôle fondamental dans la conception et la manipulation des bases de données. XML-Schema permet d'ailleurs de définir des contraintes de référence (du type des clefs primaires et clefs étrangères du modèle relationnel). Dans [BFSW01], les auteurs présentent un certain nombre de contraintes liées à la navigation dans un document, c'est à dire des contraintes de référence. Bien sûr, ces contraintes supposent que la donnée soit vue comme un graphe et non plus comme un arbre. Dans un premier temps, il est indispensable d'étudier quelles contraintes il est raisonnable d'ajouter au modèle défini pour les requêtes. L'ajout de contraintes est directement lié au problème évoqué dans les sections précédentes du compromis à faire entre efficacité et expressivité. Nous voulons ensuite étudier la prise en compte de ces contraintes au niveau du langage de requêtes. Il faut d'une part pouvoir vérifier statiquement qu'une transformation donnée d'un document ne viole pas un ensemble de contraintes. D'autre part, utiliser ces contraintes pour optimiser les requêtes. Pour terminer, ces contraintes ne sont pas toujours connues pour un document XML et nous comptons étudier des méthodes d'extraction de contraintes à partir d'une donnée. Le LIFL travaille actuellement sur ces problématiques [ACDY03].

B3 – Résultats attendus:

Une activité de recherche fondamentale est prévue dans chacun des objectifs donnés à l'annexe B2. Les résultats attendus dans le projet TRALALA sont à la fois théoriques et pratiques. Ces résultats s'échelonnent suivant le calendrier suivant. Nous indiquons pour chaque année une liste d'objectifs à remplir, avec les dates prévues pour le début et la fin, et une liste de délivrables.

Première année

Le travail d'implantation sur le langage CDuce joue un rôle central dans les objectifs du projet. En effet, c'est autour de ce logiciel que seront testés et implantés les propositions de chaque équipe. C'est ce constat qui motive un travail de développement pour stabiliser et documenter le langage durant la première année. Les autres résultats attendus consistent essentiellement en un état de l'art sur les problèmes de compression et de streaming, et sur la proposition d'un (et peut-être deux) noyau de langage de requêtes que l'on cherche à greffer à CDuce:

- travail de développement sur le langage CDuce et préparation d'une version distribuable avec sa documentation.

  Début: mois 0 (après embauche d'un Assistant ingénieur) – Fin: mois 8,

- expérimentation: on mènera une étude comparative des langages XDuce, XQuery et CDuce à partir du banc de test proposé par le W3C pour XML-Query et ce selon deux perspectives : précision du typage et performance des requêtes.

  Début: mois 6 – Fin: mois 12,

- définition de modules d'accès en mémoire secondaire pour documents XML, définition d'algorithmes d'optimisation physique.

  Début: mois 0– Fin: mois 12,

- étude du polymorphisme et de l'algèbre de filtrage pour CDuce

- mise au point et formalisation logique du, ou des, noyaux de langages de requêtes qui doivent être greffés à CDuce.

  Début: mois 0 – Fin: mois 12,

- mise au point d'un système de types pour le (ou les) langage de requête envisagé, en prenant en compte les problèmes de compatibilité avec le système de types déjà définis dans CDuce.

  Début: mois 0 – Fin: mois 12,

- état de l'art sur l'évaluation à la volée de documents XML, avec ou sans compression, et définition de nouvelles stratégies d'évaluation dans ce cadre.

  Début: mois 0 – Fin: mois 12,

– [*Sous-task non prioritaire* :] Étude de webservices pour CDuce (le succès dépendra des compétences de l'Assistant Ingénieur).

Délivrables attendus:

- **rapport de recherche /** systèmes de types avec polymorphisme dans CDuce,
- **rapport de recherche /** modules d'accès pour documents XML
- **rapport de recherche /** algèbres de filtrage pour CDuce
- **rapport de recherche /** modèles d'évaluation de requêtes à la volée avec et sans compression,
- **rapport de recherche /** rédaction de la documentation accompagnant la distribution du langage CDuce: manuel de référence et tutorial.
- **prototype /** extension de CDuce avec interfaçage à OCaml et/ou extension des l'algèbre de filtrage pour le filtrage en profondeur (non prioritaire et conjoncturel: extension de CDuce par les *webservices*).

Deuxième année

Pour la deuxième année, nous commencerons à expérimenter les idées proposées afin de vérifier la pertinence de nos hypothèses. C'est aussi l'occasion de tester le comportement de nos outils sur de très grands exemples. Les implantations se feront dans un premier temps sous forme de patches, pour permettre une évaluation pour inclusion dans la distribution officielle du langage CDuce:

- implantation du, ou des, langages de requêtes et des outils de typage.

  Début: mois 12 (après l'embauche du Ingénieur d'Études) – Fin: mois 24,

- définition et implantation d'un optimiseur générique.

  Début: mois 12 – Fin: mois 36,

- expérimentation avec des exemples de grands volumes de données.

  Début: mois 18 – Fin: mois 24,

- étude des aspects liés à la préservation statique de contraintes d'intégrité: il s'agit de proposer des analyses correctes, garantissant la validité d'une transformation respectivement à une contrainte d'intégrité donnée.

  Début: mois 12 – Fin: mois 24,

- mise au point d'algorithmes pour l'évaluation à la volée de documents XML, avec ou sans compression, et caractérisation (logique) de classe de requêtes évaluable avec ces méthodes.

  Début: mois 12 – Fin: mois 30.

Délivrables attendus:

- **rapport de recherche /** proposition de nouvelles méthodes d'évaluation de requêtes à la volée sur des données (partiellement ou non) compressées,
- **logiciel /** version « distribuable » du langage CDuce pour différentes plates-formes, avec utilisation libre à des fins scientifiques (en particulier mise-au-point d'une version Windows).
- **prototype /** extension de CDuce avec des optimisations logiques et/ou basée sur des informations de typage statique.

Troisième année

Pour la dernière année du projet, nous pensons finaliser l'expérimentation des idées théoriques développée pendant la première et deuxième année. Nous nous attendons toujours à travailler sur des aspects théoriques concernant le typage, la compression et l'évaluation à la volée, mais il serait trop audacieux de s'engager sur des points plus précis à cette distance dans le temps: il est plus sérieux d'attendre les résultats des recherches fondamentales menées pendant la deuxième années avant de prévoir des directions de recherches.

L'expérimentation se fera à travers la mise au point d'extension du langage CDuce et nécessitera de mettre au point une batterie de tests sur des grands volume de données. Nous pourrons nous inspirer pour cela des « benchmarks » effectués pendant la première année.

- fin de l'implantation du langage de requête avec ajout d'optimisations qui dépendent de contraintes (statiques) d'intégrité.

  Début: mois 24 – Fin: mois 36,

- implantation des méthodes d'interrogation à la volée de données compressées.

