ST_YX : Connecting the XML Web to the World of Semantics

Irini Fundulaki[1], Bernd Amann[1], Catriel Beeri[2], Michel Scholl[1], and Anne-Marie Vercoustre[3]

[1] Cedric-CNAM Paris, INRIA Rocquencourt
irini.fundulaki@inria.fr, amann@cnam.fr, scholl@cnam.fr
[2] Hebrew University, Israel
beeri@cs.huji.ac.il
[3] INRIA-Rocquencourt; currently on leave at CSIRO Mathematical and Information Sciences, Melbourne, Australia,
Anne-Marie.Vercoustre@cmis.csiro.au

1 Introduction

The ST_YX prototype illustrates a new way to publish and query XML resources on the Web. It has been developed as part of the C-Web (Community Web) project[1] whose main objective is to support the sharing, integration and retrieval of information in *Web communities* concerning a *specific domain of interest*.

ST_YX is based on a simple but nevertheless powerful model for publishing and querying XML resources [1]. It implements a set of *Web-based* tools for creating and exploiting *semantic portals* to XML Web resources. The main functions of such a portal can be summarized as follows: (i) XML resources can be *published/unpublished* on the fly; (ii) *structured queries* can be formulated by taking into consideration the conceptual representation of a specific domain in form of an *ontology*; (iii) query results can be customized for display and further exploitation.

The backbone of a ST_YX portal is a *domain specific ontology* comprised of *concepts* and *roles* describing the basic notions in the domain. Our approach [1] takes advantage of the presence of XML Document Type Definitions (DTDs) that capture the structure of XML documents. An XML source is published in a ST_YX portal by a set of *mapping rules* that map XML *document fragments*[2] specified by XPath[3] location paths to *ontology paths*. The user queries the set of the sources by formulating simple queries in terms of ontology paths. The objective is to be able to forward user queries to diverse XML repositories while hiding their structural heterogeneity to the end-user.

As mentioned previously a source is published by *mapping rules* that associate XPath location paths to ontology paths. Consider for example the XML resource *http://www.art.com* which stores XML documents about painters and

[1] http://cweb.inria.fr
[2] XML Fragment Interchange: http://www.w3c.org/TR/xml-fragment.html
[3] XML Path language (XPath) : http://www.w3c.org/TR/xpath.html

their paintings. The first two rules illustrated below associate `PAINTER` elements and their `NAME` attribute to instances of concept Person and role has_name. The other two rules map `PAINTING` sub-elements and their `TITLE` attribute to instances of concept Artifact reached by the role produced and role has_title respectively.

R_1: `http://www.art.com//Painter` as u_1 ← Person
R_2: u_1`/@Name` ← has_name
R_3: u_1`/Painting` as u_2 ← produced
R_4: u_2`/@Title` ← has_title

A user query, formulated in terms of ontology paths, will be rewritten into a set of XML queries that will be send to the XML resources for evaluation. For example, the simple user query Q illustrated below which requests the *"names of persons and the title of their artifacts"*, will be rewritten to the XQuery[4] Q' using the mapping rules and the latter will be send to the XML resources for evaluation (the rewriting algorithm is presented in detail in [1]). The results obtained by each source are then unioned at the mediator site before they are returned to the user.

Q: **select** b, d
from Person a,
a.has_name b,
a.produced c,
c.has_title d

Q': **FOR** \$$a$ **IN**
document("http://www.art.com//Painter")
\$$b$ **IN** \$$a$/@Name,
\$$c$ **IN** \$$a$/Painting
\$$d$ **IN** \$$c$/@Title
RETURN \$$b$, \$$d$

2 *ST$_Y$X* Portal Architecture

The architecture of the system is presented in Figure 1. XML Web resources can be published on the fly by creating/modifying/deleting mapping rules between resource fragments and the ontology using the **Source Publication Interface**. The ontology can be consulted through the **Schema Manager** which is also responsible for its loading in a *ST$_Y$X* portal. The established mapping rules are first validated by the **Rules Manager** which is also responsible for their storage. The publication of a resource also consists in providing an XSL Stylesheet[5] that can be used for formatting source data in the query result.

