Nom et prénom du coordinateur / coordinator's name	CASTAGNA Giuseppe		
Acronyme / Acronym	TYPEX		
Titre de la proposition de projet	Intégration des approches langage, logique et orientée données pour un traitement XML certifié, dirigé par les types		
Proposal title	Typeful certified XML: : integrating language, logic, and data-oriented best practices		
Comité d'évaluation / Evaluation committee	SIMI 2 - Science informatique et applications		
Projet multidisciplinaire / multidisciplinary proposal	OUI [✓ NON] Si oui indiquer un comité secondaire:		
Type de recherche/ Type of research	[✓ Recherche Fondamentale / Basic Research] Recherche Industrielle / Industrial Research Développement Expérimental / Experimental Development		
Coopération internationale / International cooperation	OUI [✓ NON]		
Aide totale demandée/Grant requested	372 557,12 €	Durée du projet / Proposal duration	36 mois

Contents

1 Résume de la proposition de projet / Proposal abstract 3

2 Contexte, positionnement et objectifs de la proposition / Context, positionning and objectives of the proposal 4
2.1 Contexte de la proposition de projet / Context of the proposal 4
2.2 État de l'art et position de la proposition de projet / state of the art and positionning of the proposal 6
2.3 Objectifs et caractère ambitieux et/ou novateur de la proposition de projet / Objectives, originality and/or novelty of the proposal 12

3 Programme scientifique et technique, organisation de la proposition de projet / Scientific and technical programme, Proposal organisation 13
3.1 Programme scientifique, structuration de la proposition de projet/ Scientific programme, proposal structure 13
3.2 Description des travaux par tâche / Description by task 15
3.3 Calendrier des tâches, livrables et jalons / Tasks schedule, deliverables and milestones 27

4 Stratégie de valorisation, de protection et d'exploitation des résultats / Dissemination and exploitation of results, intellectual property 29

5 Description du partenariat / Consortium description 32
5.1 Description, adéquation et complémentarité des partenaires / Partners description and relevance, complementarity 32
5.2 Qualification du coordinateur de la proposition de projet/ Qualification of the proposal coordinator 34
5.3 Qualification, rôle et implication des participants / Qualification and contribution of each partner 35

6 Justification scientifique des moyens demandés / Scientific justification of requested ressources 36
6.1 Partenaire 1 / Partner 1 : PPS 37
6.2 Partenaire 2 / Partner 2 : WAM 38
6.3 Partenaire 3 / Partner 3 : LRI 39

7 Annexes/Annexes 41
7.1 Références bibliographiques / References 41
7.2 Biographies / CV, resume 44
7.3 Implication des personnes dans d'autres contrats / Staff involvment in other contracts 52

1 Résume de la proposition de projet / Proposal abstract

All the three partners of this proposal work at developing formal techniques to smoothly define, statically analyze, efficiently implement and optimize, transformations of documents in XML format. To reach its objectives each partner uses a different approach (in which they have a world renowned expertise): a logical approach based on solvers for WAM, a programming language (PL) approach for PPS, and a data-oriented approach for LRI. Each approach achieves different goals and interestingly, though not surprisingly, the strengths of the one are often the weaknesses (or the "future work" issues) of the other. The objective of this project is to produce a paradigm shift for the manipulation of XML data. In order to achieve this goal we will mutualize experiences and know-how of each group and use them first to improve, by cross-fertilization, the research in each specific domain, and then to integrate them into a unique framework. Accordingly, we plan to use functional languages techniques to enrich the μ-calculus of our solver with polymorphic variables and to statically analyze (in order to efficiently materialize) transformations in XML engines. We will reengineer the techniques we developed for the solver to handle backward axes so as to enrich with upward moves the automata used to efficiently query native XML engines. We will modify the solver to fit the resolution of constraint systems generated in the type inference of the application of polymorphic functions. We will use the automata theories developed for querying XML systems to define iterators and study the parallelization of functional languages to manipulate XML. We will adapt the techniques used for typing XML programming languages to enrich the solver (or its meta-theory) with higher-order polymorphic transformations. We will transpose the PL techniques developed for static analysis of programs, to standard processing langagues for XML data. We will use the Coq proof assistant to develop formal specifications for XPath, tree automata, and μ-calculus (the core tools of our project), verify the implementations of the solver and of the query engine against these specifications, and formally guarantee their efficiency.

The highly ambitious and final goal of this project is to produce a new generation of XML programming languages stemming from the synergy of integrating the three approaches into a unique framework. Languages whose constructions are inspired by the latest results in the PL research; with precise and polymorphic type systems that merge PL typing techniques with logical-solver-based type inference; with efficient implementations issued by latest researches on tree automata and formally certified by latest theorem prover technologies; with optimizations directly issued from their types systems and the logical formalizations and whose efficiency will be formally guaranteed; with the capacity to specify and formally verify invariants, business rules, and data integrity. Languages with a direct and immediate impact on standardization processes.

2 Contexte, positionnement et objectifs de la proposition / Context, positionning and objectives of the proposal

2.1 Contexte de la proposition de projet / Context of the proposal

Major scientific and technological bottlenecks: The absence of parametric polymorphism is a major lack in the practice of programming XML transformations. This is witnessed by the fact that this feature has repeatedly been requested to and discussed in various working groups of standards (*eg*, RELAX NG [18] and XQuery [20]). Polymorphism for XML not only brings well-known advantages already verified in existing functional languages (*eg*, the typing of `map`, `fold`, and other functions that are standard in functional programming) but also new usages peculiar to XML. A typical example is SOAP [54] that provides XML "envelopes" to wrap generic content. Functions manipulating SOAP envelopes are thus working on polymorphically typed objects encapsulated in XML structures. Polymorphic higher-order functions are also needed in the development of dynamic web sites in order to register new web pages whose content and uri parameters—the so-called "query strings"[24]— are essentially polymorphic (*eg*, see [16]).

Even the most recent (native) XML engines do not fully exploit typing information nor do they account for the correction of their implementations. This is partly due to the fact that the data management community often concentrates on expressiveness and complexity issues (*eg*, what are the expressive power and related complexity of a given query language) or tries to revamp some of the relational data model characteristics in the context of XML (*eg*, how to define in such a setting functional dependencies or normal forms) rather than trying to reason about data and programs. This lack of ability to reason about data as well as programs explains why no fully satisfactory XML engine exists to this date. However, being able to reason about data and programs has always been identified as one of the major issues to yield efficient, safe, and reliable applications. As an example, a major technological bottleneck in current database programming language technology is the lack of static verification of assertions and integrity constraints. This problem is already crying apparent in relational databases. Although assertions are included in the SQL3 standard and explained in all databases textbooks, no commercial or academic relational engine implements their definition or, *a fortiori*, their static verification (the only constraint checking mechanisms effectively implemented are key constraints, foreign key constraints and triggers, which do not match the needs covered by the assertions). The presence of this bottleneck can be easily explained by the fact that solvers, theorem provers, and proof assistants, whose usage is widespread in the programming language community, have so far received a very limited consideration from the data management community which applies them only to solve expressivity and complexity problems and disregards their application to language and software engineering issues. In the particular context of XML, being able to reason *about both programs and data* is a major bottleneck, as recently pointed out at the 2010 Workshop on Updates in XML [23], that when worked out will solve many different problems such as efficient implementations of XML engines, safe behaviour of programs, and secure systems. However, hitherto, no serious effort has been devoted to exploit results from other fields, in particular type theory, static analysis, and theorem proving, to yield better languages and more efficient, safe, and secure systems. Besides providing a way to reason *about programs and data*, theorem proving can be used to guarantee key properties of the *tools themselves*, including termination, complexity or correctness of optimizations, which are barely considered usually. The development of reference implementations that provably verify the standards would drastically improve confidence in such tools and specifications. The aim is to develop formal models so that they become a practical guide during the design and implementation of XML technologies.

A very focused and tightly integrated project: The potential outcome of removing any of the previous bottlenecks has raised the interest of many research groups in world leading institutions. Thus this project will take place in a highly competitive international research context. In order to maximize the chances of success of our project in such a competitive context we kept it very focused and tightly integrated. The project is composed by just three partners with a limited number of participants from each site, in order to avoid dispersion of the research effort. These partners already have strong and well established connections: some of the participants of a site had previous post-doctoral or doctoral experience in one of the other sites (*eg*, Kim Nguyen, Matthieu Sozeau and, we hope in future, Nils Gesbert) and they all share common research experiences both in terms of joint publications, of joint development of prototypes and applications, and of joint participation to common national research projects. The scientific barriers to be attacked are clearly identified and all the participants share common interest to attack them. This is witnessed by the fact that, though at different degrees, every research task we enumerate in Section 3.2 involves participants from all partners. There will be no closed-in partitioning of the research effort, as often happens instead in larger or less focused projects.

Young researchers training and education: We plan to further tighten connections and mutual integration of the partners by jointly training young researchers. This is a key aspect of the project. At the moment of the writing of this proposal the three partners have already jointly proposed several master thesis subjects (in particular in the master curricula of MPRI in Paris and of ENS Lyon) to be jointly supervised. Each of these subjects roughly corresponds to one of the research tasks that structure our project and that we describe in Section 3.2. We aim to attract excellent students at least on a couple of these subjects so that they will be ready to start a PhD. thesis on them in fall 2011 exactly at the moment in which this project, if accepted, should start. As we explain in Section 6, we plan to transform post-doc positions in PhD. grants whenever an excellent candidate trained during spring-summer semester 2011 were available.We consider crucial such a flexibility in resource allocation, and we agree to perform transfers of resources among the participants in order to match the needs and the contingencies. As a matter of fact, we consider the personnel requests of this project as being mutualized to the whole project (though for administrative reasons they will be split among the participants) insofar as the requested personnel—both PhD. students and post-doc—will form the connecting frame between the different partners, by physical mobility, joint supervision, and collaboration. Furthermore the effort already started towards graduate students will continue all the project long (see in Section 6 the numerous master project proposals we plan to cover to achieve our objectives).

Integration in an international research framework: The tight interconnections between the different partners whose assembly we described in the previous section is just the lower layer of the two layers structure of our project. The second layer will be provided by international links since this project will unroll in and integrated within a highly competitive international research context.

All partners maintain tight links and collaborations in the international community. Besides the participation to the national project listed in Section 7.3, our research will be at the interface with many international research programs and initiatives thus playing a bridging role between national and international research collaborations. The group in PPS has active collaborations with the University of Torino (Italy), and the Imperial College of London (UK). The research on polymorphism for XML is performed in conjunction with the Institute of Software of the Chinese Academy of Science (ISCAS) in Beijing, within the framework of a research project of the National Natural Science Foundation of China ("unranked tree-structured data and semantically defined polymorphic type systems", grant no. 61070038). One of the PhD students of PPS who will take part in this project (Zhiwu Xu) is the main actor of the research on polymorphism, which unwinds in the context of a PhD. thesis jointly supervised by Paris 7 and ISCAS.

The WAM EPI has several international collaborations on the XML type-checking. Notably, James Cheney from University of Edinburgh uses WAM's XML static analyzer to perform independence analysis for XQuery updates [3]. The solver is also used to check whether an XQuery update affects the result of a query (a materialized view). The solver is also used to determine the satisfiability for conjunctive queries over trees [17]. WAM has an active collaboration with Lionel Villard, former PhD student in the team, of the T.J. Watson IBM Research Center in New York. The subject of the collaboration is the design of integrated development tools (IDEs) for XQuery. WAM started a collaboration with Moshe Vardi with the goal of unifying technical machinery used in proofs such as two-way alternating tree automata with solver satisfiability testing techniques. The goal is to reduce the gap between XML theory and practice [14, 13].

The LRI partner has active collaborations with National ICT Australia (Australia). Prior joint work include SXSI (Succinct XML Self Index, [1, 43]), an XML database handling efficiently storage of documents and XPath queries, which will be used as a framework for the implementation of theoretical results obtained in this project. SXSI is jointly developed by Kim Nguyen (of LRI) and Sebastian Maneth (NICTA).

Finally, the PPS and LRI groups have recently started to interact with Jérôme Siméon (codesigner for XQuery 1.0, for XPath 2.0, and for the XQuery update, and XQuery scripting extensions) of the T.J. Watson IBM Research Center in New York to set up a collaboration to start in September 2011 on the themes of this proposal (in particular on the aspects of this proposal that concern XQuery 3.0).

2.2 État de l'art et position de la proposition de projet / state of the art and positionning of the proposal

XML provides a way to specify document formats, but the core of the XML technology is the *manipulation* of such documents. Indeed, as XML documents are but a way to organize pieces

types\values	−	+
−	C, Java, C♯, OCaml, Haskell, PHP,. . .	XQuery, XSLT, XPath, . . .
+	Ocsigen(OCaml), JAXB, Scala, . . .	XDuce, Xtatic, ℂDuce, XHaskell, OCamlDuce, XAct. . .

Table 1: Four categories of XML programming

of information, then one needs to *query* the document to extract information, *process* the information to produce a *result* which then needs to be *published*.

The simplest way to do so is to consider the XML document in its roughest form: a text file. This technique shows rapidly its limits since hand-writing a parser can be error prone and nullifies the advantages of a generic format. Besides, document constraints (henceforth, the *type* of the document) are not considered, let alone enforced, by such a low level of programming. It is doubtful that any serious XML-based program can be written in such way.

If we consider XML programming at a higher level then, in the same way as XML documents are twofolds —the brute content and its type— we can consider two aspects of a programming language, with respect to XML: whether or not it provides syntactic support to process XML documents (value side) and whether or not it can enforce document constraints (type side).

Both the type-system and the semantics of a programming language can be either of strong or weak flavor with respect to XML programming, as shown in Table 1.

The first category, is the one of generic programming languages (C, Java, ...). On the value side, these languages do not have any specific support for XML. They therefore rely on external libraries to perform parsing, manipulation, and printing of XML documents. Note that although it is possible to write any XML transformation in these languages, the best that can be done to ensure the validity with respect to a set of constraints is to use a *validating parser* to load a document and to validate every generated document afterward, to dynamically ensure its correctness with respect to its type. Such development might lead to awkward debugging efforts since it is difficult in general to link an error in a generated document to the corresponding faulty line of code in the program.

Amongst the members of the second category, one finds for example the standards specified by the W3C. As surprising as it might seems, while the specifications for both types (DTD, XML-Schema,. . .) and languages (XPath, XQuery, . . .) are very detailed, no typing policy is formally described for these languages which are often very lightly typed. However, as they provide syntactic supports for XML data as well as highly declarative constructs inherited from database programming languages, they offer a more comfortable way to code XML transformations.

The third category may puzzle the reader: how can a language which was not designed for XML (a generic language) have an XML aware type-system? The answer is that many languages have a type-system expressive enough to encode *some* XML type constraints thus giving added safety to the use of the previously mentioned XML libraries. For example, the Ocsigen project [2, 48] is a framework for developing web pages written in OCaml. Within this framework, XHTML type constraints are encoded into OCaml's polymorphic variants (via the clever use of phantom types: types that carry extra information unrelated to the values of this types to guide the type inference algorithm and to help it enforce extra constraints). Other tools (such as JAXB [40] or Castor [15] for example), create a *mapping* from a DTD (or a Schema) to a Java class. In a nutshell, they automatically generate a Java class hierarchy. This permits to use the type system of the language to enforce XML type constraints. However the usage of such a technique (known as *data-mapping*) is quite limited. First, it lacks flexibility and modularity (one has to regenerate a new class when the corresponding XML type is modified). Second, and foremost, a type system not designed for XML cannot provide the necessary precision (*eg*, usually they cannot account for distributivity laws of the union[1]): it is a "better than nothing" approach.