  Début: mois 30 – Fin: mois 36.

Délivrables attendus:

- **prototype /** extension de CDuce avec des analyses et des optimisations dépendant de contraintes d'intégrité,
- **prototype /** extension de CDuce avec des outils d'interrogation à la volée (streaming).
- **prototype /** optimiseur de requêtes générique.

Auto-évaluation du projet

Pour s'assurer du bon déroulement du projet, et vérifier au respect du calendrier que nous proposons, nous prévoyons d'organiser des rencontres d'évaluation entre sites tout les huit mois. Pour nous aider dans notre tâche d'auto-évaluation, et pour accélérer la diffusion de nos travaux, nous prévoyons également des réunions plus importantes, avec une fréquence annuelle, auxquelles sera invité un panel de spécialistes étrangers, sorte de « comité de pilotage », constitué des responsable de chaque thèmes et de chercheurs étrangers avec qui nous collaborons sur des sujets relatifs aux projets. Nous pouvons en particulier compter sur la participation à ce panel de T. Schwentick (Université de Marburg, Allemagne) et V. Vianu (UCSD, USA), qui sont des collaborateurs habituels du projet GEMO et du LIAFA et de Haruo Hosoya, collaborateur de l'équipe Langages du LIENS. Il est déjà prévus que certains des chercheurs de ce panel seront invités à travers les moyens financiers obtenus par l'ACI.

B4 – Summary (in English):

This ACI is motivated by the increasing number of applications that produce, consume or handle large sets of data, or "*datamasses*". In many cases, these are either raw data or a collection of data from various sources, both of which lack uniform descriptive criteria. Such cases require more flexibility than the classical relational model can provide, and have given rise to the so-called semi-structured data model [ABS99], of which XML is one of the most prominent examples.

Our project intends to study the processing, querying and handling of large datamasses whenever data is available in XML format. We pay particular attention to the programming languages and query languages problems. We aim to cover in a uniform way a wide spectrum of different areas, namely: **programming languages** (expressiveness, typing, new programming primitives, query underlying logics, logical optimization), **data access** (streamed data, compression, access to secondary memory storages, persistency engines), **implementation** (pattern matching compiling, physical optimization, subtyping verification, execution models for streamed data).

We will tackle these challenges following three research directions:

query languages: one of the characteristics of the relation model is to base query languages on the relational algebra or the relational calculus. These are paradigms characterized by *high declarativity* (in the sense that they describe the result rather the way to obtain the result) and limited expressiveness (notably, they are not Turing complete). The "simplicity" of these languages is at the origin of the good performances, performances that can be improved by using the algebraic properties of the operators (logical optimization) or by secondary memory management techniques (physical optimization). Our goal is to develop a similar, or at least close, framework for the XML model, and we will pursue it as follows: theoretical study of the expressiveness and complexity of the query languages; definition of query languages for XML and their implementation; definition and validation of optimization techniques.

streaming: the possibility of process streams of data without needing of storing whole documents (if not partially) is crucial in the context of datamasses. We will consider the aspects related to streaming also when the data is compressed. always possible [SV02], so one of the main difficulty to overcome here is to identify a suitable class of "streamable" queries, with or without compression, and in the former case to determine optimal compression granularity.

document typing : type systems are used in the first place for document validation and for checking integrity constraints, but as with standard programming languages, types are at the basis of many helpful optimizations. This makes the study of typing systems one of our primary objectives.

Another motivation for line of work is our interest in integrity constraints whose satisfaction does not depend on the ordering of the fields in a document, unlike the constraints expressible in "classical" type systems for XML such as DTD. This is a natural choice when processing data originating from the fusion of several relational databases (a frequent instance of large documents), since the order of the fields is then irrelevant.

The groups involved in our project have each already been working separately on XML document handling, although this is only one of the incentives for us to work together. Indeed, we share the same fundamental theoretic approach, namely automata theory and the associated logics, and the same interest in query languages and document validation: typing, integrity constraints

Beyond our agreement on foundational tools and our agreement on goals, cooperation inside the project is further strengthened by the choice of a single software target, the CDuce language [BCF03, CDu], a joint development of LIENS and LRI, two of the sites involved in this project.

Références

[ABC+03] Andrei Arion, Angela Bonifati, Gianni Costa, Sandra D'Aguanno, Ioana Manolescu, and Andrea Pugliese. XQueC: Pushing queries to compressed xml data. In *Proceedings of the Int'l Conference on Very Large Database Management (VLDB)*, pages 1065–1068, 2003.

[ABC+04] Andrei Arion, Angela Bonifati, Gianni Costa, Sandra D'Aguanno, Ioana Manolescu, and Andrea Pugliese. Efficient query evaluation over compressed XML data. In *Proceedings of the Int'l Conference on Extending Database Technologies (EDBT)*, pages 200–218, 2004.

[ABCK02] Vincent Aguilera, Frédéric Boiscuvier, Sophie Cluet, and Bruno Koechlin. Pattern tree matching for XML queries. Gemo Technical Report number 211, 2002.

[ABM94] M. Atkinson, V. Benzaken, and D. Maier, editors. *Persistent Object Systems 6*. Workshops in Computing. Springer-Verlag, Tarascon, France, 5-9 September 1994.

[ABS99] S. Abiteboul, P. Buneman, and D. Suciu. *Data on the Web : From Relations to Semistructured Data and XML*. Morgan Kaufmann, 1999.

[ACDY03] A.-C.Caron, D.Debarbieux, and Y.Roos. Modéles de données semi-structurées et contraintes d'inclusion. In *Extraction et Gestion de Connaissances*, volume 17, pages 461–472. Hermès, 2003.

[AHV95] S. Abiteboul, R. Hull, and V. Vianu. *Foundations of Databases*. Addison-Wesley, 1995.

[BCF03] V. Benzaken, G. Castagna, and A. Frisch. CDuce: an XML-centric general-purpose language. In *ICFP '03, 8th ACM International Conference on Functional Programming*, pages 51–63, Uppsala, Sweden, 2003. ACM Press.

[BDH92] V. Benzaken, C. Delobel, and G. Harrus. Clustering strategies in o_2: an overview. In F. Bancilhon, C. Delobel, and P. Kanellakis, editors, *Building an Object-Oriented Database System: the Story of O_2*, pages 385–410. Morgan Kaufman, 1992.

[BDK92] F. Bancilhon, C. Delobel, and P. Kanellakis. *Building an object-oriented database system: the story ofO_2*. Morgan Kaufmann, 1992.

[BFG01] Robert Baumgartner, Sergio Flesca, and Georg Gottlob. Visual web information extraction with lixto. In *The VLDB Journal*, pages 119–128, 2001.

[BFSW01] Peter Buneman, Wenfei Fan, Jérôme Siméon, and Scott Weinstein. Constraints for semistructured data and XML. *SIGMOD Record (ACM Special Interest Group on Management of Data)*, 30(1):47–54, 2001.