Query Processing is done in several steps: first *user queries* can be formulated using a standard Web browser. They are either created by a generic **Query Interface**, an HTML form or simply be stored in form of a hypertext link (URL). The **Query Interface** communicates with the **Schema Manager** allowing the user to browse the ontology for the formulation of a query. Second, the **Query Interface** forwards the query to the **Query Parser** which performs a syntactical analysis of the query and does some type-checking w.r.t. the ontology. It produces a language neutral intermediate representation of the query,

[4] XQuery : An XML Query Language (http://www.w3.org/TR/xquery/).
[5] XSL Transformations (XSLT : http://www.w3c.org/TR/xslt)

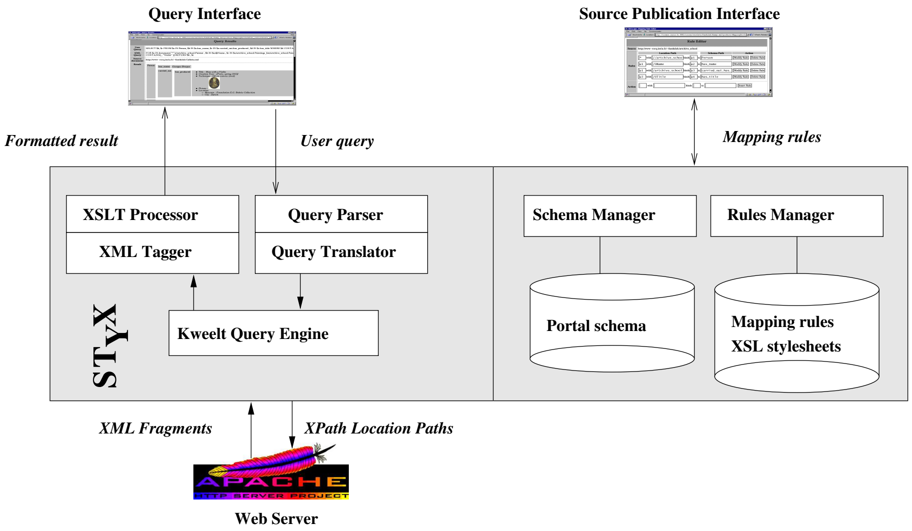

Fig. 1. *STYX* Portal Architecture

which is then forwarded to the **Query Translator**. Third, based on the mapping rules and the ontology, the **Query Translator** generates for the query a *set* of *XML queries*. Last, XML queries are then evaluated using the **Kweelt query engine**[6]. Finally the results obtained by the source are reformatted before returned to the user : the **XML Tagger** inserts schema specific tags using the mapping rules and then the **XSLT Processor** (Cocoon[7]) finally transforms the result into an HTML document which is displayed to the browser of the user.

One of our objectives in the development of *STYX* was to follow and to exploit as much as possible standard XML technologies and recommendations such as XQuery, XPath and XSLT. It shows once more that XML is not only a flexible format for data exchange but has become a mature technology for building Web portals.

Demonstration Summary : During the system demonstration we will illustrate (i) the publication of cultural XML Web resources using the Source Publication Interface, (ii) the formulation of user queries through the Query Interface and (iii) the evaluation of user queries over the published XML resources.

References

1. B. Amann, C. Beeri, I. Fundulaki, M. Scholl, and A-M. Vercoustre. Mapping XML Fragments to Community Web Ontologies. Presented at the Fourth International Workshop on Web and Databases WebDB, Santa Barbara California, May 2001. http://cedric.cnam.fr/PUBLIS/RC255.pdf.

[6] Kweelt Query Engine : http://db.cis.upenn.edu/Kweelt/

[7] http://xml.apache.org/cocoon