Finally, the last (but not least) category, is the one of statically typed languages designed for XML processing. These are languages whose design is directly issued from the study of XML types (more generally of regular tree types [35]), and whose main constructs are specifically conceived to efficiently work on regular trees, in particular by using the static XML type information available.

Our project will focus on the languages in the rightmost column of Table 1. In particular we aim at:

1. providing exact type systems which are missing for the second category of languages (in particular XQuery) and add to them advanced features such as higher-order functions, pattern matching, and statically verified assertions (as proposed in the XQuery 3.0 draft).

2. improving and certifying the implementations of the second category of languages and enhance their XML processing capabilities by enriching them with syntactic constructs developed in the fourth category of languages (in particular for XDuce and ℂDuce),

3. extending the type systems of the fourth category of languages (in particular ℂDuce) with polymorphism, a features often bespoken for the languages in the second category. The final goal being to transpose the results we will obtain on polymorphic functions for ℂDuce also to the languages of the second category.

In the rest of this section we review the state of the art for the techniques at the core of our project, that is, the study of functional languages designed for XML programming (§2.2.1), the use of solvers to infer precise types of XML standards (§2.2.2), the techniques used to analyze XML data and programs (§2.2.3) the use of tree automata for efficient implementation of XML querying and processing (§2.2.4).

[1]For instance `<a>(Int|Bool)</a>` and `(<a>Int</a>)|(<a>Bool</a>)` are considered distinct types.

2.2.1 Functional programming languages for XML

One of the safest (and in our opinion most elegant) ways to program with XML is to "take types seriously" into account. Indeed, in a type-system which is fully XML-aware, if one can check that a transformation (a function) has type $t \to s$ then it is guaranteed that the output of the transformation will always have type s, for any input of type t. In the case where s is an XML type (say the XHTML DTD for example), then the type-system *ensures* that the program will *always* output a valid XHTML document, without the need of further validation. This is the trend that has been initiated with Haruo Hosoya's work [35, 38, 39] (in collaboration with Benjamin Pierce and Jérôme Vouillon) which led to the development of the XDuce language. In the cited works, Hosoya and his collegues explain that DTD, Relax-NG or XML-Schema are all nothing but some concrete representations of *regular expression types*. Regular expression types are nothing else than regular tree languages and as such, have nice properties of closure under Boolean connectives, decidability of emptiness, membership and all the operations required to forge a type system (see for example [19] for a complete survey on tree languages processing and recognition).

In XDuce, documents are first class values and types are regular expression types. In this setting, the first major contribution of XDuce, is the definition of *regular pattern matching*, a generalization of *pattern-matching* as found in many (but mostly functional) programming languages, such as SML, OCaml or Haskell, which selects efficiently sub-parts of an input document in a precisely typed way. The other contribution of paramount importance is the definition of the so-called semantic subtyping, which is uniquely suited to finely check constraints on XML documents.

XDuce has been further extended, in many different directions. A first extension, to which the WAM group contributed, is the work by Benjamin Pierce *et al.* on Xtatic, which consists of the addition of XDuce regular expression types and regular expression patterns to C♯ (see [30]). This work is is quite original in the research on the topic insofar as it embeds XDuce features in an *object-oriented* language. While being of great benefit for a C♯ programmer and having led to several works on efficient implementation of pattern matching, the merge between XDuce notion of subtyping and C♯ presents no conceptual difficulty, since C♯ subtyping relation is rather simple (inheritance) and defined by the programmer; when a class inherits from another, both classes are added to the subtype relation.

Another extension, which is one core of expertise of the LRI and PPS partners, is the ℂDuce language defined by these two partners of the project. The core of ℂDuce is to extend the notion of (semantic) subtyping to arrow types, and add (amongst other things), higher-order and overloaded functions to XDuce (in which functions were not first class values). We also generalized the notion of regular pattern matching, exposing all Boolean connectives to the programmer (intersection and negation of patterns and types, while XDuce only provides union to the user). Furthermore, we use an efficient execution model for patterns, based on special kind of automata (dubbed NUA, for Non-Uniform Automata) enriched with type information [26]. In particular, these automata compute the matching of a value by a pattern *without backtracking* and by doing as few operations as possible.

The absence of parametric polymorphism (of the kind we describe right below) is a very important lack in the practice of programming XML transformations. This is witnessed by the fact that this feature has repeatedly been requested to and discussed in various working groups of standards (*eg*, RELAX NG [18] and XQuery [20]). In spite of that *all* the languages in Table 1 lack polymorphism for XML types. As a matter of fact, for the languages in the fourth category, which are the richest in terms of XML expressivity, the addition of polymorphic has been deemed unfeasible on the light of some results we presented at POPL'05 [36, 37]. A very recent breakthrough obtained by the PPS group has shown that this was not the cases, and we were able to provide a semantic subtyping relation for regular tree types with type variables [16]. Furthermore results developed by the LRI partner [47] show that parametric polymorphism of the kind studied by [16] is not the only possible way to add polymorphism to XML document transformations.

2.2.2 Logics for XML programming

Logics are good yardsticks to study the expressive power of programming languages. In particular they are very appropriate to study the standard languages in the second category. This is because these standards have been developed and became widely used very quickly in order to overcome the growing need for standard XML processing. Fundamental research has been focusing on the foundations of these languages only recently. In particular, Marx showed that a slight extension of XPath makes it first-order complete [44]. More expressive logics such as monadic second order logic make it possible to capture not only XPath but also regular tree types. Genevès and Layaïda showed the limitations of using monadic second order logic solvers, such as MONA, to solve simple static analysis tasks such XPath containment [32]. In that context, modal logics such as the μ-calculus are specifically interesting because they are equivalent to monadic second order logic in terms of expressive power, yet admitting much more efficient decision procedures (typically exponential time). In [33], a tree logic based on the μ-calculus is proposed. It constitutes the best balance between expressivity, complexity, and practical efficiency known today. Another advantage of modal logics over classical logics is that they are much more robust by extension. The logic presented in [33] opens the way to the development of even more expressive and succinct yet decidable tree logics adapted for XML processing, providing support for counting features, nominals, data value comparisons, etc. One other perspective is to transform logical solver in order to build the set of all satisfying models of a given formula, and not only one of them, like it is the case for most logical satisfiability checkers. We believe this could be an important step in the study of polymorphism from a logical perspective.

2.2.3 Reasoning about XML data and programs

Recent (native) XML/XQuery engines, among them the most advanced ones [29, 45], do not really exploit typing information nor do they account for the correction of their implementations. This is partly due to the fact that the data management community often concentrates on expressiveness and complexity issues (*eg*, what are the expressive power and related complexity of a given query language) or tries to revamp some of the relational data model specificities in the context of XML (*eg*, how to define in such a setting functional dependencies or normal forms [25, 9]). However, several, though scattered, techniques have been used in the past to analyze and reason about data processing. In the context of object-oriented databases static analysis techniques based on abstract interpretation combined with first order theorem proving have already been applied [5, 6] by the LRI partner to solve the problem of static checking of integrity constraints. In [50, 51] the same problem has been addressed using a higher order theorem prover (Isabelle). Surprisingly, those works have been published in the programming language community and received little interest in the database one. Information flow type systems, developed in the area of language-based security [53, 52, 49] are starting to be applied to XML data management systems to help track data provenance [11, 12, 10] and confidentiality [4].

Systems of dependent types including data constraints (*eg*, as in Z, VDM, etc) have long been able to express database integrity constraints, but with recent advances in automation (*eg*, SMT solvers) typecheckers can now verify statically that queries and updates respect these constraints [7]. The quest for verified software is starting to come true, thanks in no small part to advanced type systems found in interactive proof assistants [41]. Recent work on verified implementations of database systems starts to address the correctness question for database implementations [42] or DOM ones [31].

However, to this date no XML engine relies on and more importantly integrates those techniques in a unique environment. In particular, no or very little attention has been devoted to the implementation of type aware XML engines nor correctness issues of their implementations have seriously been pursued.

2.2.4 Tree automata for XML processing

Introduced in the late 60s as a proof mechanism, tree automata have recently become popular in the context of XML programming. Firstly, tree automata are tool of choice to represent XML types (DTDs, XML-Schemas or more generally, regular tree types). Their theoretical properties are well known, and “well-behaved” (for instance they enjoy closure properties under union, intersection and negation, and their inclusion is decidable). They are thus the perfect candidate to represent XML type systems. But tree automata (and their extensions) have also be used to reason about XML programs (more precisely about XML queries, such as XPath queries [46, 8]). The recent work of S. Maneth and K. Nguyen ([1, 43]) has shown that these same tree automata can be used as an effective compilation target for XPath queries (it was already known that this was the case for XML pattern matching *à la* ℂDuce). Therefore they seem to be the key formalism to bridge a long standing gap between two aspects of XML programing. On one hand, extensive work as been done on formal properties of XML languages (e.g. expressivity of XPath, complexity of several problems, . . .). On the other hand –and almost completely disconnected– there exists efficient XML query engine (such as MonetDB/XQuery or QizX). Such engines work very well in practice but little is known on their real complexity or even their correctness: they rely on folklore and clever sets of optimizations, none of which have been proved formally.

2.3 Objectifs et caractère ambitieux et/ou novateur de la proposition de projet / Objectives, originality and/or novelty of the proposal

The main novelty and originality of this project is the three-pronged approach it proposes. The idea is to combine the research in programming languages (specifically, in program verification and in XML processing) with a logic-driven approach and the results coming from the research on the query engine technology. As long as we know, the integration of the results and achievements of all the three fields together into a unique framework was never tried before.

Of course there already exist several points of contact and intersections between these topics but they always concern just two of the three approaches and they focus on specific problems. For instance, the implementation of both static and dynamic semantics of typed languages for XML such as ℂDuce and XDuce lies in the tree automata techniques used for query engines. Similarly, the database community has been heavily using the connections between databases and temporal logic to establish new complexity results for various problems. Finally, to close the circle, temporal logics were introduced in the formal verification community for specifying and characterizing the behavior of programs, applied for modeling program states, and studied from the perspective of model-checking (more examples were given in Section 2.2).

The goal of this project is to remove *all* the major scientific and technological bottlenecks we detailed in Section 2.1 and, in doing so, to trigger a paradigm shift to process XML data. We will achieve it by mutualizing and integrating the experience and know-how of each group and use it to advance the research in order to provide programming techniques and tools covering the major part of the features found in standard programming languages (XPath, XQuery, etc.) with a good balance between expressivity, complexity, and practical effectiveness. Rather than attacking a particular problem (*eg*, view independence or security policies enforcement for XML) and solving it with ad-hoc solutions our approach is, in essence, generic. We seek for general, well-founded solutions that will then be instantiated to specific problems and practical issues. As a matter of fact, the quest for fully verified software is starting to come true thanks in no small part to breakthroughs found in type systems, theorem provers and interactive proof assistants. The purpose is to rely on the most modern formal tools: from advanced type systems (with semantic subtyping to handle semi-structured data), theorem provers (SMT, domain specific solvers) and last to proof assistants (Coq) to yield the production of fully verified XML-centric software.

The highly ambitious and final goal of this project is to produce a new generation of XML programming languages by combining the best efforts of several communities working on different aspects of XML: languages whose constructions are inspired by the latest advances in the programming language research; with precise type systems that merge PL typing techniques with logical-solver-based type inference; with efficient implementations issued by latest researches on tree automata and formally certified by latest theorem prover technologies; with optimizations directly issued from their types systems and the logical formalizations, whose efficiency will be formally guaranteed; with the capability to specify and formally verify assertions, business rules, and data integrity. Languages with a direct and immediate impact on standardization processes.

3 Programme scientifique et technique, organisation de la proposition de projet / Scientific and technical programme, Proposal organisation

3.1 Programme scientifique, structuration de la proposition de projet/ Scientific programme, proposal structure

As we outlined in Section 2.1 our project will have a two layered structure.

The lower level will be grounded in the tight integration of the different partners focused on common objectives. This integration already relies on common experience (exchanges of researchers, joint publications, joint work of development, joint participation to national research projects) and it will be tightened via a joint training program for young researchers. We prepared the ground for this project ahead of its submission by proposing jointly supervised master projects targeting at the tasks that form the workload of the proposal and described below. We aim at attracting several excellent students on these topics so that they will be ready to start a PhD. thesis right at the starting point of the project. These theses will be jointly supervised by at least two partners (with a program providing for physical mobility between different partners) and will reinforce the connection frame of the project. Topics that could not be covered by PhD. students will be covered by post-doctoral research.

On the higher level this project will graft onto a network of international research collaborations spanning four continents, insofar as whole parts of our project (*eg*, the definition of polymorphic extension of ℂDuce, the implementation of the SXSI query engine, or several aspects of the XQuery 3.0 draft) will be developed in conjunction with our international partners.

The tight integration of the partners will allow us to implement a very flexible policy of resource (re)allocation, thus maximizing chances of success. The partners agree on the principle of reallocation of human resources to match all contingencies we may meet as well in the development of the scientific program as in the recruitment process.

The planned workload is organized in tasks. Some tasks involve different tracks of research and their timeline is organized into steps. We have singled out five different tasks: the first task concerns coordination activities while research activities are in the remaining tasks two, three, four (which will develop in parallel), and five (which will combine achivements of the previous tasks).

Task two will concern the design and implementation of polymorphism for an XML platform. To this purpose we will integrate polymorphism both the in logical solvers and in the XML programming language ℂDuce. Both integrations will unroll conjointly, being the advancement of the one dependent from the other. From the ℂDuce perspective, we will extend type inference to polymorphic functions and their applications. A first step will be to study the class of constraints that must be solved in order to perform this kind of inference. In particular, we will define constraint generation for type inference local to polymorphic function application with semantic subtyping. These constraints will then feed the solver for resolution. Currently, the WAM solver is applied to first-order monomorphic typing problems, as the type-checking problem is defined in terms of satisfiability of an input-ouput dependency of XML types. To cope with polymorphic type inference and ℂDuce constraint resolution we must extend the solver in two directions. First, we will need to perform type synthesis in the solver since, in this context, not only it must check whether the constraints it receives are satisfiable but, if such is the case, it must also produce a type: the goal is to synthesize workable type representations, directly issued from the logical model, to be returned to ℂDuce. Second, the solver will have to actually find a solution (either theoretical or meta-theoretical) for the constraints carrying higher order transformations/types. The solver needs to be extended to support this kind of constraints. We will gauge the overall quality of our polymorphic extensions by applying them to the development of web applications and the manipulation of SOAP envelopes.