[BGK03] Peter Buneman, Martin Grohe, and Christoph Koch. Path queryes on compressed XML. In *International Conference on Very Large Databases (VLDB)*, 2003.

[BL98] Tim Berners-Lee. Semantic web road map. `http://www.w3.org/DesignIssues/Semantic.html`, 1998.

[BMS03] Gavin Bierman, Erik Meijer, and Wolfram Schulte. Programming with rectangles, triangles, and circles. In *Proc. of XML 2003*, 2003.

[BS02] Alexandru Berlea and Helmut Seidl. Binary queries. In *Proceedings of Extreme Markup Languages*, 2002.

[BT04] I. Boneva and J.-M. Talbot. When Ambients Cannot be Opened. *Theoretical Computer Science*, 2004.

[BY98] N. Bidoit and M. Ykhlef. Fixpoint calculus for querying semistructured data. In *Int. Workshop on World Wide Web and Databases (WebDB)*, 1998.

[CDJ+99] H. Comon, M. Dauchet, F. Jacquemard, S. Tison D. Lugiez, and M. Tommasi. *Tree Automata and their application*. To appear as a book, 1999.

[CDu] CDuce: A modern programming language, adapted to the manipulation of XML documents. `http://www.cduce.org`.

[CFMR03] Don Chamberlin, Peter Fankhauser, Massimo Marchiori, and Jonathan Robie. Xml query (xquery) requirements. Technical Report 20030627, World Wide Web Consortium, 2003.

[CFR00] Don Chamberlin, Daniela Florescu, and Jonathan Robie. Quilt: an XML query language for heterogeneous data sources. In *International Workshop on the Web and Databases (WebDB)*, 2000.

[CG01] Luca Cardelli and Giorgio Ghelli. A query language based on the Ambient logic. In *Proc. of ESOP'01*, volume 2028 of *LNCS*, pages 1–22. Springer-Verlag, 2001.

[CGA+02] G. Conforti, G. Ghelli, A. Albano, D. Colazzo, P. Manghi, and C. Sartiani. The Query Language TQL. In *Proc. of 5th International Workshop on Web and Databases (WebDB 2002)*, 2002.

[CM01] James Clark and Murata Makoto, editors. *RELAX-NG Tutorial*. OASIS, 2001.

[CN04] James Cheng and Wilfred Ng. XQzip: Querying compressed XML using structural indexing. In *International Conference on Extending Database Technologies (EDBT)*, 2004.

[CNT04] Julien Carme, Joachim Niehren, and Marc Tommasi. Querying unranked trees with stepwise tree automata, January 2004.

[DFF+98] A. Deutsch, M. Fernández, D. Florescu, A. Levy, and D. Suciu. Xml-ql: A query language for XML, 1998. `http://www.w3.org/TR/NOTE-xml-ql`.

[DL03] Silvano Dal Zilio and Denis Lugiez. XML schema, tree logic and sheaves automata. In *Rewriting Techniques and Applications (RTA)*, 2003.

[DLM04] Silvano Dal Zilio, Denis Lugiez, and Charles Meyssonnier. A logic you can count on. In *POPL 2004 – 31st Annual ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages*, 2004.

[DOM] Document Object Model (dom). `http://www.w3.org/DOM/`.

[FCB02] Alain Frisch, Giuseppe Castagna, and Véronique Benzaken. Semantic subtyping. In *IEEE Symposium on Logic In Computer Science (LICS)*, 2002.

[FGK03] Markus Frick, Martin Grohe, and Christoph Koch. Query evaluation on compressed trees. In *Symposium on Logics in Computer Science (LICS)*, 2003.

[FK99] D. Florescu and D. Kossmann. Storing and querying XML data using an RDMBS. In *IEEE Data Engineering Bulletin*, 1999.

[FS03] Irini Fundulaki and Arnaud Sahuguet. A language-based approach for distributed user profile data management. Lucent Technical Report, 2003.

[FSW+99] M. Fernández, J. Siméon, P. Wadler, S. Cluet, A. Deutsch, D. Florescu, A. Levy, D. Maier, J. McHugh, J. Robie, D. Suciu, and J. Widom. Xml query languages: Experiences and exemplars. 1999.

[FSW00a] Mary Fernández, Jérôme Siméon, and Philip Wadler. An algebra for XML query. In *Foundations of Software Technology and Theoretical Computer Science*, 2000.

[FSW00b] Mary Fernández, Jérôme Siméon, and Philip Wadler. An algebra for XML query. `http://www.cs.bell-labs.com/~wadler/topics/xml.html`, 2000. Submitted as an input to the W3C XML Query working group.

[GIS] Open GIS Consortium. `http://www.opengis.org/index.htm`.

[GK02a] G. Gottlob and C. Koch. Monadic datalog and the expressive power of languages for web information extraction. In *Symposium on Principles of Database Systems (PODS)*, pages 17–28, 2002.

[GK02b] G. Gottlob and C. Koch. Monadic queries over tree-structured data. In *Proceedings of the* 17^{th} *LICS*, Lecture Notes in Computer Science, Copenhagen, 2002. Springer Verlag.

[GMOS03] Todd J. Green, Gerome Miklau, Makoto Onizuka, and Dan Suciu. Processing XML streams with deterministic automata. In *International Conference on Database Theory (ICDT)*, 2003.

[GO] Gene Ontology Consortium. `http://www.geneontology.org/`.

[GP03] V. Gapayev and B. Pierce. Regular object types. In *Proc. of ECOOP '03*, Lecture Notes in Computer Science. Springer, 2003.

[GRI] ACI globalisation des ressources informatiques et des données (GRID). http://www.recherche.gouv.fr/recherche/aci/grid.htm.

[GS03] Ashish Gupta and Dan Suciu. Stream processing of XPath queries with predicates. In *ACM SIGMOD Conference on Management of Data*, 2003.

[HVJT01] D. Srivastava H. V. Jagadish, L. V. S. Lakshmanan and K. Thompson. TAX: A tree algebra for XML. In *Proc. of DBPL*, 2001.

[JAXa] Java API for XML Binding (jaxb). http://java.sun.com/xml/jaxb/.

[JAXb] Java API for XML Processing (jaxp). http://java.sun.com/xml/jaxp/.

[JMTT00] Philippe Devienne Jean-Marc Talbot and Sophie Tison. Generalized definite set constraints. *Constraints, an International Journal*, 5:25–39, 2000.

[KMS] Nils Klarlund, Anders Møller, and Michael I. Schwartzbach. DSD: A schema language for XML. http://www.brics.dk/DSD/.

[Kwe01] The Kweelt XML querying platform. http://kweelt.sourceforge.net, 2001.

[LD02] Denis Lugiez and Silvano Dal Zilio. Multitrees automata, Presburger's constraints and tree logics. Technical Report 08-2002, LIF, June 2002.