Task three is devoted to the extension and formal verification of the core tools of the project: the SXSI query engine and the μ-calculus solver. The research thereof aims at giving formal certifications for these efficient implementations, guaranteeing their correctness (even in presence of optimizations) and their algorithmic complexity. It is organized in two tracks. A first track will investigate the addition of backward axes to the XPath fragment handled by SXSI, making it a complete query engine. A second track will focus on the certification of our tools: it will start, in parallel to the first track, with the formal development of libraries (in the Coq proof assistant) for the representation of XPath queries and alternating tree automata (as used in the SXSI engine) and merge with the first track to produce a proof that the extension of SXSI with backward axes is correct w.r.t. the XPath model. On the same vein, in this second track we will certify the solver by producing formal proofs of its correctness and completeness w.r.t. the μ-calculus, as well as an automated proof of its complexity. This will set the basis for future extensions of the solver, and will also be put to use in the following task, to handle verification conditions formalized as μ-calculus formulæ or general logical constraints.

Task four is devoted to the addition of various kinds of static analysis of the XQuery runtime and in particular to provide a fully type aware *implementation* of an XQuery engine and to endow XQuery with verification capabilities. A first track within this task consists of the addition of ℂDuce's type system to XQuery. Here, the major challenge is typechecking in the presence of backward axes and possibly of updates. A second track, developed in parallel will add semantic annotations to XQuery programs (by means of program *assertions*, a new feature of the current XQuery 3.0 working draft). Such annotations will specify logical constraints, invariants and properties that a program must meet. To solve these logical constraints we will first resort to the μ-calculus solver developed by the WAM team and next improve the process by resorting to more sophisticated tools (SMT solvers or the Coq proof assistant).

Finally, Task five will integrate in a unique framework all achievements of the previous three tasks. It will describe the core of a unique paradigm that will include the advanced programming features studied in Task 3, with a precise and polymorphic typing as developed in Task 2 and with specification and verification capabilities developed in Task 3. Its implementation will directly stem from the XML engine issued from Task 3 and whose execution, optimizations, and complexity will be thus formally certified. This will form a coherent and complete definition of the fundamentals of next generation standard XML manipulation languages.

For all tasks we have established some milestones associated with concrete measurable success criteria. Intermediate milestones constitute the checkpoints in which we gauge the advancement of each task. As well as assessing the project's results, such evaluation is intended to be used by the Steering Committee for the scientific guidance of the project. Accomplishment of intermediate milestones will also trigger the start of new phases of the project: for instance the Task five will start as soon as at least two out of four selected milestones in Tasks two, three, and four will be completed. Final milestones are used to asses the success of tasks or of their sub-tracks. Finally, we have foreseen possible failures in matching some milestones's success criteria and already planned ahead possible countermeasures and alternative research directions (*cf.* Section 3.3). Dependencies *within and between* tasks are highlighted in Figure 1.

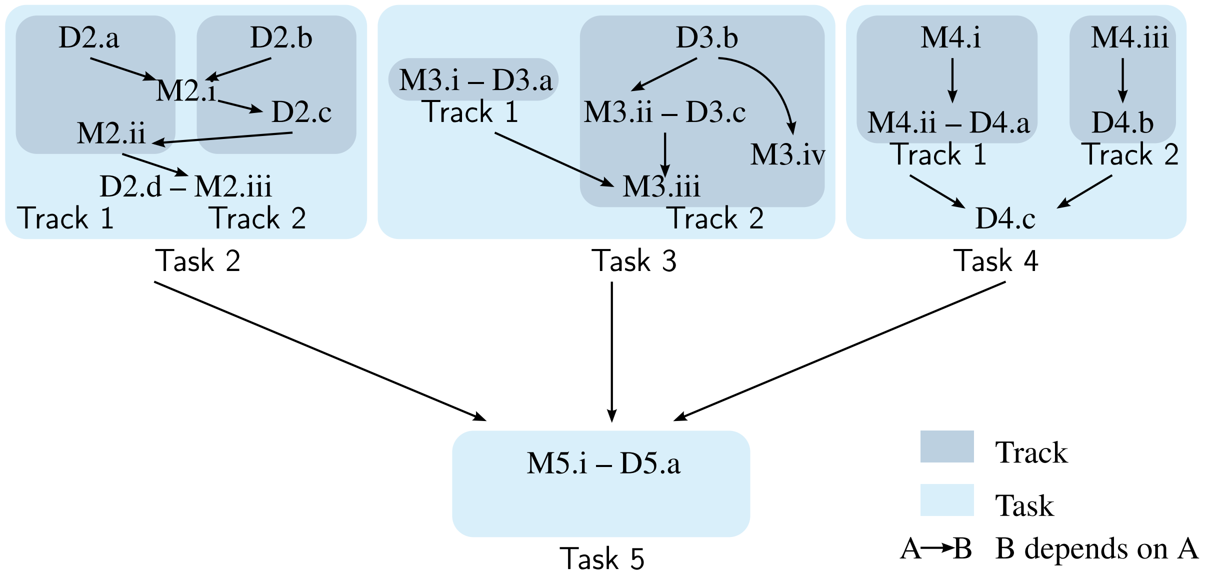

Figure 1: Dependency graph

3.2 Description des travaux par tâche / Description by task

Task 1: Project management and coordination.

Leader: Giuseppe Castagna

Participants: Site leaders

Teams: All

Typex Steering Committee. General decision will be coordinated by the Typex steering committee (SC), consisting of the site leaders, and headed by the coordinator. The SC is responsible for the scientific and strategic directions of the project. The SC meets at least annually. It is responsible for the resolution of the technical issues raised by the workplan, planning periodical objectives and establishing a self-evaluation plan, monitoring the annual advancement of the activities, check the success with respect to the self-evaluation plan and in case of a discrepancy deciding on possible modifications and contingent adjustments.

It is also in charge of the overall legal, contractual, ethical aspects and the coordination of knowledge management and technological transfer.

The Task Leaders assure the coordination and communication of the work done in the project on their themes. In particular, each Task Leader follows the work done in each subtrack of the task at issues, ensures the achievement of its objectives, and is responsible of the preparation of the connected deliverables and of the coordination with dependent tasks.

A project's Advisory Board will be composed of invited personalities external to the project. The main role of this board is to review the scientific quality and relevance of the project's productions, further to the internal reviews regularly carried out by the SC. It may also be consulted, where the SC deems useful, on selected technical matters concerning the project. The SC will be responsible for inviting personalities to sit on the Advisory Board. Members of the Board are expected to be world renowned experts in their field and we will privilege people with whom we have active international collaborations, in order to enhance the interface role of the project.

Typex Regular Meetings. The main exchange forum will be common meetings with scientific presentations, software demonstrations, and discussions.

We plan meetings of an average length of 1.5 days, at least once a year. We will attach meetings to appropriate international conferences (ICFP, POPL, PLDI, ETAPS, WWW, VLDB, ICDE, ICSE, ...) if possible, so that we can benefit from the international community, while enhancing the visibility of French research world wide. When possible we will organize joint meetings with related ANR projects.

Exchange between partners will be mostly promoted via exchange visits both of PhD students and confirmed researchers and via partial meetings on specific project tasks.

Whenever possible, international experts will be invited to take part to all these meetings. Invitation will target foremost members of the Advisory Board but also members of similar foreign projects, in order to gauge the respective advancements.

Typex e-collaboration: The coordinator will be charged to maintain a Wiki Web site, that will always contain up to date information to current research. Large parts of the Wiki will be public, while others are reserved for internal discussions and co-operations.

In order to minimize the costs and maximize responsiveness, we will massively resort to e-cooperation resources such as audio and visio conferencing systems. In particular we will use the open (and free) EVO distributed collaboration system developed by Caltech. EVO provides collaboration services such as visio-conferencing, shared documents, shared white-boards and shared screens and it fully meets the requirements for usability, quality, scalability, adaptability to a wide range of working environments, reliability and cost. All partners have a longstanding experience of performing scientific research via e-collaboration resources and already have the equipment necessary to it: as a matter of fact the large bulk of this proposal was prepared via visio-conferencing.

Whenever technically possible, we will video-record presentations at the project meetings and make it available on the web-site.

Self evaluation plan and risk analysis: Typex is a highly ambitious project, as it aims at integrating a yet scattered range of foundational disciplines and techniques towards practical, data manipulation-oriented goals, which has great potential for a significant impact on the scientific and technological advances in the information society.

In general, and high-level terms, the success of the research plan will be determined by the scientific community's acceptance and adoption of its ideas and developments (both theoretical and practical), and by the resulting evaluation of the project's outcomes to contribute to creation of new standards and to influence the evolution of existing ones.

At a finer level of granularity, the progress of activities in Typex will be monitored and evaluated by assessing each task systematically and thoroughly against its targets, as measured by set quality and success indicators. As well as assessing the project's results, such evaluation is intended to inform the SC in the scientific guidance of the project and in the day-to-day management of contingencies.

The evaluation will take place in rounds coincident with major milestones of the project. In each round, a number of tasks steps will be assessed: those that have completed since the previous round will be measured against the respective quality criteria, those still in the run against their set intermediate checkpoints.

The outcomes of the self-assessment process is mostly internal to the project and if deemed necessary the SC may resort to the project advisory board.

Financial and legal management: The project coordinator will be supported by a new service for management and exploitation of intellectual property set up by the University Paris Diderot - Paris 7. In this service an ANR project officer will provide administrative and financial monitoring of the program and back-office support, and notify the coordinator of the deadlines established by the ANR (interim progress reports, receipt of the final costs ...). Administrative project management will be provided by a Design Engineer (Ingénieur d'Études) member of project coordinator staff who will be assisted by the Office of Contracts. The latter will provide the advance of funds, check the eligibility of expenditures and establish the balance sheets of the project.

Task 2: Polymorphism and solvers.

Leader: Nabil Layaïda

Participants: V. Benzaken, G. Castagna, P. Genevès, N. Layaïda, K. Nguyen, A. Schmitt, J. Vouillon

Teams: LRI, PPS, WAM

This task constitutes an important part of this project in terms of research effort. It aims at combining the results on polymorphism obtained in the research on functional languages with the inference capabilities obtained by the research on solvers.

This task will be organized in two distinct, though strictly connected and intertwined, tracks.

Track 1. Polymorphic functions for XML programming A first research track will aim at adding polymorphism to ℂDuce's semantic subtyping. Although the recent results on polymorphic subtyping we hinted at in Section 2.2 can be straightforwardly used to add polymorphic types and subtyping to ℂDuce type system, the addition of polymorphic *functions* to ℂDuce still is a very challenging problem. The problem is in the combination of polymorphism and subtyping. The presence in the ℂDuce type system of intersections, negations, and unions makes type inference algorithmically very complex since it generates sets of constraints that are very difficult to deal with (in particular because of the presence of negation, that makes most inductive reasoning fail). A complete inference of types is already impossible even without type variables: because of overloading—ie, intersection types—functions must be already annotated by their types in the ℂDuce language (failing to do so would yield an unsound system: *cf.* [28]). However, a light form of type inference is performed when applying functions since the type of the result is computed by taking into account the subtyping relation (the so-called subtype polymorphism) thus avoiding the use of explicit coercions. We want to extend this type of inference also for the application of polymorphic functions and thus obtain "local" type inference. The crux of problem will be that we will have to infer not only type coercions but also type instantiations.

Track 2. Polymorphism and higher-order types for logical solvers Current solver technology is essentially applied to first-order monomorphic typing problems insofar as the type-checking problem is defined in terms of satisfiability of an input-ouput dependency of XML types: it consists in checking whether the (logically encoded) execution of an XML transformation under some input type constraint is included in an output type. This approach is monomorphic since it checks satisfiability for single type constraints rather than for families of type constraints. In particular it cannot track how the transformation operates on generic parts of the input type (for instance it cannot follow how the content of a SOAP envelope will occur in the result: it can do it only for a SOAP envelope whose content is of a specific type). Furthermore the approach is not higher order since neither it can reason about transformations that are parametric in other transformations, nor it accounts for types of the transformations (just XML document—*ie* values—types).

This second research track aims at enriching the theory so as to be able to include polymorphic types and find a solution (either theoretical or meta-theoretical) for dealing with higher-order transformations/types.

Timeline: The research for this task will proceed in four steps, each steps belonging to both tracks (we cannot really assign the steps to a particular track, as we do in the following Tasks since although steps 1 and 2 have a more track 1 and track 2 flavor, respectively, the advancement of the two tracks will be interleaved).

1. A first step will be to study the class of set of constraints that we will need to solve in order to perform this kind of inference. To do that we will define a calculus (similar to the core calculus of [27]) enriched with explicit type coercions and type instantiations, look for canonical forms of their use (typically by defining a weakly normalizing rewriting systems on terms) and study how and when it is possible to infer the coercions and instantiations in the canonical forms. The main contributors of this research will be the LRI and PPS. In particular LRI will bring its expertise in defining polymorphic paradigms (Nguyen), while the PPS will bring its double competence in subtyping polymorphic types and inference of type constraints (Castagna, Vouillon). The results of this research will be summarized in a research report that is expected to be ready by the month 12 of the project.

   Deliverable 2.a: **(report)** Constraint generation for inferring types in polymorphic function application with semantic subtyping (M12)

2. In parallel, we propose to study and implement type synthesis, which paves the way for the research on polymorphism from a logical perspective. The logical solver currently looks for a finite tree over which the logical formula is satisfied. We propose to study how to compute the set of all satisfying trees. One difficulty is to build a workable yet succinct representation for the synthesized type from the logical model of types and paths. In particular, recursion should be captured and backward axes may need to be rewritten in the forward direction if we stick to the usual representation of tree grammars, such as the ones used in ℂDuce. This work will be coupled with an extensive search of heuristic in order to present to the programmer readable types. The main contributors of this research will be the WAM team with their past experience and expertise in the design and implementation of the logical solver. This phase is necessary to reach a point where we can cross-fertilize research on polymorphism, and in particular make use of the solver techniques to verify whether the constraints defined above are satisfiable and, if so to find a solution for them.

   Deliverable 2.b: **(report)** Type synthesis for the logical solver (M12)

   These first two steps naturally induces a first important milestone

   Milestone 2.*I **(M15):* Polymorphic types for languages and solvers.
   Success criteria: higher-order transformations can be polymorphically typed with local inference.

   this is a checkpoint whose satisfaction is a must attend event for the prosecution of the rest of the tasks.

3. The next step will consist in studying polymorphism for the types that are synthesized by the solver. For this purpose, the research will concentrate on annotating the input type and tracking dependencies between input and output variables so as to accurately detect copies between input and output types. An alternative but possibly equivalent research track consists in adding either natively or by an encoding, an arrow constructor to mu-calculus

expressions and use the solver to decide subtyping. It may be necessary to introduce type variables to mu-calculus (introducing quantifiers in formulae). The results of this research will be summarized in a research report that is expected to be ready by the month 21 of the project. All partners will contribute to this research. In particular LRI and PPS will bring the expertise accumulated at the first step while WAM will bring its expertise with the solver.

Deliverable 2.c: **(report)** Resolution of set of constraints by means of an extended logical solver (M21)

The accomplishment of the second and third steps will constitute a second milestone of the task. We will gauge their success by testing the formalism against common usage patterns of polymorphism in XML. In particular we will test whether it is possible to express the polymorphic functions needed to bridge Ocsigen and ℂDuce, as well as typical usage patterns of SOAP envelopes.

***Milestone 2.*II **(M24):** Polymorphic functions application.
Success criteria: can express SOAP patterns and Ocsigen/ℂDuce bridges.

4. Based on the previous result we will build on them the definition and implementation of an extension of the language ℂDuce with polymorphic functions. The gap between the theoretical/prototypical work and the implementation into a full-fledged language such as ℂDuce is important and we forecast that to fill it we will need at least one year of work. In particular there will be a work on the design of the language extension, the extension of the polymorphic typing and type inference to all the types and constructions of ℂDuce, the integration with the XSD validation present in ℂDuce, the extension of the interface with OCaml to deal with OCaml polymorphism, the implementation of the subtyping algorithm for polymorphic types, the reworking of static message errors and the heuristics they are based on, to account for errors in polymorphic function applications, as well as, for error generated by constraint solving.

   We think, optimistically, that a first prototype including many of the features described above will be available by month 30 of the project.

   Deliverable 2.d: **(prototype)** Design and implementation of the extension of ℂDuce with polymorphic functions (M30)

   The main contributors of this research will be the LRI and PPS with their past experience and expertise in the definition and implementation of the ℂDuce language. We will design these first extensions both in order to be backward compatible but above all so that they will not need code refactoring when a full implementation will be available. In this way we will be able to use immediately the prototype to program applications in production code.

   ***Milestone 2.*III **(M36):** Polymorphic ℂDuce.
   Success criteria: used in production code.

Task 3: Logically-driven implementation and certification of XML standards.

Leader: Matthieu Sozeau

Participants: P. Genevès, K. Nguyen, C. Paulin, A. Schmitt, M. Sozeau, J. Vouillon

Teams: LRI, PPS, WAM

This task aims at bridging a longstanding gap within the data-oriented XML community between theoretical studies of XML transformations and concrete implementations thereof. On the one hand, several theoretical formalisms have been studied, with great success, to reason about XML programs: μ-calculus, monadic second-order logic, tree automata, and so on. Thanks to these formalisms, many properties of XML queries (complexity, expressive power, ...) are now well understood. On the other hand, practical XML query engines heavily rely on ad-hoc optimizations, "programming" techniques, and heuristics. For these practical implementations, little can be said about their correctness or their complexity. The goal of this task is therefore to provide implementations of XML standards whose correctness is formally proved and whose efficiency is formally guaranteed. In particular we will tackle two different aspects of the problem, each aspect yielding a different track within this task:

Track 1. Tree automata based XQuery engine. Recent work, in particular the introduction of the SXSI query engine ([1, 43]), demonstrates that it is possible to provide an *efficient* XPath engine built on tree automata. However, SXSI only handles a fragment of XPath. This track aims at extending SXSI, first to a full-fledged XPath engine and then to a full-fledged XQuery engine. The challenging part here is the treatment of the backward axes with tree automata. It should be noted that this problem bears several similarities with the treatment of the so-called "inverse modalities" in the μ-calculus. We will therefore try to backport to the SXSI engine the techniques developed by the WAM group to handle backward axes in their solver. The contributors of this part will be the mainly the WAM group, which will bring its know-how on the implementation of the solver and the LRI group, in particular K. Nguyen, who will provide the most important research and implementation effort in this track, as a main developer of the SXSI engine. The expected results will be implemented in a prototype

Deliverable 3.a: **(prototype)** SXSI with backward axes (M21)

whose achievement will mark a milestone of this task:

Milestone 3.*I **(M21):* Handling of backward axes within SXSI.
Success criteria: SXSI passes the XMark and XPathMark correctness benchmarks.

Track 2. Formalization of XML related concepts in Coq. This track aims at developing tools to formally prove properties of XML programs in the Coq proof assistant. This involves formalizing parts of the standards (XPath in particular, building on top of prior work by Pierre Genevès [34]) but also formalizing models such as tree automata or the μ-calculus. The main contributors to this research will be M. Sozeau and J. Vouillon from PPS, K. Nguyen from LRI, and A. Schmitt and P. Genevès from WAM. This track is composed of two parts.

1. The first part of this track aims at delivering guarantees on SXSI's implementation. To do so, we will develop formal semantics for the different theories at issue: XPath and tree automata. We will build upon earlier work on XPath semantics in Coq by P. Genevès using it as a starting point for the development of a realistic XPath manipulation library. Concurrently, we will develop a new library for the manipulation of alternating tree automata as used in the SXSI engine and model the engine inside Coq.

   Deliverable 3.b: **(prototype)** Coq libraries for XPath and alternating tree automata manipulation (M15)

   Once these foundations are laid, we will attack the problem of proving a semantics preservation theorem for the compilation from the XPath queries to tree automata.

   The main contributors of this research will be the LRI and PPS with an active participation of the WAM group. LRI will bring the expertise of K. Nguyen on SXSI and tree automata, PPS will bring the expertise of M. Sozeau on Coq, while WAM will bring P. Genevès experience of XPath formalization. Once completed we will describe this novel formal proof of an efficient XPath engine in a deliverable report:

   Deliverable 3.c: **(report)** Formal proof of the semantics preservation theorem (M24).

   This proof will mark the second milestone of the task:

   ***Milestone 3.*II (M24):** SXSI's core functionality is verified.
   Success criteria: The proof models the SXSI engine faithfully.

   Based on the results from M3.I we will then adapt the automata definition and the proof to obtain a verified implementation of the SXSI engine with backward axes. The success of this part of the task heavily depends on the success of M3.I.

   ***Milestone 3.*III (M36):** SXSI with backward axes is verified.
   Success criteria: Completed proof of the SXSI engine with backward axes.

2. In the same vein, we want to investigate formal properties of the μ-calculus solver and consider its extension to richer calculi. After defining the logic and the solver's algorithm, we will prove the correction, completeness, and complexity of the solver with respect to the μ-calculus, relying on the existing paper proof.

   ***Milestone 3.*IV (M30):** The algorithm of the solver is verified.
   Success criteria: Completed proof of the algorithm's correctness, completeness, and complexity.

   The automated proofs will be made publicly available in a "report" deliverable:

   Deliverable 3.d: **(report)** Automated proofs for the solver (M30)

Building upon this formalization, we will extend the calculus to provide richer query features, such as the ability to write XPath queries that involve counting operators.

The main contributor to this research will be A. Schmitt from WAM relying on the PPS and LRI groups for Coq expertise.

Task 4: Reasoning about XML data and programs.

Leader: Kim Nguyen

Participants: V. Benzaken, G. Castagna, P. Genevès, N. Layaïda, K. Nguyen, C. Paulin, A. Schmitt, M. Sozeau

Teams: LRI, PPS, WAM

This task aims at enriching XML languages with various kinds of static analyses. While the use of type systems, solvers and proof assistants is widespread in the Programming Language community, it had so far little impact in the database realm, especially in XML data management engines. The challenge of this task is to adapt these techniques to the peculiarities of XML programming, helping developers to write safer, more maintainable, and more efficient code. This task will be organized in two tracks whose results will be merged in a final prototype deliverable. The timeline of each track is decomposed in two steps.

Track 1. Type aware query engine for efficient XML programming The work poured into the ℂDuce compiler has been a significant step forward to ensure the safety and efficiency of functional XML programs. It seems however a bit disappointing that more mainstream (or standardized) XML languages cannot benefit from such a precise typing discipline. The situation is even more unfortunate when one puts it in the light of the recent XQuery 3.0 draft ([22], whose latest revision dates from December 14, 2010). It is well known that prior to the introduction of updates (with the extension to XQuery 1.0 specified in [21]) XQuery was considered to be a *pure* language (if not *functional*), in the sense that there was no notion of *reference*: all operations and especially XML transformations are functional: building a new (XML) value from an existing one. Interestingly, this functional trait is reinforced in the XQuery 3.0 draft with several additions. First and foremost, *higher-order functions* are added to XQuery ([22] defines new constructs dubbed *dynamic function invocation* and *inline function definitions* which essentially allow the programmer to nest function definitions within one another and return functions as the result of computation, store them in data-structures or pass them as parameters to other functions). Second, a new *switch* construct is introduced, allowing one to define a multiple-conditional by case analysis on the shape of the argument, also known as *pattern-matching*. In this respect it seems only natural to see XQuery as a full fledged functional language and equip it with a strong static type system. The goal of this first track is to create an XQuery implementation equipped with the static type checking of ℂDuce

1. The first step consists in characterizing a functional core for the XQuery 3.0 proposal and its purpose will be twofold. First, this will allow us to focus on formal work while

ignoring, at first, the complexity of the full XQuery standard. Second, we believe that a small functional core for XQuery would be very close to ℂDuce and therefore facilitate the adaptation of ℂDuce's static type system to XQuery. It is important to stress that this preliminary work on XQuery 3.0 will be of general interest *per se*: when XQuery 1.0 was introduced, its so-called "Core" version was proposed to allow people to reason about XQuery programs without loosing any of its functionalities while reducing its verbosity and the various "crufts" of the W3C standard.

***Milestone 4.*i (M12):** A functional Core for XQuery 3.0.
Success criteria: The use cases for XQuery 3.0 programs can be translated in XQuery 3.0 Core.

2. The second step will be the definition of the proper type system. It should be noted that even if we focus just on a functional core for XQuery, at least one major challenge remains: XQuery uses XPath as querying paradigm while ℂDuce favors pattern-matching. However, the prior work on a query language for ℂDuce (it's ℂQL extension which augments ℂDuce with a SELECT-like operator), on tree iterators, and type projections ([47]) as well as the work of the WAM team on satisfiability of XPath queries in the presence of types, give high chances of success to porting ℂDuce's type system to XQuery.

The main contributors for this first research track will be the LRI members, who will bring their expertise both in XQuery/XPath and in declarative query languages (ℂQL) and PPS for the expertise in the ℂDuce type system.

***Deliverable 4.a:* (report)** XQuery equipped with ℂDuce's static type system. (M24)

The achievement of this track is marked by a milestone which will trigger the start of Task 5:

***Milestone 4.*ii (M24):** Type system for XQuery with higher-order functions and pattern matching.
Success criteria: Can type XMark's XQuery benchmarks and reproduce common usage patterns of ℂDuce higher-order functions and patterns.

Track 2. XQuery annotations and static verifications Not only does the XQuery 3.0 standard add new functional features, it also adds the interesting concept of *Annotations* and *Annotation assertions* (point 13 and 14 of the current working draft, [22]). As often with W3C standards only a few such assertions are specified by the standard, and implementations are free to provide their own in a custom namespace. Essentially, annotations are just textual information that decorate function or variable declarations. For instance, the standard adds a new annotation "*external*" that notifies the engine that the value of a variable is defined in a separate module. But one could define also semantic annotations, for instance, one could annotate a function definition with a logical property such as "*eg:non-null*". Here "*eg*" is simply a namespace telling where the "*non-null*" annotation is defined. Using this mechanism, our engine will allow the programmer to add such logical annotations and check them statically, for instance using SMT solvers or the μ-calculus solver. Again, this track is organized in two steps:

1. The first step is to cleanly define logical annotations and assertions of XQuery programs. We will re-use and adapt the work done by the WAM team on the μ-calculus solver, which can already solve a wealth of problems on XQuery programs (dead code detection, emptiness of queries, ...). We will thus check to which extent the μ-calculus solver can be used and extended to *automatically* verify that an XQuery program complies with its logical annotations. It is hard to define a measurable success criteria for such a work already for SQL—*a fortiori* for XQuery—since although the ISO SQL3 standard includes assertions, no commercial or academic engine implements it. So we will use as yardstick the examples used to explain assertions in the SQL3 standard whenever they can be transposed to the XML realm.

   ***Milestone 4.*III (M21):** Statically verified XQuery assertions.
   Success criteria: it is possible to transpose to XQuery and verify selected assertions used in the SQL3 standard definition.

   The achievement of this milestone, not only will make XQuery surpass the current query language technology of relational databases, but also trigger the work on Task 5.

2. The second step is to make the logic for annotations modular, in the sense that, these annotations and their verification could be offloaded to an external tool. Besides the μ-calculus solver, two other interesting choices come to mind: the first choice will be to use SMT solvers (like Alt-ergo or Z3 for instance). While they handle a logic weaker than the μ-calculus, they can handle other features—*eg*, arithmetic—- and have very good performances allowing us to check computation intensive programs. The second choice will be to use a richer (non-decidable) logic and generate proof obligations for a theorem prover such as Coq. The programmer would then have to provide a formal proof for the invariants at issue. We expect the proof generated this way to be the perfect test case and application for the Coq library on XPath developed as part of Task 3 (deliverable 3.b).

The main contributors to this second research track will be the WAM partner, who will provide expertise for the integration of logical annotations to XQuery and build and extend their prior work, the LRI partner, who will provide the know-how and development effort to integrate and adapt this work to XQuery as well the know-how about SMT solvers (in particular Alt-ergo) and the PPS partner for the generation of Coq proof obligations.

***Deliverable 4.b:* (report)** XQuery equipped semantic annotations (M30)

These two tracks will be developed in parallel but the final result for this task will be the merger of their results in a full fledged implementation to which all the participants of this task (actually, of the whole project) will contribute.

***Deliverable 4.c:* (prototype)** An XQuery engine (based on the SXSI runtime) equipped with a static type system and logical assertions, compatible with the XQuery 3.0 draft (or its evolution by that time thereof). (M36)

Task 5: Integration.

Leader: Giuseppe Castagna

Participants: All

Teams: LRI, PPS, WAM

This task will start as soon as (at least two of) the milestones M.2.II, M3.I, M.4.I and M.4.II will be satisfied (that is, once we can perform local inference for polymorphic applications, backward axes are implemented in SXSI, a type system for higher-order functions and pattern matching is available for XQuery, and a primitive form of assertion verification is implemented). The aim is to integrate the features proposed in XQuery 3.0 and developed in Task 4 with the polymorphic type systems of Task 2 and the reasoning and certifying capabilities of Task 3 and with a execution engine directly stemming from the work in Task 3.

We expect this task to yield a single paradigm. It will be directly issued from the XQuery core defined in the first track of Task 4 enriched with the annotation and static verifications of the second track of Task 3. Thus it will provide higher-order functions and pattern matching and by adapting the result of Task 2 these will have precise and polymorphic typing. The implementation will directly stem from the XML engine extended and certified in Task 3, thus providing an execution model whose correctness and complexity is formally proved.

This is a highly risky task, since its success depends on the complete success of the other three previous technical tasks. It will yield as deliverable (at least) a proposal for or (at best) a prototype of the next generation standard of XML transformation languages:

Deliverable 5.a: **(report)** Proposal for a standard XML transformation language with pattern matching, polymorphism, assertions, precise type checking, efficient certified implementation (M36)

While complete success may be difficult to achieve, partial success definitively seems within reach. Indeed also partial integrations make much sense here. In particular since we plan to use ℂDuce type system for higher order functions and pattern matching in Task 4 and we will polymorphically extend ℂDuce type system in Task 2, an integration limited to these three features does not seem very difficult. Similarly, the integration of the results of Task 3 (both track 1 on the XQuery engine, and track 2 on on the formalization in Coq) with those obtained in the track 2 of Task 4 (that is the addition of annotations and static verifications) seems mostly orthogonal and therefore should not pose much trouble. The same holds true for the track 1 of Task 3 (on the XQuery engine) and the track 2 of Task 2 (on the solver).

The outcome (even partial) of this task will yield the fundamentals for the standardization of next generation XML languages.