[Lis] Kirill Lisovsky. Semistructured data and the scheme language. http://www196.pair.com/lisovsky/sxml/.

[LS00] Harmut Liefke and Dan Suciu. XMill: an efficient compressor for xml data. In *ACM SIGMOD Conference on Management of Data*, 2000.

[MA00] Makoto Murata and Tomoharu Asami. Relaxer: Java classes from relax modules. In *Extreme Markup Languages*, 2000.

[MPC03] J. Ki Min, M. Park, and C. Chung. XPRESS: A queriable compression for XML data. In *International ACM Conference on the Management of Data*, 2003.

[MS99] Erik Meijer and Mark Shields. XMλ: A functional programming language for constructing and manipulating XML documents. Draft, 1999.

[MSS03] Anca Muscholl, Thomas Schwentick, and Helmut Seidl. Numerical document queries. In *Principles of Database Systems (PODS)*, 2003.

[Mur01] Makoto Murata. Extended path expression for XML. In *Proc. of Symposium on Principles of Database Systems (PODS)*. ACM Press, 2001.

[NS98] Andreas Neumann and Helmut Seidl. Locating matches of tree patterns in forests. In *Foundations of Software Technology and Theoretical Computer Science*, pages 134–145, 1998.

[NS02] Frank Neven and Thomas Schwentick. Query automata over finite trees. *Theoretical Computer Science*, 275(1-2):633–674, 2002.

[Seg03] Luc Segoufin. Typing and querying xml documents: some complexity bounds. In *Principle of Databases Systems (PODS)*, 2003.

[SGT+99] J. Shanmugasundaram, H. Gang, K. Tufte, C. Zhang, D. DeWitt, and J. Naughton. Relational databases for querying XML documents: Limitations and opportunities. In *Int'l. Conference on Very Large Databases (VLDB)*, 1999.

[SV02] Luc Ségoufin and Victor Vianu. Validating streaming XML documents. In *Symposium on Principles of Database Systems (PODS)*, 2002.

[SWK+02] Albrecht Schmidt, Florian Waas, Martin L. Kersten, Michael J. Carey, Ioana Manolescu, and Ralph Busse. Xmark: A benchmark for xml data management. In *Proceedings of the Int'l. Conference on Very Large Database Management (VLDB)*, pages 974–985, 2002.

[TH02] Pankaj Tolani and Jayant Haritsa. XGRIND: A query-friendly XML compressor. In *International Conference on Data Engineering (ICDE)*, 2002.

[Via01] V. Vianu. A web odissey: from Codd to XML. In *Proc. of International Conference on Principles of Database Systems (PODS '01)*, pages 1–15. ACM Press, 2001.

[WCBF00] F. Wattez, S. Cluet, V. Benzaken, and C. Fiegel. Benchmarking queries over trees: Learning the hard truth the hard way. In *ACM SIGMOD International Conference*, Dallas, May 2000. ACM.

[WR99] Malcolm Wallace and Colin Runciman. Haskell and XML: Generic combinators or type-based translation? In *International Conference on Functional Programming (ICFP)*, 1999.

[XDu] XDuce: A typed XML processing language. `http://xduce.sourceforge.net/`.

[XML98] Extensible markup language (XML™), 1998. XML 1.0, W3C Recommendation, `http://www.w3.org/XML/`.

[XPa] XML Path Language (XPath). `http://www.w3.org/TR/xpath`.

[XQL99] XQL: (XML query language). http://www.ibiblio.org/xql/xql-proposal.html, 1999.

[XQu01] XQuery: An XML query language. `http://www.w3.org/TR/xquery/`, 2001. W3C Working Draft.

[XS00] XML Schema Part 0: Primer, W3C Working Draft. `http://www.w3.org/TR/xmlschema-0/`, 2000.

[XSL99] XSL Transformations (XSLT), 1999. `http://www.w3.org/TR/xslt`.

C – MOYENS FINANCIERS ET HUMAINS DEMANDEÉS PAR CHAQUE ÉQUIPE [6]

C0 - Critères pour la détermination des moyens financiers

Nous avons utilisé des critères uniformes à toutes les équipes participantes au projet pour calculer les besoins financiers pour la mise en oeuvre de la recherche. Ainsi,

CDD. Le niveau de rémunération des agents sur CDD est fixé en fonction du diplôme de la personne concernée et de son niveau d'emploi. Pour cela nous avons utilisé la rémunération d'un agent titulaire CNRS débutant, toutes primes comprises, selon le barème publié par le CNRS pour l'évaluation des coûts complets de contrats de recherche. Nous avons fait référence aux coûts salariaux annuels 2003 des personnels CNRS réévalués de 2%. Ce qui donne les barèmes suivants:

1. **Postdoc:** € 45792, correspondant au coût annuel d'un Chargé de Recherche 2ème classe, 1er échelon.
2. **Ingénieur d'études:** € 40550, correspondant au coût annuel d'un Ingénieur d'Études 2ème classe, 1er échelon
3. **Assistant Ingénieur:** € 36587, correspondant au coût annuel d'un Assistant Ingénieur 1er échelon
4. **Stagiaire:** € 2426/mois, correspondant au coût mensuel d'un Adjoint Technique de la Recherche 1er échelon.

Matériel. Les dépenses affectées au matériel sont calculées singulièrement par chaque participant.

Missions. Pour les missions nous avons calculé un coût moyen de € 800 pour les mission en France et € 1200 pour celles à l'étranger. Pour les invitations de chercheurs étrangers nous avons calculé un coût moyen de € 800 par sémaine.

Gestion et consommables. Les frais de gestion sont calculées de manière forfaitaires pour un montant du 15% des frais de fonctionnement. Le frais pour consommables sont calculées de manière forfetaire à € 1000 par an. Le choix de démander des frais de géstion et d'inclure ou pas dans ces dernières les frais pour consommables rélève de chaque équipe.