***Milestone 5.*I (M36):** Fundamentals of XQuery 4.0.
Success criteria: synthesizes the successes of all previous tasks.

3.3 Calendrier des tâches, livrables et jalons / Tasks schedule, deliverables and milestones

Calendrier des tâches / Tasks schedule

The overall time diagram is summarized in Table 2 (grayed parts denote no work on the task).

Month	3	6	9	12	15	18	21	24	27	30	33	36
Task 1	△	⊖	▽□	⊕	▽△	⊖		⊕	▽△	⊖	▽	◊
Task 2				D2.a D2.b	M2.I		D2.c	M2.II		D2.d		M2.III
Task 3					D3.b		M3.I D3.a	M3.II D3.c		M3.IV D3.d		M3.III
Task 4				M4.I			M4.III	M4.II D4.a		D4.b		D4.c
Task 5	▨	▨	▨	▨	▨	▨	▨					D5.a M1.I

⊖	progress report	□	consortium agreement
⊕	progress and expense report	△	project meeting
◊	final scientific and expense reports	▽	steering committee meeting

Table 2: Synthetic time diagram with deliverables (Dx.x) and milestones (Mx.x).

Récapitulatif des livrables / Deliverables summary

Deliverables are summarized in Table 3. We denote by **R** written reports (articles etc.), and by **P** prototypes or more generally, software components.

Présentation des jalons et des risques / Milestones and risks

***Milestone 2.*I (M15):** Polymorphic types for languages and solvers.(Success criteria: *higher-order transformations can be polymorphically typed with local inference*).

This is a convergence checkpoint for the results of step 1 and 2 of Task 1. Failure (that we deem unlikely) will trigger a lowering of the expectations: we will ask the user to provide more information via explicit annotations in order to help the type inference process. Any delay in this milestone will passed over the whole task.

***Milestone 2.*II (M24):** Polymorphic functions application.(Success criteria: *can express SOAP patterns and Ocsigen/*ℂDuce *bridges*).

In case of failure we will lower expectations by reducing the degree of local type inference.

***Milestone 2.*III (M36):** Polymorphic ℂDuce.(Success criteria: *used in production code*).

No	Title	Type	Date	Responsible
2.a	Constraint generation for inferring types in polymorphic function application with semantic subtyping	**R**	M12	PPS
2.b	Type synthesis for the logical solver	**R**	M12	WAM
2.c	Resolution of set of constraints by means of an extended logical solver	**R**	M21	WAM
2.d	Design and implementation of the extension of ℂDuce with polymorphic functions	**P**	M30	LRI
3.a	SXSI with backward axes	**P**	M21	LRI
3.b	Coq libraries for XPath and alternating tree automata manipulation	**P**	M15	PPS
3.c	Formal proof of SXSI's core	**R**	M24	PPS
3.d	Automated proofs for the solver	**R**	M30	WAM
4.a	XQuery equipped with static type system	**R**	M24	LRI
4.b	XQuery equipped with semantic annotations	**R**	M30	WAM
4.c	XQuery engine with static type system and logical assertions	**P**	M36	LRI
5.a	Proposal for a standard XML transformation language with pattern matching, polymorphism, assertions, precise type checking, efficient certified implementation	**R**	M36	PPS

Table 3: Detailed deliverables table.

This is one of the success criteria of the whole task and of the project in general. No possible countermeasure in case of failure.

***Milestone 3.*I (M21):** Handling of backward axes within SXSI.(Success criteria: *SXSI passes the XMark and XPathMark correctness benchmarks*).

Failure (or delay) of this milestone will not have any impact on the rest of program. It will only imply that the XML engine will not as efficient as we expected.

***Milestone 3.*II (M24):** SXSI's core functionality is verified.(Success criteria: *Completed proof of the semantics preservation theorem*).

This milestone has a high chance of success as parts of the proofs have been sketched in previous work. The main difficulty to be expected will be to find the appropriate formalization of the alternating tree automata. In case of failure we will lower expectations by restricting the XPath fragment to be handled.

***Milestone 3.*III (M36):** SXSI with backward axes is verified.(Success criteria: *Completed proof of the semantics preservation theorem for the updated formalization*).

This is the success criteria of a whole track of the task, no possible countermeasure in case of failure. This milestone directly depends on the success of M3.I.

***Milestone 3.*IV (M30):** The algorithm of the solver is verified.(Success criteria: *Completed proof of the algorithm's correctness, completeness, and complexity*).

This task is not on the critical path for the goal of the project, but is used to increase confidence in the satisfiability algorithm and propaedeutic for its extensions. We will not implement any countermeasure in case of failure.

***Milestone 4.*I (M12):** A functional Core for XQuery 3.0.(Success criteria: *The use cases for XQuery 3.0 programs can be translated in XQuery 3.0 Core*).

This milestone has a high chance of success. Indeed, XQuery Core has already been defined for the less powerful XQuery 1.0 specification. While it is necessary to update/enrich it to account for higher-order functions and pattern-matching, this will not pose any difficulty (several such languages already exists, including ℂDuce).

***Milestone 4.*II (M24):** Type system for XQuery with higher-order functions and pattern matching.(Success criteria: *Can type XMark's XQuery benchmarks and reproduce common usage patterns of* ℂDuce *higher-order functions and patterns*).

The key difficulty here, as we hinted in the description of Task 4, is the interaction between the type-system and backward axes. More precisely it is desirable to have a *precise* type-system in the presence of backward axes, which is quite difficult. In the worst case, we can resort to using the solution of the XQuery Static Semantic proposal, where backward axes are typed with the imprecise (but sound) type "*element*".

***Milestone 4.*III (M21):** Statically verified XQuery assertions.(Success criteria: *it is possible to transpose to XQuery and verify selected assertions used in the SQL3 standard definition*).

We said that it is quite difficult to establish measurable success criteria for this point since any achievement in this area will already be an advancement in current technology since even relational databases engines do not implement assertions. Similarly it is very difficult to anticipate failures: we will navigate at sight in this context.

***Milestone 5.*I (M36):** Fundamentals of XQuery 4.0.(Success criteria: *synthesizes the successes of all previous tasks*).

This is a highly risky milestone. The alternatives in case of (possibly partial) failure have been discussed in details in Task 5.

4 Stratégie de valorisation, de protection et d'exploitation des résultats / Dissemination and exploitation of results, intellectual property

4.1 Dissemination strategy

The success of the dissemination strategy, in particular the scientific community's acceptance and adoption of its ideas and developments (both theoretical and practical), as well as a direct and immediate impact on standardization processes will be the two main success indicators for our project.

Scientific communication. We aim at publishing our results in the main conferences of the domain (*eg*, POPL, ICFP, PLDI, PPDP, ESOP, VLDB, ICDE, ICSE, WWW, ...). Whenever possible we will propose to organize project meetings in coincident or as satellite events of important scientific venues (*eg*, ETAPS) with external invited speakers and open participation so as to increase visibility of our project and the impact of its results.

Technological impact and normalization/standardization processes. Our research aims at contributing at the rich interaction between academic research and industrial players, taking place within standardization bodies, in international technical meetings, and more generally in the worldwide electronic arena of those interested in XML technologies. In particular Task 2 aims to solve problems explicitly raised in RELAX NG and XQuery standards working groups, Task 4 specifically targets additions performed in the XQuery 3.0 Draft standard, while Task 5 specifically targets the standardization of the next generation XML languages.

The yardstick of the success of this work will the ability to come up with solutions that can deal with some representative use cases and be proposed for XQuery standardization and implementation effort. Members of the project have already co-authored and co-edited several W3C Recommendations, such as SMIL 1.0, 2.0 and 2.1 (Synchronized Multimedia Integration Language) within the SYMM (Synchronized Multimedia) working group. Using this experience, we intend to disseminate the results of this project through a direct participation in the appropriate World Wide Web Consortium Working Groups so has to have a ***direct impact on standardization processes within three years from the end of the project***.

Societal impact. Global data exchange is the main contribution of the Web. In the last few years, the eXtensible Markup Language (XML) has become the format of choice for data interchange on the Web. XML-based systems are now widely deployed in a number of application fields with a clear impact on many societal aspects, as witnessed by the steadily increasing importance of the so-called e-society (virtuality, mobility of the "connected" workers, social networking, e-learning, e-business, e-finance, e-marketing). Our project will provide theoretically grounded means to efficiently and safely process information stored and communicated in the XML format.

4.2 Intellectual property strategy and principles of exploitation of project results

Intellectual property (IP) rules will be laid down in a consortium agreement independently of this proposal (agreement protocol), once this proposal is accepted by ANR. The following IP principles will apply:

4.1.1 Definitions

The term Prior Knowledge designates all knowledge whether patented or not, know-how, trade secret or any other type of information in any format, belonging to or owned by one of the partners prior to the project, or acquired by the partner independently of the project's advancement, but needed for the project or for exploiting the project results. The term Own Results designates any knowledge whether patented or not, know-how, trade secret or any other type of information in any format, directly resulted from the project's work and obtained by the members of a single partner, without collaboration of another partner's members. The term Joint Results designates any knowledge whether patented or not, know-how, trade secret or any other type of information in any format, directly resulting from the project work and jointly obtained by members of two partners.

4.1.2 Intellectual property

Prior Knowledge: Each partner keeps complete and exclusive ownership of its Prior Knowledge. However, for the project's needs and duration alone, each partner may, upon explicit solicitation and under confidentiality agreements, use without financial counterparts the Prior Knowledge of another partner.

Own Results: Each partner will own its Own Results, that it will have developed alone, without intervention of members of another partner team, and will decide alone on the opportunity and the nature of protection measures to take, in its own name and at its own expense.

Joint Results: Joint Results will be the joint property of partners, which will decide together on the opportunity and the nature of protection measures to take, in their common name and at their shared expense. The partners will draw up a co-ownership agreement prior to any exploitation.

4.1.3 Exploitation principle

Exploitation of Prior Knowledge: Each partner will freely dispose of its Prior Knowledge. Each partner promises to provide to another partner, upon explicit request, licenses for its own Prior Knowledge needed for the exploitation of Own Results or Joint Results of the project, under preferential financial conditions.

Exploitation of Own Results: Each partner will freely dispose and use the Own Results of which it is an owner, directly or by licensing. Own Results of a partner needed to exploit the Own Results of another partner are subject to a license to negotiate, under preferential financial conditions.

Exploitation of Joint Results: Partners which jointly own Joint Results will agree on the exploitation details in the above-mentioned co-ownership agreement.

4.3 Exploitation structure and strategy

The three partners can rely on ad hoc structures and services for intellectual property management and exploitation.

In Paris Diderot - Paris 7 *Diderot Valorisation* is a structure directly attached to the presidency of the University. It is the interface between the research laboratories of the University and the socio-economic development sectors. It is responsible for protecting the interests of the University and its researchers for what concerns intellectual property: consortium agreements, patents, copyrights, software, know-how. It stimulates, directs and establishes partnerships with the private sector and industry (negotiations and partnership management, counseling as provided by the Innovation Act—Loi sur l'Innovation—). It identifies potential industrial interest in the results of research projects or it directs towards and assists in the creation of Young Innovative Companies (Jeunes Entreprises Innovantes) or on-site spin-off work.

In INRIA, the team REV (standing for "Relations Extérieures et Valorisation") is in charge of being the interface between the research center and its external partners (academic, industrial and institutional) for issues related to the fields of technology transfer and scientific communication. It provides support, advice, and help for legal aspects of contracts, intellectual property issues as well as issues with software distribution (APP deposits, patents, etc.). It also provides help for project leaders that want to create spin-offs; linking with support structures (incubators,etc.) and industrial research partnerships. For this purpose, the service works in conjunction with the entire center staff (researchers, services and the committee of projects), and in coordination with the INRIA management for the Development, Industrial Relations, Information Science, Communication and Financial Affairs.

In Paris Sud the *SAIC* (Service d'Action Industrielle et Commerciale) is in charge of protecting the interests of the University and its researchers for what concerns intellectual property: consortium agreements, patents, copyrights, software, know-how. It advices, stimulates, directs and help in establishing partnerships with the private sector and industry (as provided by the Innovation Act—Loi sur l'Innovation—). It identifies potential industrial interest in the results of research projects or it directs towards and assists in the creation of Young Innovative Companies (Jeunes Entreprises Innovantes) or on-site spin-off work.

5 Description du partenariat / Consortium description

5.1 Description, adéquation et complémentarité des partenaires / Partners description and relevance, complementarity

5.1.1 PPS

PPS is a leading research laboratory that federates mathematicians and computer scientists to work on programming languages, distributed systems, and their logical foundations. It has a internationally recognized expertize in XML programming languages having developed or co-developed reference languages such as XDuce and ℂDuce, in particular for what concerns XML type systems. The group actively contributes to the theoretical research around the formalism that underlies the Coq proof assistant, as well as to the development of/in Coq of which it is one of the main current contributors.

The PPS participants will bring to the project mainly their expertise and know-how about the programming-language oriented techniques: in particular, type systems, language design, and use and implementation of proof assistants.

5.1.2 LRI

The LRI partner gathers the data-centric languages and the proof of programs groups of ProVal, a project-team of the INRIA Saclay - Île-de-France research center, joint with LRI (CNRS and University of Paris Sud), located in Orsay. The former group has a long and recognized experience in developing frameworks adapted to reason about and efficiently implement query languages. It actively contributed, and still does, in the design, maintenance and further extensions of ℂDuce. The latter group is a world-wide recognized leader in the design, implementation and maintenance of the Coq proof assistant. The objective of the team is to propose methods and tools that can be integrated into the software development cycle and that make it possible to produce code that is proved to be correct with respect to its expected behavior. The team develops a generic program proof environment (the Why platform), that is able to generate proof demands that can then be delegated to automatic (*eg*, Alt-ergo) or interactive provers. Dedicated environments to prove C programs (Caduceus) and Java programs (Krakatoa) annotated with formulas describing the expected behavior, are constructed on top of this tool.

The LRI participants will bring to the project mainly their expertise and know-how about the data-oriented and verification techniques: in particular, the definition, implementation, and optimization of XML engines, of logical solvers, and the application of the latter to program verification.

5.1.3 WAM EPI

WAM is a project-team of the INRIA Grenoble - Rhône-Alpes research center, joint with LIG (CNRS and University of Grenoble), located in Montbonnot, France.

This group has a long and recognized experience in developing XML Document processing systems, automatic content adaptation by transformation and multimedia mobile document systems. The group actively contributes to the design and implementation of XML/XPath static analyzers with logical solvers, XML document processing tool-chains and document standards. The group is involved in the World Wide Web Consortium activities since 1995. Vincent Quint, the group leader, is member of the W3C Advisory Committee and former deputy director for Europe of the W3C. Nabil Layaïda is also a member of the W3C Synchronized Multimedia working group.

The WAM participants will bring to the project mainly their expertise and know-how about the standardization processes and the logic-oriented techniques: in particular, the definition and implementation of XML query engines, the static verification of program assertions and integrity constraints.

5.2 Qualification du coordinateur de la proposition de projet/ Qualification of the proposal coordinator

Giuseppe Castagna has past experience as project coordinator. He coordinated the RNTL project “GraphDUCE” (2 academic partners, 1 start-up company, 2003-2006) and the ACI project Tralala (5 academic partners 2004-2008). He was/is co-leader of a common project initiative LTL (with INRIA Sophia-Antipolis), of five bilateral international projects three with Italian universities, another with the University of Sussex and a fifth, recently started, with the Academy of Science of China. He was/is also site leader of a European research project (Myths), of an ANR project (Codex), and of an ACI project (Casc)

He has organizational experience also in scientific management organizations having served for four years as secretary of the AFIF (Association Française d’Informatique Fondamentale: French chapter of EATCS) and as the responsible of a thematic group in the GDR Programmation.

His scientific interests span over the whole spectrum of the proposal with contributions in programming languages, XML, and the application of PL techniques to databases. These contributions are recognized by the scientific community as witnessed by his experience as Program Chair of ESOP and joint keynote speaker of the conferences ICALP and PPDP (for programming languages), Program Chair, Steering committee member of Plan-X and invited speaker at XSym (for the XML part), invited speaker at the DBPL Symposium (for the database part), and finally as a member of the steering committee of ETAPS (for theoretical computer science in general). More generally, he served as chair or member of over 30 program, organization, or steering committees of international events among which the reference events of different research areas: programming languages (POPL, ESOP, FoSSaCS), object-oriented languages (ECOOP, OOPSLA), concurrency theory (CONCUR), XML language technologies (PLAN-X). Together with Joachim Parrow and Matthew Hennessy, Giuseppe Castagna formed the advisory board of the European project Mikado.

5.3 Qualification, rôle et implication des participants / Qualification and contribution of each partner

Partner	Name	First name	Position	Field of research	Person. month	Contribution to the proposal 4 lines max
PPS	Castagna	Giuseppe	DR CNRS	Informat.	24	Coordinator. Inference for polymorphic function application. Design and implementation of polymorphic extension of ℂDuce
PPS	Sozeau	Matthieu	CR INRIA	Informat.	9	Formalization in Coq, XQuery verification condition generation
PPS	Vouillon	Jérôme	CR CNRS	Informat.	3	Polymorphism and XML type systems. Formalization in Coq
PPS	Xu	Zhiwu	Doctorant	Informat.	12	Inference for polymorphic function application. Design and implementation of polymorphic extension of ℂDuce
PPS	Kende	Mathias	Doctorant	Informat.	18	Implementation of functional languages; parallelization and optimization
LRI	Benzaken	Véronique	Professeur	Informat.	18	Type aware query engines, XQuery annotation and verification
LRI	Nguyen	Kim	MdC	Informat.	18	Design and implementation of XQuery engine on top of SXSI
LRI	Paulin	Christine	Professeur	Informat.	3	Formalization in Coq, Why platform, deductive verification of programs
WAM	Layaïda	Nabil	CR INRIA	Informat.	12	Solver extensions for polymorphic types and types synthesis, Type-aware query evaluation
WAM	Genevès	Pierre	CR CNRS	Informat.	9	Solver extensions for polymorphic types and types synthesis, formalization in Coq
WAM	Schmitt	Alan	CR INRIA	Informat.	7	Logics for XPath satisfiability, type checking, formalization in Coq

6 Justification scientifique des moyens demandés / Scientific justification of requested ressources

General guidelines

Staff. The partners require 18 months of experienced postdoc each. The cost of an experienced postdoc varies according to (the administrative services of) each partner being 45588.96€ per year for WAM, 49000€ per year for PPS, and 53213.97€ per year for LRI. Although to conform to administrative rules we assigned a different postdoc to each partner we really consider the non permanent staff requests for this project as being mutualized to the whole project insofar as the requested staff will form the connecting frame between the different partners, by physical mobility, joint supervision, and collaboration. In particular we plan to operate resources reallocations if needed by possible reassignments of some months. As we explained in Section 2.1 we planned this project ahead and we are proposing several master student projects that correspond to the tasks in this proposal. We hope thus to attract excellent students on this topic and if necessary we would like to transform some months of post-doc into a PhD. grant. We envisage three different scenarios:

1. We do not need to fund any PhD. grant. In that case each site will have 18 months of post-doc to work as planned in the following sections, with possible mutually agreed readjustments between the partners to match contingency in the candidate selection.
2. One partner needs to fund a PhD. grant. In that case the other two partners will offer it 2 months of their postdoc. They will have the choice between funding 16 months of experienced post-doc or 17 months of entry-level post-doc
3. Two partners need to fund a PhD. grant. In that case the third partner will offer 7 months of its postdoc to the other two partners will be able to fund 12 months of entry-level post-doc (all charging off a small amount of the money for master projects).

Expenses other than staff. We used the following unified fees:

- *Trip to attend a project meeting or to visit partner*: 500 € per person (this amount was computed as an average between two days missions to attend project meetings and 3-4 days missions to cooperate with other partners)
- *Trip to a scientific event abroad*: 2250 € per person (this amount was computed as an average of costs between missions within Europe—2000 Euros—and outside Europe—2500 Euros—and includes an average cost of 600€ for inscription fees).
- *One week invitation* of an external scientist: 1400 €
- *One month of work placement gratuity* for a master student (indemnité de stage): 417€.[2]

[2]This gratuity is mandatory in France for master student projects that require at least two months of work. The amount is computed as 12,5% of the lowest legal hourly pay. This yields 417.09€ on 2011 January, the 1st.

- *Workstation or Laptop*: 1800 €
- *No coordination costs*: the specific costs of coordination will be assumed by the University of Paris Diderot - Paris 7.

6.1 Partenaire 1 / Partner 1 : PPS

Équipement / Equipment None.

Personnel / Staff We require 18 months of experienced postdoc. In the recruitment process we will give priority to cover the topics of Task 2 and, according to the timeline, Task 5. We will target a recruitement starting in the second or third semester of the project. In particular the post doc will constitute the main bridge between PPS and WAM research groups. The recruited postdoc will be required to already have or to rapidly acquire a double competence in advanced type systems and μ-calculus solvers. The person will be responsible to produce the deliverable 2.a, ensure the success of milestone 2.II, contribute to the production and success of deliverable 2.b and milestone 2.I, and, according to the time frame and contingencies, ensure or take part in the production of deliverables 2.d and 5.a. The post-doc will also be required to contribute to the supervision of master projects on the related topics (see later in this section). According to the profile of the candidates the position may also/alternatively be used to work on the proof assistant part assigned to PPS, in particular to produce deliverables 3.b and 3.c, contribute to the success of milestone 3.III and to the supervision of the master projects on Coq (see later in this section).

Total: 73500€

Prestation de service externe / Subcontracting None.

Missions / Travel We estimate that a PhD working full time on the project will need to finance two travels to attend a workshop/conference or a thematic school and the same for a permanent or postdoc researcher working half time on the project (including possible occasional participations in standardization working groups). So we include in our budget 1 travel for every 18 months of PhD provided and 1 travel for every 9 months of permanent or post-doc researcher provided. We round it in excess to account for a possible third PhD student which is not included in our costs. This yields a total of 9 travels (for the six or seven persons of the project in the three years). 2250€ $\times$ 9= 20250€.

We estimate that at least three persons of the project will attend one project meeting per year. 500€ $\times$ 9 = 4500€.

Finally we will need to invite external scientists for two weeks at least twice in the project. 1400€ $\times$ 4 = 4600€.

Total: 29350€

Dépenses justifiées sur facturation interne / Costs justified by internal invoices None.

Autres dépenses de fonctionnement / Other expenses The participation of the PPS site will consists of 5 or 6 persons: two permanent researchers, one post-doc, at least two PhD students with the possible addition of a third one if we successful recruit a student on the themes of this project. We estimate to need three computers for the three year period: one to equip the Post-doc, a second to renew one of the computers of the four persons that already belong to PPS, and a third to equip the new PhD student or to renew the existing ones: 1800€ × 3 = 5400€ for laptops or workstations

We also estimate the need of about 300€ per year for stationary and office items. 300€ × 3 = 900€

Finally we estimate to need 300€ for catering services to host project meetings.

We require 8 months of work placement gratuity per year. This will allow us to fund six four-month or four six-month master projects (according to the qualification of the candidate students: M1 or M2), centered on the programming languages part (subjects to be covered: (1) implementation of a type constraint solver, (2) implementation of a prototype polymorphic language for XML, (3) integration and implementation of polymorphic types with pattern matching for XML, (4) extension of ℂDuce by polymorphic functions, (5) extension of the interface between ℂDuce and OCaml to include (OCaml) polymorphic functions and polymorphic variant types) and on the Coq part (subjects to be covered: (1) implementation of the library for tree automata, (2) specification and proof of SXSI's compilation scheme): 417€ × 24 = 10008€.

Total: 17904€

6.2 Partenaire 2 / Partner 2 : WAM

Équipement / Equipment None.

Personnel / Staff We require 18 months of experienced postdoc. In the recruitment process we will give priority to cover the topics of Task 2 and, according to the timeline, Task 5. In particular the post doc will work on solver extensions for polymorphic types and types synthesis. The recruited postdoc will need to acquire competence in advanced type systems and μ-calculus solvers. The person will be responsible to produce the deliverable 2.b and 2.c, ensure the success of milestone 2.II, contribute to the success of milestone 2.I and 5.I. The post-doc will also be required to contribute to the supervision of master projects on the related topics (see later in this section) and to the definition of the a new XML transformation standard language (deliverable 5.a).

Total: 68383€

Prestation de service externe / Subcontracting None.

Missions / Travel First, we estimate that a postdoc and a PhD working full time on the project will need one travel each per 18 months to attend a workshop/conference or a thematic school (2250). Permanent researchers will need one travel per year for either an international conference, a demo or a W3C meeting for standardization ($3 \times 3 \times 3 \times 2500$ = 22500€). This yields a total of 10 travels (for the three permanent people and the postdoc of the project in the three years). 2250×10 = 22500€.

We estimate that at least three persons of the project will attend one project meeting per year. 500€$\times$ 9 = 4500€.

Finally we will need to invite external scientists for two weeks at least twice in the project. 1400€$\times$ 4 = 4600€.

Total: 31600€

Dépenses justifiées sur facturation interne / Costs justified by internal invoices None.

Autres dépenses de fonctionnement / Other expenses The participation of the WAM team site will consists in 4 or 5 persons: three permanent researchers, one post-doc and at least one PhD student. We estimate to need three computers for the three year period: one to equip the post-doc, a second to renew one of the computers of the three permanent persons, and a third to equip students or to renew the existing ones: 1800€$\times$ 3 = 5400€ for laptops or workstations

We need to fund 4 Master students for the entire project, for 6 months each: one centered on the solver type synthesis, one for higher order extensions for the solver, one for type inference, one on XQuery typing application and one for solver certification in Coq: 417×24 = 10008€.

We also estimate the need of about 500€ per year for stationary and office items. 500€$\times$ 3 = 1500€

Finally we estimate to need 300€ for catering services to host project meetings.

Total: 17208€

6.3 Partenaire 3 / Partner 3 : LRI

Équipement / Equipment None.

Personnel / Staff In the recruitment process we will give priority to cover the topics of Task 3, Task 4 and, according to the timeline, Task 5. We plan a recruitement starting in the second or third semester of the project. The recruited postdoc will be required to already have or to rapidly acquire competences in advanced type systems and tree automata theory, μ-calculus solvers and theorem provers. The candidate will be involved in the work on the deliverable 3.a, contribute to the success of milestone 4.I, ensure the production and success of deliverable 4.b and milestone 4.I and 4.II, and, according to the time frame and contingencies, ensure or take part in the production of deliverables 4.c and 5.a. The post-doc will also be required to contribute to the supervision of master projects on the related topics (see later in this section).

Total: 79821€

Prestation de service externe / Subcontracting None.

Missions / Travel We estimate that a PhD working full time on the project will need to finance two travels to attend a workshop/conference or a thematic school and the same for a permanent or postdoc researcher working half time on the project. So we include in our budget 1 travel for every 18 months of PhD provided and 1 travel for every 9 months of permanent or post-doc researcher provided. This yields a total of 7 travels (for the three or four persons of the project in the three years). 2250€$\times$ 7= 15750€.

We estimate that at least three persons of the project will attend one project meeting per year. 500€$\times$ 9 = 4500€.

Finally we will need to invite external scientists for two weeks at least twice in the project. 1400€$\times$ 4 = 4600€.

Total: 24850€

Dépenses justifiées sur facturation interne / Costs justified by internal invoices None.

Autres dépenses de fonctionnement / Other expenses We require 8 months of work placement gratuity per year. This will allow us to fund two master stages per year mainly corresponding to our involvement in Task 3 and Task 4. A first one will be related to the extension of SXSI with backward axis, a second will be centered on the definition of a functionnal core for XQuery, a third one will focus on the integration of ℂDuce type system in XQuery and last the fourth, fifth and sixth internships propositions will aim at equipping XQuery with annotations/assertions and study their static checking based on different solvers and in particular the WAM solver. Hence the total cost: 417€ $\times$ 24 = 10008€.

The participation of the LRI site will consists in 4 or 5 persons: three permanent researchers, one post-doc, and one PhD student. We estimate to need three computers for the three year period: one to equip the Post-doc, and two to renew the computers of the three persons that already belong to LRI: 1800€$\times$ 3 = 5400€ for laptops or workstations

We also estimate the need of about 500€ per year for stationary and office items. 500€$\times$ 3 = 1500€

Finally we estimate to need 300€ for catering services to host project meetings.

Total: 17208€

7 Annexes/Annexes

7.1 Références bibliographiques / References

[1] Diego Arroyuelo, Francisco Claude, Sebastian Maneth, Veli Mäkinen, Gonzalo Navarro, Kim Nguyen, Jouni Sirén, and Niko Välimäki. Fast in-memory xpath search using compressed indexes. In *ICDE*, pages 417–428, 2010.

[2] V. Balat. The Ocsigen: Typing Web Interaction with Objective Caml. ACM Sigplan Workshop on ML, 2006.

[3] Michael Benedikt and James Cheney. Destabilizers and independence of XML updates. *PVLDB*, 3(1):906–917, 2010.

[4] V. Benzaken, M. Burelle, and G. Castagna. Information flow security for XML transformations. In V. Saraswat, editor, *Eight Asian Computing Science Conference (ASIAN'03)*, Lecture Notes in Computer Science, Mumbai, India, 10-13 December 2003. Springer-Verlag.

[5] V. Benzaken and X. Schaefer. Static integrity constraint management in object-oriented database programming languages via predicate transformers. In Aksit and Matsuoka, editors, *European Conference on Object-Oriented Programming (ECOOP'97)*, number 1024 in Lecture Notes in Computer Science, pages 60–84. Springer-Verlag, 1997.

[6] V. Benzaken and X. Schaefer. Static management of integrity in object-oriented databases: Design and implementation. In Hans-Jorg Schek, editor, *6th International Conference on Extending Database Technology (EDBT'98)*, Valencia, Spain, March 23-27 1998. Springer Verlag.

[7] Gavin M. Bierman, Andrew D. Gordon, Catalin Hritcu, and David E. Langworthy. Semantic subtyping with an smt solver. In *ICFP*, pages 105–116, 2010.

[8] Henrik Björklund, Wouter Gelade, and Wim Martens. Incremental xpath evaluation. *ACM Trans. Database Syst.*, 35:29:1–29:43, October 2010.