6. Une fiche C doit être remplie pour chaque laboratoire ou équipe partenaire.

Action Concertée Incitative

MASSES DE DONNÉES

Descriptif complet du projet

C1 - Demandes effectuées dans le cadre de l'ACI pour le présent projet :

Nom de l'équipe ou du laboratoire : Projet GEMO & LIAFA

Moyens demandés dans le cadre de la présente ACI (en K euros TTC) :

Financements via le Fonds National de la Science :

	2004 (09.04-08.05)	2005 (09.05-08.06)	2006 (09.06-08.07)	Total
Équipement	3	0	0	3
Fonctionnement (dont CDD décrits ci-dessous)	60,8 (45,8 CDD)	15	15	90,8 (45,8 CDD)
Total / année	63,8	15	15	93,8

Dépenses de personnels (CDD) [7] :

Nature de l'emploi (post-doc, ingénieur, assistant-ingénieur,. . .)	Post-Doc
Durée de l'emploi (en mois) [8]	12 mois (mois 1-12 du projet)
Coût total de l'emploi	45,8 K€

Financements via les organismes de recherche :

	2004–2005	2005–2006	2007–2007	Total
Nombre de post-docs (préciser pour chaque demande la durée en mois) [9]		1 x 12 mois		1 an
Nombre d'accueils de chercheurs étrangers (préciser pour chaque demande la durée en mois)	1 x 1 mois + 1 x 2 semaines	1 x 1 mois + 1 x 2 semaines	1 x 1 mois + 1 x 2 semaines	4,5 mois
Nombre d'acceuils en délégations ou détachements [10]				0

7. Un tableau doit être rempli pour chaque demande de CDD.
8. Doit être inférieure à 24 mois.
9. Sauf demande argumentée, la durée d'un contrat de type post-doc ne pourra excéder 12 mois.
10. Certaines des demandes déjà faites pour 2004-2005 pourront être attribuées au titre de l'ACI.

Action Concertée Incitative

MASSES DE DONNÉES

Descriptif complet du projet

Allocations de recherche :

	2004–2005	2005–2006	2006–2007	Total
Nombre d'allocations de recherche débutant en :				

Justifications scientifiques de l'ensemble des demandes :

L'objectif des chercheurs de ce site est d'identifier, parmi les diverses propositions existantes de langages de requêtes pour XML, un noyau central qui soit à la foi simple, générique (indépendant de l'application), facile à évaluer, et dont les avantages et les limitations soient maîtrisées. Le but est d'obtenir une implantation efficace sur CDuce et d'expérimenter ce noyau dans le contexte spécifique de notre étude: évaluation de requêtes à la volée (streaming) et évaluation de requêtes sur des documents compressés.

Ce travail demande d'établir un état de l'art complet, ce qui nécessite la participation à divers colloques et conférences spécialisées dans le domaine. À ce titre, les frais de fonctionnement comportent 15 K€ par an de frais de missions, qui doivent couvrir la participation à ces colloques, en plus des frais de déplacements dans les autres sites du projet pour les réunions d'avancement semestrielles. La demande de financement contient également des demandes d'accueil de chercheurs étrangers. Ces séjours sont destinés à Michael Benedikt (Bell-labs, USA) et Victor Vianu (UCSD,USA):

Michael Benedikt, du laboratoire Bell-labs de Lucent, est un expert en langage de requêtes et XML. Il a une grande expérience sur la théorie des automates et a déjà travaillé sur l'évaluation de requêtes à la volée. Il a collaboré avec Luc Segoufin à de nombreuses reprises.

Victor Vianu, est un spécialiste des automates et de XML. Il a publié de nombreux résultats sur le typage et XML. Il a collaboré avec Luc Segoufin à de nombreuses reprises.

Pour la deuxième année du projet, on prévoit la participation d'un chercheur en post-doc pour travailler sur les aspects compression et évaluation à la volée. Les liens de coopération avec Georg Gottlob (TU Vienne), à travers le projet GAMES commun avec Anca Muscholl, offre la possibilité de recruter un chercheur expérimenté dans les domaines XML.

Les dépenses d'équipement couvrent l'achat d'un PC (3 K€) qui permettra de fournir une machine aux stagiaires et au doctorant qui travailleront sur le projet.

C2 - Autres soutiens financiers apportés au projet :

- Luc Segoufin est responsable d'un PAI projet européen Procope (collaboration franco-allemande) avec Thomas Schwentick, Université de Marburg, Allemagne.
- Anca Muscholl est la coordinatrice française du projet européen "Research and Training Network" GAMES: (durée 4 ans, début 2002), portant sur les applications des jeux en vérification, coordonné par Erich Grädel (Aix-la-Chapelle).

Action Concertée Incitative

MASSES DE DONNÉES

Descriptif complet du projet

Nom de l'équipe ou du laboratoire : LIENS

Moyens demandés dans le cadre de la présente ACI (en K euros TTC) :

Financements via le Fonds National de la Science :

	2004 (09.04-08.05)	2005 (09.05-08.06)	2006 (09.06-08.07)	Total
Équipement		4		4
Fonctionnement (dont CDD décrits ci-dessous)	49 (36,6 CDD)	30,8 (20,3 CDD)	55,8 (40,6 CDD)	135,6 (97,5 CDD)
Total / année	49	34,8	55,8	139,6

Dépenses de personnels (CDD) [11] :

Nature de l'emploi (post-doc, ingénieur, assistant-ingénieur,. . .)	Ingénieur d'études
Durée de l'emploi (en mois) [12]	18 mois (mois 18-36 du projet)
Coût total de l'emploi	60,9K€

Nature de l'emploi (post-doc, ingénieur, assistant-ingénieur,. . .)	Assistant Ingénieur
Durée de l'emploi (en mois) [13]	12 mois (mois 1-12 du projet)
Coût total de l'emploi	36,6K€

Financements via les organismes de recherche :

	2004–2005	2005–2006	2006–2007	Total
Nombre de post-docs (préciser pour chaque demande la durée en mois) [14]		1 x 12 mois		12 mois
Nombre d'accueils de chercheurs étrangers (préciser pour chaque demande la durée en mois)				
Nombre d'acceuils en délégations ou détachements [15]				

12. Un tableau doit être rempli pour chaque demande de CDD.
13. Doit être inférieure à 24 mois.
14. Sauf demande argumentée, la durée d'un contrat de type post-doc ne pourra excéder 12 mois.
15. Certaines des demandes déjà faites pour 2004-2005 pourront être attribuées au titre de l'ACI.

Action Concertée Incitative

MASSES DE DONNÉES

Descriptif complet du projet

Allocations de recherche :

	2004–2005	2005–2006	2006–2007	Total
Nombre d'allocations de recherche débutant en :				

Justifications scientifiques de l'ensemble des demandes :

Le travail d'implantation de l'équipe « Langages » joue un rôle central dans les objectifs du projet. En effet, c'est autour du logiciel développé par cette équipe (en partenariat avec le LRI) que seront testés et implantés les propositions de chaque équipe. C'est ce constat qui motive la demande de deux postes d'ingénieur de recherches. Il faut noter que c'est la seule équipe de ce projet à demander des poste d'ingénieur de recherche, la personne occupant le second poste étant amené à travailler en commun avec chaque équipe.

- Le premier poste d'ingénieur se tiendra dans les huit premiers mois de l'an 1. Son but est l'implantation finale des algorithmes qui sont en train d'être développé dans l'équipe et la préparation d'un manuel de référence, d'un tutorial et d'une description technique de l'implantation de CDuce. Ces manuels (prévus pour le mois 10) sont important pour les membres du projet car ils constitueront la base essejntielle de la documentation. Cet ingénieur est également essentiel à la préparation d'une version distribuable de CDuce (packaging, binaires, portage)
- Le deuxième poste d'ingénieur, qui doit commencer a moitié de l'an 2, est destiné à l'implantation des idées issues du projet. Il doit aussi assurer une assistance technique sur la plateforme CDuce auprès des autres membres du projet. L'implantation se fera dans un premier temps sous forme de patches, pour permettre une évaluation pour inclusion dans la distribution officielle. Enfin cet ingénieur assurera la maintenance de la distribution CDuce et des outils de développement.