[9] Loreto Bravo, Wenfei Fan, Floris Geerts, and Shuai Ma. Increasing the expressivity of conditional functional dependencies without extra complexity. In *ICDE*, pages 516–525. IEEE, 2008.

[10] Peter Buneman, James Cheney, and Stijn Vansummeren. On the expressiveness of implicit provenance in query and update languages. In Thomas Schwentick and Dan Suciu, editors, *ICDT*, volume 4353 of *Lecture Notes in Computer Science*, pages 209–223. Springer, 2007.

[11] Peter Buneman, Sanjeev Khanna, and Wang Chiew Tan. Why and where: A characterization of data provenance. In Jan Van den Bussche and Victor Vianu, editors, *ICDT*, volume 1973 of *Lecture Notes in Computer Science*, pages 316–330. Springer, 2001.

[12] Peter Buneman and Wang Chiew Tan. Provenance in databases. In Chee Yong Chan, Beng Chin Ooi, and Aoying Zhou, editors, *SIGMOD Conference*, pages 1171–1173. ACM, 2007.

[13] Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Moshe Y. Vardi. An automata-theoretic approach to regular xpath. In Philippa Gardner and Floris Geerts, editors, *DBPL*, volume 5708 of *Lecture Notes in Computer Science*, pages 18–35. Springer, 2009.

[14] Diego Calvanese, Giuseppe De Giacomo, Maurizio Lenzerini, and Moshe Y. Vardi. Node selection query languages for trees. In Maria Fox and David Poole, editors, *AAAI*. AAAI Press, 2010.

[15] The Castor project. `http://www.castor.org/`.

[16] G. Castagna and Z. Xu. Set-theoretic foundation of parametric polymorphism and subtyping. Submitted. Type-checker prototype available at `http://www.pps.jussieu.fr/~zhiwu/files/subtype-checker-0.3.tar.gz`, July 2010.

[17] James Cheney. Satisfiability algorithms for conjunctive queries over trees. In *To appear, ICDT*, 2011.

[18] J. Clark and M. Murata. RELAX NG, 2001. `http://www.relaxng.org`.

[19] H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi. Tree automata techniques and applications. Available at: `http://www.grappa.univ-lille3.fr/tata`, 1997. Release October, the 1st 2002.

[20] Denise Draper et al. XQuery 1.0 and XPath 2.0 Formal Semantics, 2007. `http://www.w3.org/TR/query-semantics/`.

[21] Don Chmaberlin et al. Xquery update facility, 2009. `http://www.w3.org/TR/xquery-update-10/`.

[22] Johnathan Robie et al. Xquery 3.0: An XML query language (working draft 2010/12/14), 2010. `http://www.w3.org/TR/xquery-30/`.

[23] M. Benedikt et. al. Report on the EDBT/ICDT 2010 Workshop on Updates in XML. *SIGMOD Record*, 39(1), 2010.

[24] T. Berners-Lee et al. *Uniform Resource Identifier*, January 2005. RFC 3986, STD 66.

[25] Wenfei Fan and Jérôme Siméon. Integrity constraints for XML. *J. Comput. Syst. Sci.*, 66(1):254–291, 2003.

[26] A. Frisch. Regular tree language recognition with static information. In *Proc. IFIP Conf. on Theor. Comput. Sci. (TCS)*. Kleuwer, 2004.

[27] A. Frisch, G. Castagna, and V. Benzaken. Semantic Subtyping. In *LICS '02, 17th Annual IEEE Symposium on Logic in Computer Science*, pages 137–146. IEEE Computer Society Press, 2002.

[28] A. Frisch, G. Castagna, and V. Benzaken. Semantic subtyping: dealing set-theoretically with function, union, intersection, and negation types. *The Journal of ACM*, 55(4):1–64, 2008.

[29] Galax: The xquery implementation. `http://www.galaxquery.org/`.

[30] V. Gapeyev, M. Y. Levin, B. C. Pierce, and A. Schmitt. The Xtatic experience. In *PLAN-X*, 2005.

[31] Ph. Gardner, Gareth Smith, Mark Wheelhouse, and Uri Zarfaty. Local hoare reasoning about DOM. In *ACM Principles of Database Systems (PODS)*, 2008.

[32] Pierre Genevès and Nabil Layaida. Deciding XPath containment with MSO. *Data & Knowledge Engineering*, 63(1):108–136, October 2007.

[33] Pierre Genevès, Nabil Layaïda, and Alan Schmitt. Efficient static analysis of XML paths and types. In *PLDI '07: Proceedings of the ACM SIGPLAN Conference on Programming Language Design and Implementation*, pages 342–351, 2007.

[34] Pierre Genevès and Jean-Yves Vion-Dury. XPath formal semantics and beyond: A Coq-based approach. In *TPHOLs '04: Emerging Trends Proceedings of the 17th International Conference on Theorem Proving in Higher Order Logics*, pages 181–198, Salt Lake City, Utah, United States, August 2004. University Of Utah.

[35] H. Hosoya. *Regular Expression Types for XML.* PhD thesis, University of Tokyo, 2000.

[36] H. Hosoya, A. Frisch, and G. Castagna. Parametric polymorphism for XML. In *POPL '05, 32nd ACM Symposium on Principles of Programming Languages*. ACM Press, 2005.

[37] H. Hosoya, A. Frisch, and G. Castagna. Parametric polymorphism for XML. *ACM Transactions on Programming Languages and Systems*, 32(1):1–56, 2009.

[38] H. Hosoya and B.C. Pierce. Regular expression pattern matching for XML. In *POPL '01*, 2001.

[39] Haruo Hosoya, Jérôme Vouillon, and Benjamin C. Pierce. Regular expression types for XML. *ACM Trans. Program. Lang. Syst.*, 27(1):46–90, 2005. Short version in ICFP 2000.

[40] The jaxb api. `http://java.sun.com/developer/technicalArticles/WebServices/jaxb/`.

[41] Xavier Leroy. A formally verified compiler back-end. *J. Autom. Reasoning*, 43(4):363–446, 2009.

[42] Gregory Malecha, Greg Morrisett, Avraham Shinnar, and Ryan Wisnesky. Toward a verified relational database management system. In *ACM International Conference on Principles of Programming Languages (POPL*, 2010.

[43] Sebastian Maneth and Kim Nguyen. Xpath whole query optimization. *PVLDB*, 3(1):882–893, 2010.

[44] Maarten Marx. Conditional XPath, the first order complete XPath dialect. In *PODS '04: Proceedings of the twenty-third ACM Symposium on Principles of Database Systems*, pages 13–22, New York, NY, USA, 2004. ACM Press.

[45] The MonetDB database server. `http://monetdb.cwi.nl/MonetDB/index.html`.

[46] Frank Neven and Thomas Schwentick. Query automata. In *Proceedings of the eighteenth ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems*, PODS '99, pages 205–214, New York, NY, USA, 1999. ACM.

[47] K. Nguyen. *Language of Combinators for XML: conception, typing, implementation*. PhD thesis, Université de Paris Sud 11, 2008.

[48] The Ocsigen Project. `http://www.ocsigen.org/`.

[49] Francois Pottier and Vincent Simonet. Information flow inference for ML. In *Symposium on Principles of Programming Languages*, pages 319–330, 2002.

[50] David Spelt and Herman Balsters. Automatic verification of transactions on an object-oriented database. In Sophie Cluet and Richard Hull, editors, *DBPL*, volume 1369 of *Lecture Notes in Computer Science*, pages 396–412. Springer, 1997.

[51] David Spelt and Susan Even. A theorem prover-based analysis tool for object-oriented databases. In Rance Cleaveland, editor, *TACAS*, volume 1579 of *Lecture Notes in Computer Science*, pages 375–389. Springer, 1999.

[52] D. Volpano and G. Smith. A type-based approach to program security. *Lecture Notes in Computer Science*, 1214:607–621, 1997.

[53] Dennis Volpano, Geoffrey Smith, and Cynthia Irvine. A sound type system for secure flow analysis. *Journal of Computer Security*, 4(3):167–187, 1996.

[54] W3C. *SOAP Version 1.2*. `http://www.w3.org/TR/soap`.

7.2 Biographies / CV, resume

7.2.1 Giuseppe Castagna (44 years)

University Curriculum: Laurea in Scienze dell'Informazione, University of Pisa (1990); Master in Computer Science, University Paris 7 (1990); PhD. in Computer Science, University Paris 7 (1994); Habilitation à diriger des recherches, University Paris 7 (2002).

Positions: CNRS Researcher (CR) École Normale Supérieure de Paris (1994-2006); CNRS Senior Researcher (DR) University Paris 7 (2006-present).

Résumé: After having started and then lead for five years the “Programming Languages” group in École Normale Supérieure, in fall 2006 Giuseppe Castagna moved to the PPS Laboratory in University Paris 7 where he has CNRS Senior Researcher position. Secretary of AFIF (Association Française d’Informatique Fondamentale: French chapter of EATCS: 2003-2007), member of the Conseil National des Universités (1999-2003), member of recruiting committees, reviewer for the Italian Ministry of Scientific Research and the Natural Sciences and Engineering Research Council of Canada, member of the advisory board of a European project, reviewer for several French national research organisations. Proposer and leader of several European national and international research programs and of business development and technological transfer actions. Invited speaker in 9 international conferences. Chair or member of over 30 program, organizations, or steering committees of international events. Supervisor of 11 PhD. theses (8 defended of which 3 were awarded by prizes). Lecturer of international tutorials, international doctoral schools and 6 master courses. Castagna’s main contributions are in the domain of object-oriented languages, concurrent mobile calculi, transformation languages for XML documents and foundation of web-services.

Publications: Coeditor of several journal special issues (Information and Computation, Theoretical Computer Science, Logical Methods in Computer Science), of proceedings (ESOP ’09, PLAN-X ’06), author of a book (Birkhäuser - Boston - 379 page), of two book chapters and of 55 articles in international journals and refereed conferences. Bibliometry: over 3100 citations recorded, excluding self-citations; h-index = 29 : 29 publications with at least 29 citations each; g-index = 55 : the most cited 55 publications reach a total over 3025 (=55^2) citations. A selection of five publications in the last five years:

1. G. Castagna, N. Gesbert, and L. Padovani. A theory of contracts for web services. *ACM Transactions on Programming Languages and Systems*, 31(5):1–61, 2009.
2. H. Hosoya, A. Frisch, and G. Castagna. Parametric polymorphism for XML. *ACM Transactions on Programming Languages and Systems*, 32(1):1–56, 2009.
3. G. Castagna, R. De Nicola, and D. Varacca. Semantic subtyping for the π-calculus. *Theoretical Computer Science*, 398(1-3):217–242, 2008.
4. A. Frisch, G. Castagna, and V. Benzaken. Semantic subtyping: dealing set-theoretically with function, union, intersection, and negation types. *Journal of the ACM*, 55(4):1–64, 2008.
5. G. Castagna and K. Nguyen. Typed iterators for XML. In *ICFP ’08: 13th ACM-SIGPLAN International Conference on Functional Programming*, pages 15–26, April 2008.

7.2.2 Matthieu Sozeau (28)

University curriculum: Master 2 Recherche Informatique, Université Paris 7 (2005); PhD. in Computer Science, University Paris 11 (2008).

Positions: Post-doctoral fellow at Harvard University, Cambridge, USA, with Greg Morrisett (2009-2010); Junior researcher at INRIA, $\pi.r^2$ Project, Paris (CR2) since October 2010.

Résumé: Matthieu Sozeau is a junior researcher at INRIA since october 2010, in the $\pi.r^2$ team developing the Coq proof assistant. He defended his PhD thesis at Université Paris-Sud in the INRIA ProVal team under the supervision of Christine Paulin in 2008. He was a post-doctoral researcher at Harvard University between March 2009 and October 2010, under the supervision of Greg Morrisett.

Matthieu's research is centered around programming languages and interactive theorem proving. He is one of the main developers of the Coq proof assistant, a state-of-the-art theorem prover developed at INRIA. He designed during his Ph.D. thesis a new programming language to ease verification of strongly specified programs, integrated concepts such as type classes in the underlying programming language and developed automated reasoning tools to simplify proofs. He has a good knowledge of XML processing, having developed XPath and XSLT engines in the past.

Publications: In both fields (Programming Languages and Theorem Proving), Matthieu Sozeau has published his research in top-tier conferences: ICFP and TPHOLs (both ranked A according to the Core classification). His software contributions have been widely adopted by Coq users and he also served on the program committees of TLDI and ICFP.

1. Matthieu Sozeau and Nicolas Oury. First-Class Type Classes. *Theorem Proving in Higher Order Logics, 21th International Conference*, LNCS 5170:278–293. 2008.
2. Matthieu Sozeau. Equations: A Dependent Pattern-Matching Compiler. *First International Conference on Interactive Theorem Proving*. 2010.
3. Matthieu Sozeau. A New Look at Generalized Rewriting in Type Theory. *Journal of Formalized Reasoning*, 2 (1):41–62, 2009.
4. Matthieu Sozeau. Program-ing Finger Trees in Coq. In *ICFP'07: Proceedings of the 2007 ACM SIGPLAN International Conference on Functional Programming,* pp.13–24, 2007.
5. Matthieu Sozeau. Subset Coercions in Coq. *TYPES'06*. LNCS 4502:237–252, 2007.

7.2.3 Nabil Layaïda (42)

University curriculum: DEA in computer science and applied maths, Université Joseph Fourier (1993); PhD. in Computer Science, Université Joseph Fourier (1997).

Positions: Research Engineer Bull S.A. (1993). Postdoc researcher at Ottawa University-Carleton (1997–1998), Canada; Experienced Researcher at INRIA Rhône-Alpes, WAM Project, (October 1998–present); W3C Invited expert (1996–2007); Science and Technology Advisor RaisePartner S.A. (2004–2010)

Résumé: Nabil Layaïda is a research scientist in the WAM EPI at INRIA Grenoble Rhône-Alpes since 1998. From 1996 to 2007, he was member of the SYMM Working group of the World Wide Web where he co-authored and co-edited SMIL 1.0, 2.0 et 2.1 W3C recommendations. From 2004 to 2010, he co-founded RaisePartner S.A., an INRIA spin-off where he was the science and technology advisor. He is also member of the scientists employment committee of INRIA Grenoble Rhône-Alpes since 2007.

His main research activity is the design of safe and efficient adaptive document processing infrastructures. In particular, he contributed to the design of context-aware document processing and adaptation systems and infrastructures for mobile devices. He is also interested in structured documents; query and transformation languages, the verification and optimization of document manipulating-programming languages and incremental document processing. More recently, he contributed to the design of logical reasoning solvers for finite trees together with exact XML type-checkers.

Publications: Nabil Layaïda has published over 50 papers in international conferences and journals, including ICFP, PLDI, WWW, IJCAI, ACMMM and TOIS.

1. Pierre Genevès and Nabil Layaïda and Vincent Quint. Impact of XML Schema Evolution. ACM Transactions on Internet Technology (to appear, 2011).

2. Nabil Layaïda and Pierre Genevès. Debugging standard document formats. In WWW '10: Proceedings of the 19th International Conference on World Wide Web.