Le poste de post-doc est démandé pour compenser le probable départ d'Alain Frisch de l'équipe.

Les frais de fonctionnement, hors frais de géstion et consommables, sont de 6 K€ les deux premières années et de 6 K€ la dernière année. Ces montants correspondent à des frais de missions pour un montant de 2 K€ la première et deuxième année et 6 K€ la dernière année (cette croissance étant due à l'arrivée au mois 18 du poste d'IE). Elles se justifient par le fait que l'ingénieur sera au service de toutes les équipes du projet, ce qui implique des séjours de longue durée sur les autres sites, notamment Marseille et Lille. Le restant 4K€ de chaque année servent à financer une visite de 5 semaines par an, qui sera effectuée par Benjamin Pierce ou Haruo Hosoya, deux des membres étrangers associés au projet.

La dépense d'équipement, 4 K€ pour l'année 2, doit servir à l'achat d'une station de travail avec une importante quantité de mémoire (au moins 2 Gb de RAM). Cet achat doit permettre de tester les logiciels sur des exemples de grande taille. En effet, les algorithmes à base d'automates d'arbres utilisé dans ce projet peuvent être parfois très gourmant en mémoire.

Aux frais de fonctionnement sont ajoutés des frais de gestion, calculées comme 15% du total, qui comprennent les frais de consommables (1 K€ par an), et qui correspondent a 6,4 K€ pour l'année 1, 4,5 K€ pour l'année 2 et 7,3 K€ pour la 3ème.

C2 - Autres soutiens financiers apportés au projet :

- L'équipe Langages du LIENS, en collaboration avec l'équipe BD du LRI, participe à un projet RNTL de 18 mois - GraphDuce, démarré en décembre 2003, avec la société BrixLogic, visant au développement d'interfaces graphiques pour CDuce dans la cadre d'applications financières.
- L'équipe Langages du LIENS et l'équipe BD du LRI sont membres du consortium du projet CASC, ACI Sécurité Informatique 2003.
- L'équipe Langages du LIENS participe au projet Européen "MyThS" qui doit se terminer en décembre 2004.

Action Concertée Incitative

MASSES DE DONNÉES

Descriptif complet du projet

Nom de l'équipe ou du laboratoire : LIF

Moyens demandés dans le cadre de la présente ACI (en K euros TTC) :

Financements via le Fonds National de la Science :

	2004 (09.04-08.05)	2005 (09.05-08.06)	2006 (09.06-08.07)	Total
Équipement	3	3		6
Fonctionnement (dont CDD décrits ci-dessous)	15,3	15,3	15,3	45,9
Total / année	18,3	18,3	15,3	51,9

Dépenses de personnels (CDD) [16]:

Nature de l'emploi (post-doc, ingénieur, assistant-ingénieur,. . .)	
Durée de l'emploi (en mois) [17]	
Coût total de l'emploi	

Financements via les organismes de recherche :

	2004–2005	2005–2006	2006–2007	Total
Nombre de post-docs (préciser pour chaque demande la durée en mois) [18]				
Nombre d'accueils de chercheurs étrangers (préciser pour chaque demande la durée en mois)		1 mois	1 mois	2 mois
Nombre d'acceuils en délégations ou détachements [19]				

16. Un tableau doit être rempli pour chaque demande de CDD.
17. Doit être inférieure à 24 mois.
18. Sauf demande argumentée, la durée d'un contrat de type post-doc ne pourra excéder 12 mois.
19. Certaines des demandes déjà faites pour 2004-2005 pourront être attribuées au titre de l'ACI.

Action Concertée Incitative

MASSES DE DONNÉES

Descriptif complet du projet

Allocations de recherche :

	2004–2005	2005–2006	2006–2007	Total
Nombre d'allocations de recherche débutant en :		1		1

Justifications scientifiques de l'ensemble des demandes :

Le travail de l'équipe Move, dans ce projet, tourne essentiellement autour du développement de modèles d'automates d'arbres pour le typage et la validation de documents XML. Ces outils fondamentaux doivent servir de bases au système de types utilisé dans le noyau de langage de requêtes que nous proposons d'ajouter à CDuce. Le passage d'un modèle purement théorique à un modèle implantable soulève de nombreux problèmes scientifiques. La demande de financement contient également des demandes d'accueil de chercheurs étrangers (4 mois, 2 démandes au FNS et 2 aux organismes de recherche), qui sont destinés à accueillir un chercheur de l'équipe travaillant sur TQL, à l'université de Pise ou l'un des membres du panel de spécialistes que nous allons mettre sur pied pour participer à l'auto-évaluation de notre projet. Nous ne connaissons pas encore à ce jour la liste exacte des chercheurs qui seront invités.

Les frais de fonctionnement sont de 15,3 K€ par an et correspondent en grande partie à des frais de missions pour les participants de l'équipe (auquel s'ajoutera le doctorant à partir de l'an 2) et en frais d'invitation pour les chercheurs étranges. Ce montant annuel à été calculé comme il suit: 3 missions en France (2,4 K€), 2 mission à l'étranger (2 K€), 2 mois d'invitation pour chercheurs étrangers (8 K€), des frais de gestions calculées comme 15% de la somme de montants précédents (1,9 K€), et des frais pour consommables (1 K€). Notre demande couvre donc une moyenne de 6 missions à l'étranger (présentation dans des conférences, participation à des écoles) et de 9 missions en France par an, qui doivent permettre de couvrir l'organisation et la participation aux réunions pluriannuelles du projet. De plus notre doctorant sera amené à se déplacer régulièrement à Paris pour se mettre en relation avec les équipes qui implantent le langage CDuce.

Les dépenses d'équipement couvrent l'achat d'un PC (3 K€) pendant 2 ans et permettront de fournir une machine aux stagiaires et au doctorant qui travailleront sur le projet.

C2 - Autres soutiens financiers apportés au projet :

- Silvano Dal Zilio est responsable du projet ATIP 2002 jeunes chercheurs "Fondements de l'interrogation des données semi-structurées", auquel participe Denis Lugiez et Charles Meyssonnier.
- Silvano Dal Zilio est membre du projet européen FET-IST "Global Computing" Profundis, dans lequel il contribue à des recherches sur les logiques modales d'arbres. Ces travaux sont basé sur le lien existant entre logiques spatiales et langages de requêtes pour les données semi-structurées, illustré par exemple dans les travaux de L. Cardelli et G. Ghelli sur le langage TQL.

Action Concertée Incitative

MASSES DE DONNÉES

Descriptif complet du projet

Nom de l'équipe ou du laboratoire : MOSTRARE - LIFL

Moyens demandés dans le cadre de la présente ACI (en K euros TTC) :

Financements via le Fonds National de la Science :

	2004 (09.04-08.05)	2005 (09.05-08.06)	2006 (09.06-08.07)	Total
Équipement	10	3	3	16
Fonctionnement (dont CDD décrits ci-dessous)	8	15,2 (7,2 CDD)	15,2 (7,2 CDD)	38,4 (14,4 CDD)
Total / année	18	18,2	18,2	54,4

Dépenses de personnels (CDD) [20] :

Nature de l'emploi (post-doc, ingénieur, assistant-ingénieur,...)	Stagiaire
Durée de l'emploi (en mois) [21]	3 mois (an 2)
Coût total de l'emploi	7,2 K€

Nature de l'emploi (post-doc, ingénieur, assistant-ingénieur,...)	Stagiaire
Durée de l'emploi (en mois)	3 mois (an 3)
Coût total de l'emploi	7,2 K€

Financements via les organismes de recherche :

	2004–2005	2005–2006	2006–2007	Total
Nombre de post-docs (préciser pour chaque demande la durée en mois) [22]				
Nombre d'accueils de chercheurs étrangers (préciser pour chaque demande la durée en mois)				
Nombre d'accueils en délégations ou détachements [23]				

20. Un tableau doit être rempli pour chaque demande de CDD.
21. Doit être inférieure à 24 mois.
22. Sauf demande argumentée, la durée d'un contrat de type post-doc ne pourra excéder 12 mois.
23. Certaines des demandes déjà faites pour 2003-2004 pourront être attribuées au titre de l'ACI.

Action Concertée Incitative

MASSES DE DONNÉES

Descriptif complet du projet

Allocations de recherche :

	2004–2005	2005–2006	2006–2007	Total
Nombre d'allocations de recherche débutant en :				

Justifications scientifiques de l'ensemble des demandes :

Les frais de fonctionnement permettent de prendre en charge les déplacements à hauteur de 8 K€ par an : visite des autres sites, réunions semestrielles, conférences.

Nous développons en ce moment des outils pour l'inférence de contraintes de chemins et leur prise en compte pour l'optimisation de requêtes. Nous voulons accentuer ces développements et étudier leur intégration à CDuce. C'est pourquoi les frais de fonctionnement prévoient également 6 K€ par an pour payer des stagiaires, à partir de la deuxième année. Il s'agit de postes en CDD d'une durée de trois mois destinés à être occupés par des étudiants titulaires d'une maîtrise (comme emploi d'été, en dehors des périodes de stages classiques) ou par des étudiants d'IUP ou de DESS en stage pendant la période universitaire (avec indemnités de stages.)

Les dépenses d'équipement couvrent l'achat de deux PC et d'un portable la première année et d'un PC la deuxième et troisième année. Le portable sera utilisé pour les déplacements sur les autres sites.

C2 - Autres soutiens financiers apportés au projet :

Néant

Action Concertée Incitative

MASSES DE DONNÉES

Descriptif complet du projet

Nom de l'équipe ou du laboratoire : LRI, UMR 8623, équipe Bases de Données

Moyens demandés dans le cadre de la présente ACI (en K euros TTC) :

Financements via le Fonds National de la Science :

	2004 (09.04-08.05)	2005 (09.05-08.06)	2006 (09.06-08.07)	Total
Équipement	4	6,5	3	13,5
Fonctionnement (dont CDD décrits ci-dessous)	20 (4,8 CDD)	50,6 (27,7 CDD)	46,6 (27,7 CDD)	117,2 (60,2 CDD)
Total / année	24	57,1	49,6	130,7

Dépenses de personnels (CDD) [24] :

Nature de l'emploi (post-doc, ingénieur, assistant-ingénieur,. . .)	Post-doc
Durée de l'emploi (en mois) [25]	12 mois (mois 18-30 du projet)
Coût total de l'emploi	45,8 K€

Nature de l'emploi (post-doc, ingénieur, assistant-ingénieur,. . .)	Stagiaire
Durée de l'emploi (en mois)	2 mois (an 1)
Coût total de l'emploi	4,8 K€

Nature de l'emploi (post-doc, ingénieur, assistant-ingénieur,. . .)	Stagiaire
Durée de l'emploi (en mois)	2 mois (an 2)
Coût total de l'emploi	4,8 K€

Nature de l'emploi (post-doc, ingénieur, assistant-ingénieur,. . .)	Stagiaire
Durée de l'emploi (en mois)	2 mois (an 3)
Coût total de l'emploi	4,8 K€

Financements via les organismes de recherche :

	2004–2005	2005–2006	2006–2007	Total
Nombre de post-docs (préciser pour chaque demande la durée en mois) [26]				
Nombre d'accueils de chercheurs étrangers (préciser pour chaque demande la durée en mois)				
Nombre d'acceuils en délégations ou détachements [27]		1		1

24. Un tableau doit être rempli pour chaque demande de CDD.
25. Doit être inférieure à 24 mois.

Action Concertée Incitative

MASSES DE DONNÉES

Descriptif complet du projet

Allocations de recherche :

	2004–2005	2005–2006	2006–2007	Total
Nombre d'allocations de recherche débutant en :	1			1

Justifications scientifiques de l'ensemble des demandes :

Le LRI, à travers la coopération étroite de Véronique Benzaken avec le site du LIENS, est partie prenante dans le développement du langage CDuce. Alors que le site du LIENS s'occupe plus particulièrement des aspects typage et langage de transformation de documents, le LRI est chargé du travail d'implantation autour d'un noyau « langage de requête » qui doit être greffé à CDuce. Ce travail est mené par des membres de l'équipe, mais les développements prévus, à la fois théorique et pratique, nécessitent le recrutement pour un an d'un jeune chercheur confirmé.

En particulier, une fois que la partie requête, avec ses aspects optimisations, sera intégrée à CDuce (intégration prévue pour le mois 18 du projet), il conviendra d'étudier des aspects liés à la préservation statique de contraintes d'intégrité. Les six premiers mois du travail de post-doc seront consacrés à l'étude théorique et consistera à proposer des analyses correctes garantissant la validité d'une transformation respectivement à une contrainte d'intégrité. Les six mois restant seront consacrés à l'intégration de ces outils d'analyse au sein de CDuce. Les travaux portant sur l'optimisation physique de requêtes et les mesures à grande echelle requièrent une forte activité d'implantation. C'est pourquoi les frais de fonctionnement prévoient également 4,8 K€ par an pour payer des stagiaires, à partir de la première année. Il s'agit de postes en CDD d'une durée de deux mois destinés à être occupés par des étudiants titulaires d'un master (comme emploi d'été, en dehors des périodes de stages classiques) ou par des étudiants en parcours d'IUP-Miage ou de Master d'informatique en stage pendant la période universitaire (avec indemnités de stages.)