3. Pierre Genevès, Nabil Layaïda and V. Quint. Identifying query incompatibilities with evolving XML schemas. ICFP'09: In proceeding of the 14th ACM SIGPLAN international conference on Functional programming.

4. Pierre Genevès, Nabil Layaïda, and Alan Schmitt. Efficient Static Analysis of XML Paths and Types. In PLDI '07: Proceedings of the 2007 ACM SIGPLAN conference on Programming language design and implementation, 2007.

5. Pierre Genevès and Nabil Layaïda. A system for the static analysis of XPath. ACM Transactions on Informations Systems, 24(4): 475-502 (2006).

7.2.4 Alan Schmitt (36)

University curriculum: DEA in semantics, proofs and programming, Université Paris 7 (1999); PhD. in Computer Science, École Polytechnique (2002).

Positions: Postdoc researcher at University of Pennsylvania, Philadelphia, with Benjamin Pierce (2002–2004); Researcher at INRIA Rhône-Alpes, Sardes Project, Experienced researcher (CR1) since January 2005 (2004–present); Sabbatical at University of Bologna, Italy, with Davide Sangiorgi (2007–2008).

Résumé: Alan Schmitt is a research scientist in the Sardes project-team at INRIA Grenoble Rhône-Alpes since 2004. He defended his PhD thesis at École Polytechnique in 2002, supervised by Jean-Jacques Lévy from the Moscova EPI at INRIA Rocquencourt. From October 2002 to January 2004, he was a post-doctoral researcher at University of Pennsylvania under the supervision of Benjamin Pierce.

Alan Schmitt's research is centered on static analyses for the manipulation of structured data. This includes type systems for message-oriented middleware, domain-specific language design for bidirectional manipulation of tree-structured data and text, type systems for pattern matching XML data, satisfiability checking of XPath queries, and development of observational equivalences for higher-order process calculi. More recently, he has been learning to use the Coq proof assistant to formalize these static analyses.

Alan is currently leading the PiCoq ANR project, whose goal is to develop tools and libraries to use the Coq proof assistant to formalize results on higher-order process calculi. He has lead the BACON associated team between the Sardes project team and University of Bologna. He is a steering member of the French conference on functional languages JFLA.

Publications: Alan Schmitt has published 22 papers in international conferences and journals, including POPL, PLDI, LICS, ICALP, JCSS, TOPLAS, and I&C.

1. J. Nathan Foster, Michael B. Greenwald, Jonathan T. Moore, Benjamin C. Pierce, and Alan Schmitt. Combinators for bidirectional tree transformations: A linguistic approach to the view-update problem. ACM Transactions on Programming Languages and Systems, 29(3):17, 2007.

2. Pierre Genevès, Nabil Layaïda, and Alan Schmitt. Efficient Static Analysis of XML Paths and Types. In PLDI '07: Proceedings of the 2007 ACM SIGPLAN conference on Programming language design and implementation, 2007.

3. Aaron Bohannon, J. Nathan Foster, Benjamin C. Pierce, Alexandre Pilkiewicz, and Alan Schmitt. Boomerang: Resourceful Lenses for String Data. In ACM SIGPLAN-SIGACT Symposium on Principles of Programming Languages (POPL'08), 2008.

4. Ivan Lanese, Jorge A. Pérez, Davide Sangiorgi, and Alan Schmitt. On the Expressiveness and Decidability of Higher-Order Process Calculi. In Proceedings of the 23rd Annual IEEE Symposium on Logic in Computer Science (LICS 2008), 2008. Long version to appear in Information & Computation.

5. Ivan Lanese, Jorge A. Pérez, Davide Sangiorgi, and Alan Schmitt. On the Expressiveness of Polyadic and Synchronous Communication in Higher-Order Process Calculi. In Proceedings of the 37th International Colloquium on Automata, Languages and Programming (ICALP 2010), 2010.

7.2.5 Pierre Genevès (30)

University curriculum: DEA and Magistère in Computer Science, Université Joseph Fourier (2003); PhD. in Computer Science, Institut National Polytechnique de Grenoble (2006).

Positions: Founder and Managing Director at Pierresoft.com (since 1999). Research Engineer at IBM T.J. Watson Research Center, New York, USA (2003,2004). Research Assistant at INRIA Rhône-Alpes (2003-2006). Postdoctoral researcher at École Polytechnique Fédérale de Lausanne, Switzerland (2007). Researcher at CNRS (since 2007). Scientific Advisor at RaisePartner (2009-2010).

Résumé: Pierre Genevès is a CNRS research scientist in the WAM EPI at INRIA Grenoble Rhône-Alpes since 2007. Previously, he was a postdoctoral researcher at Ecole Polytechnique Fédérale de Lausanne and a research officer at IBM T.J. Watson Research Center. Before that, Pierre operated a self-made small software company in the field of image editing. He has been advisor for a few startups since then.

His recent research works focused on developing foundations, methods and tools for the safe and efficient manipulation of structured information, like XML data. Specifically, he focused on the design and implementation of logics as type systems for query languages over tree structures, with published applications in programming languages (PLDI, ICFP), web (TOIT, WWW), databases (TOIS, ICDE), etc.

In 2007, he obtained the EADS Foundation Prize (Information Sciences) and the Prize for the best PhD thesis of the INPG University. He was also Finalist for the ERCIM Cor Baayen award in 2008 and was granted an IBM Invention Achievement Award in 2004.

Publications: Pierre Genevès' publications include venues in international conferences and journals such as PLDI, ICFP, WWW, ICDE, ICSE, TOIS and TOIT.

1. Pierre Genevès and Nabil Layaïda and Vincent Quint. Impact of XML Schema Evolution. ACM Transactions on Internet Technology (to appear, 2011).

2. Pierre Genevès. Logics for XML: Reasoning with Trees. ISBN 3639193717, VDM Verlag, 2009.

3. Pierre Genevès, Nabil Layaïda and V. Quint. Identifying query incompatibilities with evolving XML schemas. In ICFP'09: In proceeding of the 14th ACM SIGPLAN international conference on Functional programming, 2009.

4. Pierre Genevès, Nabil Layaïda, and Alan Schmitt. Efficient Static Analysis of XML Paths and Types. In PLDI '07: Proceedings of the 2007 ACM SIGPLAN conference on Programming language design and implementation, 2007.

5. Pierre Genevès and Nabil Layaïda. A system for the static analysis of XPath. ACM Transactions on Informations Systems, 24(4): 475-502 (2006).

7.2.6 Véronique Benzaken (50)

University curriculum: D.E.A. d'Informatique Fondamentale, Université Paris 7 (1986); PhD. in Computer Science, University Paris 11 (1990); Habilitation à diriger des recherches - Université Paris 11 (1996).

Positions: Engineer in an industrial company (1983-1985). Associate Professor, Université Paris I Sorbonne (1990-1998) Full professor, Université Paris 11 (1998-present).

Résumé: Véronique Benzaken is full professor at Université de Paris Sud since 1998. From 1998 until 2003 she has been heading the school of computer science and management (IUP - Miage) of the computer science department. She has been part of Paris Sud's scientific board (conseil scientifique), of the computer science department's board and she is currently a member of the recruiting comittee of the computer science department. As national services, Prof., Benzaken served as a member of the "Comité Technique Paritaire Universitaire", she served in many junior researchers recruiting comittees for INRIA and she also participates, as a scientific expert, in AERES's evalution comittees. She has been reviewer for different international research agencies among them NWO (the dutch national research agency).

Prof., Benzaken's research activity mainly focus on the application of static analysis to data related problems and is at the confluence of programming languages and data management. More specifically, she is interested in the foundational design and implementation of data-centric languages. In this line she developped the Thémis framework based on asbtract interpretaion combined with first order theorem proving to certify object-oriented transaction with respect to integrity constraints (which are nothing but programs invariants). More recently, Véronique Benzaken actively participated in the foundation and design of ℂDuce an XML-centric functionnal language part of major Linux distributions. She supervised ten PhD thesis.

Professor Benzaken frequently serves as a program comittee member in fisrt venue international (database) conferences (VLDB, ICDE, XSYM ...).

Publications: Véronique Benzaken's major publications include venues such as JACM, LICS, ACM-ICFP, ACM-PPPDP, PADL, ECOOP for her contributions in the programming languages field and venues such as VLDB-Journal, ACM-SIGMOD, ICDT, ICDE, EDBT or VLDB for the data management part of her research. She is author or co-author of about 60 peer-reviewed publications totalizing 912 citations. A selection of five publications in the last five years

1. V. Benzaken, G. Castagna, H. Hosoya, B.C. Pierce, and S. Vansummeren. *The Encyclopedia of Database Systems*, chapter “XML Typechecking”. Springer, 2009.

2. Véronique Benzaken, Giuseppe Castagna, and Alain Frisch. Semantic subtyping: Dealing set-theoretically with function, union, intersection, and negation types. *Journal of the ACM*, 55(4):1–64, 2008.

3. V. Benzaken, G.Castagna, D. Colazzo, and C. Miachon. Pattern by example: Type-driven visual programming of XML queries". In *10th International ACM SIGPLAN Symposium on Principles and Practice of Declarative Programming*, 2008

4. A. Arion, V. Benzaken, I. Manolescu, and Y. Papakonstantinu. Structured materialized views for XML queries. In *33rd International Conference on Very Large Databases (VLDB 2007)*, September 2007.

5. V. Benzaken, G.Castagna, D. Colazzo, and K. Nguyen. Type-based XML projection. In *32nd International Conference on Very Large Databases (VLDB 2006)*, September 2006.

7.2.7 Kim Nguyen (29)

University curriculum D.E.A Sémantique, Preuves, Programmes Université Paris-Sud 11 (2003-2004, Mention Bien); PhD. in Computer Science, University Paris 11 (2008).

Positions Post-doctoral researcher at National ICT Australia, NRL Lab Sydney (2008-2010); Associate Professor at Université Paris-Sud 11 (2010-present).

Résumé: Kim Nguyen is an assistant professor at Université Paris-Sud since September 2010, in the ProVal team. He defended his PhD thesis at Université Paris-Sud, in the Database Team under the joint supervision of Véronique Benzaken and Giuseppe Castagna in 2008. He was a post-doctoral researcher at National ICT Australia between October 2008 and June 2010, under the supervision of Sebastian Maneth.

Kim's research is at the intersection of programming languages and databases, with primary focus on XML processing. He designed during his Ph.D. thesis a language of combinators for XML, granting languages such as CDuce the ability to express highly polymorphic XML transformations and yet type them precisely, a clear contribution to the state of the art. During his post-doctoral research he was developped (jointly with S. Maneth and others) SXSI, an efficient XML XPath engine based an tree automata theory and state of the art succinct indexes. He was also the lead developper within the SXSI project.

In both fields (Programming Languages and Databases), Kim Nguyen has published his research in top-tier conferences and journals: ICFP, VLDB, ICDE and VLDB Journal (all ranked A+ or A* according to the Core classification). He his also the co-author of several patent proposals for XML querying.

publications:

1. D. Arroyuelo, F. Claude, S. Maneth, V. Mäkinen, G. Navarro, K. Nguyen, J. Siren, N. Välimäki. *Fast In-Memory XPath Search using Compressed Indexes*, ICDE 2010, 417 - 428
2. Sebastian Maneth, Kim Nguyen. *XPath Whole Query Optimization*. PVLDB 3(1): 882-893 (2010)
3. Giuseppe Castagna, Kim Nguyen. *Typed iterators for XML*. ICFP 2008: 15-26
4. Véronique Benzaken, Giuseppe Castagna, Dario Colazzo, Kim Nguyen. *Type-Based XML Projection*. VLDB 2006: 271-282
5. Sebastian Maneth, Kim Nguyen and National ICT Australia Limited, *Query Processing of Tree-structured data*, Australian patent number 2010900322.

7.3 Implication des personnes dans d'autres contrats / Staff involvment in other contracts

The scientific leaders of the three partners as well as two other members of this proposal have been taking part in a common ANR project, the Codex project, that will end approximatively in concomitance with the expected kick-off of Typex. Therefore it seems important to write at least two paragraphs to explain why this proposal *is not* a continuation of Codex.

The aim of Codex project has been to offer an efficient large-scale infrastructure for XML processing: (1) distributed XML repositories, (2) parallel XML processing, (3) data and schema evolution, (4) interoperability and composition. None of the first three goals is covered in Typex, nor the results obtained there will constitute a starting point or will be reused in Typex. The fourth topic shares with Typex some high level *goals* and *tools*: *goals* since one of the way to check composition is to use types, but in Codex the great bulk of the work on types is formed by behavioural types based on process algebras that describe the observable behaviour of distributed processes (they are, thus, well far away from the types considered here); *tools* since to test our results on service composition we have embedded ℂDuce into Ocsigen (by while in Codex ℂDuceis a tool, in Typex it constitute a centra subject of research).

The goal of the Typex project is to build a new generation of programming languages with stronger formal foundations. Its originality is twofold: first, Typex aims at making explicit and mechanically proved the tasks traditionally hidden in compilers and runtimes; second, it aims at boosting languages and compilers by equipping them with necessary reasoning capabilities needed in the evaluation of XML transformation, in their complex typing processes, and in the specification and verification of expected behaviour. In summary, the separation line between Typex and Codex is quite demarcated since the former essentially aims to contribute at a programming language level, while the latter has beeing aiming to formalize and build an infrastrustructure for massively distributed and remotely interacting XML repositories.

Part.	Name	Person. month	Project name, financing institution, grant allocated	Project title	Coordinator name	Start and end dates
1	G. Castagna	16 [1]	Programme Domaines Emergents (DEFIS), ANR, 110490€	Codex	I. Manolescu	03/2009 02/2012
2	G. Castagna	==	National Science Foundation of China, 320000RMB (∼36000€)	Unranked tree-structured data and semantically defined polymorphic type systems	G. Castagna and H. Chen	2010 2013
3	P. Genevès	7.2 [0]	Programme Domaines Emergents (DEFIS), ANR, 83539€	Codex	I. Manolescu	03/2009 02/2012
4	N. Layaïda	7.2 [0]	Programme Domaines Emergents (DEFIS), ANR, 83539€	Codex	I. Manolescu	03/2009 02/2012
5	V. Benzaken	6 [0]	Programme Domaines Emergents (DEFIS), ANR, 101088€	Codex	I. Manolescu	03/2009 02/2012
6	V. Benzaken	7.2 [0]	Digiteo Ediflow grant no. 2008-44D	EdiFlow	I. Manolescu	09/2008 12/2011
7	A. Schmitt	3.6 [0]	Programme Domaines Emergents (DEFIS), ANR, 83539€	Codex	I. Manolescu	03/2009 02/2012
8	A. Schmitt	38.4 [31]	Programme Blanc, ANR, 195322€	PiCoq	A. Schmitt	01/2011 12/2014
9	J. Vouillon	10 [5]	Programme Domaines Emergents (DEFIS), ANR, 170872€	PWD	M. Serrano	01/2010 12/2013
10	C. Paulin	12 [1]	Programme SESUR, ANR, 81432 €	SCALP	Y. Lakhnech	01/2008 12/2011

NOTA BENE: In column "Person.month" we indicated the total involvement on the whole length of the project, and between square brackets the *residual* involvement in September 2011, the expected starting date of Typex.

Table 4: Staff involvement in other contracts.