Notre demande d'allocation de recherche permettrait de recruter un doctorant qui travaillerait a la définition et l'implantation d'un optimiseur générique pour données XML. Enfin, compte tenu du lourd travail tant dans le domaine de la définition du langage de requête, de son optimisation, de la définition d'un optimiseur générique que de l'étude de la préservation statique de contraintes d'intégrité dans le cadre de CDuce, un acceuil au CNRS en délégation ou en détachement a été demandé.

Les frais d'équipement servent à couvrir pour la première année l'achat d'un video-projecteur qui servira à la présentation des travaux de l'équipe (4 K€) ; d'un ordinateur portable (3,5 K€) et d'une station de travail (3 K€) la deuxième année. La première station de travail sera utilisée pour l'implantation de la partie requête pure dans CDuce. L'autre machine sera utilisée pour l'expérimentation à grande échelle qui nécessite le couplage du langage à un gestionnaire de données XML natif et/ou un système de facture plus classique. Une fois déduit le coût du post-doc, dont l'emploi est à cheval sur les années 2 et 3, les frais de fonctionnement se résument à des frais de mission, la première année (7,6 K€) couvrant cinq missions en France prévues pour participer aux réunion de travail dans le cadre de l'ACI et trois missions à l'étranger. Les seconde et troisième années, les missions s'élèvent à (8,8 k€) par an couvrant cinq missions en France et quatre missions à l'étranger dont une servira à prendre en charge la venue et le retour du post-doctorant.

Les frais pour consommables sont calculés a 1 K€ par an et les frais de gestion sont calculés à hauteur de 15% des frais totaux ce qui donne 2,6 K€, 6,6 K€, 6,1 K€.

C2 - Autres soutiens financiers apportés au projet :

- L'équipe BD du LRI, en collaboration avec l'équipe Langages du LIENS, participe à un projet RNTL de 18 mois - GraphDuce, démarré en décembre 2003, avec la société BrixLogic, visant au développement d'interfaces graphiques pour CDuce dans la cadre d'applications financières.
- L'équipe BD du LRI, ansi que l'équipe Langages du LIENS, sont membres du consortium du projet CASC, ACI Sécurité Informatique 2003.

26. Sauf demande argumentée, la durée d'un contrat de type post-doc ne pourra excéder 12 mois.
27. Certaines des demandes déjà faites pour 2003-2004 pourront être attribuées au titre de l'ACI.

Action Concertée Incitative

MASSES DE DONNÉES

Descriptif complet du projet

D - RÉCAPITULATIF GLOBAL DES DEMANDES DU PROJET :

Financements via le Fonds National de la Science :

	2004 (09.04-08.05)	2005 (09.05-08.06)	2006 (09.06-08.07)	Total
Équipement	20	16,5	6	42,5
Fonctionnement (dont CDD décrits ci-dessous)	153,1 (87,2 CDD)	126,9 (55,2 CDD)	147,9 (75,5 CDD)	427,9 (217,9 CDD)
Total / année	173,1	143,4	153,9	470,4

Dépenses de personnels (CDD):

Nature de l'emploi (post-doc, ingénieur, assistant-ingénieur,. . .)	Post-Doc
Durée de l'emploi (en mois)	12 mois (mois 1-12 du projet)
Coût total de l'emploi	45,8 K€

Nature de l'emploi (post-doc, ingénieur, assistant-ingénieur,. . .)	Ingénieur d'études
Durée de l'emploi (en mois)	18 mois (mois 18-36 du projet)
Coût total de l'emploi	60,9K€

Nature de l'emploi (post-doc, ingénieur, assistant-ingénieur,. . .)	Assistant Ingénieur
Durée de l'emploi (en mois)	12 mois (mois 1-12 du projet)
Coût total de l'emploi	36,6K€

Nature de l'emploi (post-doc, ingénieur, assistant-ingénieur,. . .)	Stagiaire
Durée de l'emploi (en mois)	3 mois (an 2)
Coût total de l'emploi	7,2 K€

Nature de l'emploi (post-doc, ingénieur, assistant-ingénieur,. . .)	Stagiaire
Durée de l'emploi (en mois)	3 mois (an 3)
Coût total de l'emploi	7,2 K€

Nature de l'emploi (post-doc, ingénieur, assistant-ingénieur,. . .)	Post-doc
Durée de l'emploi (en mois)	12 mois (mois 18-30 du projet)
Coût total de l'emploi	45,8 K€

Nature de l'emploi (post-doc, ingénieur, assistant-ingénieur,. . .)	Stagiaire
Durée de l'emploi (en mois)	2 mois (an 1)
Coût total de l'emploi	4,8 K€

Nature de l'emploi (post-doc, ingénieur, assistant-ingénieur,. . .)	Stagiaire
Durée de l'emploi (en mois)	2 mois (an 2)
Coût total de l'emploi	4,8 K€

Nature de l'emploi (post-doc, ingénieur, assistant-ingénieur,. . .)	Stagiaire
Durée de l'emploi (en mois)	2 mois (an 3)
Coût total de l'emploi	4,8 K€

Financements via les organismes de recherche :

	2004–2005	2005–2006	2006–2007	Total
Nombre de post-docs (préciser pour chaque demande la durée en mois) [28]		2 x 12 mois		2 x 12 mois
Nombre d'accueils de chercheurs étrangers (préciser pour chaque demande la durée en mois)	1 x 1 mois + 1 x 2 semaines	2 x 1 mois + 1 x 2 semaines	2 x 1 mois + 1 x 2 semaines	6,5 mois
Nombre d'acceuils en délégations ou détachements		1		

Allocations de recherche :

	2004–2005	2005–2006	2006–2007	Total
Nombre d'allocations de recherche débutant en :	1	1		2

28. Sauf demande argumentée, la durée d'un contrat de type post-doc ne pourra excéder 12 mois.

Action Concertée Incitative

MASSES DE DONNÉES

Descriptif complet du projet

E - ENGAGEMENT DU COORDINATEUR DU PROJET :

La présente page ne sera remplie que dans la version sous forme papier.

Je soussigné, **Giuseppe Castagna**, coordinateur du projet **TRALALA**, m'engage dans l'hypothèse où le présent projet serait retenu à :

- fournir un rapport d'évaluation à mi-parcours permettant au Conseil Scientifique d'apprécier l'avancement des travaux et la coopération des équipes participantes,
- un rapport à la fin de l'exécution du projet,
- maintenir régulièrement une page web résumant l'ensemble des activités du projet.

Signature du coordinateur du projet :

Visa du Directeur du Laboratoire ou de l'Unité de Recherche auquel appartient le coordinateur du projet: