Enhancing Course Reusability through XML Schemas Integration

Aida Boukottaya

Media Research Group
Swiss Federal Institute of Technology (EPFL)
In Ecublens, 1015 Lausanne, Switzerland
aida.boukottaya @ epfl.ch

ABSTRACT. Due to the extensive use of XML markup language in the education domain, there has been a great interest on proposing rich data models (DTDs or recently XML Schemas) that reflects course semantics. The existence of such rich schemas has made a large amount of heterogeneously structured courses widely available in distributed repositories. In this framework, sharing and reusing structured courses by several applications and users is of major concern. In this paper, we present the FAVORITE project, which proposes and describes a novel strategy to integrate heterogeneous XML Schemas. Two main aspects are considered in our approach. Firstly, integration aims to facilitate access to multiple distributed structured courses by providing global views that abstract and unify the original course sources. Secondly, integration also facilitates the task of adaptation, through conflict resolution techniques, which are responsible for extracting relevant semantic and structural information.

KEYWORDS: structured courses, reusability, schemas integration.

RÉSUMÉ. L'utilisation croissante du langage XML dans le domaine de l’éducation est à la base de la conception de modèles de documents élaborés, que ce soit via des DTD ou, plus récemment via des schémas XML, visant à refléter la structure sémantique de contenus pédagogiques. L’existence de tels modèles a favorisé la production et la disponibilité de telles ressources, toutefois structurées de manière hétérogène. Dans un tel contexte, partager et réutiliser des ressources existantes s’avère d’un intérêt incontestable, à la fois en termes d’interopérabilité entre les applications et entre les différents auteurs. Le projet FAVORITE décrit notre approche basée sur l’intégration de schémas XML en vue de répondre à deux besoins essentiels en termes de réutilisabilité : l’accès aux données pédagogiques et leur adaptation à un contexte spécifique. D’une part, l’accès aux données est procuré au travers d’un schéma unifié ; d’autre part, sur cette base, nous facilitons leur adaptation à un nouveau contexte en dégageant les contraintes structurelles et sémantiques susceptibles d’être source de conflits.

MOTS-CLÉS : cours structurés, réutilisation, intégration de schémas.
1. Introduction

Due to the extensive use of XML markup language (XML 98) in the education domain, there has been a great interest in proposing rich data models also called schemas (DTDs or recently XMLSchemas (XMLSchema01)) that reflect course structure and semantics (Teege 02), (Stuhrenberg 01), (Wiest et al., 2001).

The growing use of such models opens attractive perspectives: educational documents become dynamic and rich components of distributed information systems. They act as processable pieces of data that integrate a semantic dimension, captured through using adequate element names such as introduction, chapter, and exercise, the incorporation of meta data descriptions such as course type, author, owner, title, format, etc., and the organization of element and their combination within the document. In this framework, sharing and reusing structured courses by several applications and users is of major concern.

Currently, the research community has started to focus on pedagogical materials reuse and sharing. The idea is to provide collective shared databases for pedagogical materials. The goal is to reuse those materials by large number of authors and teachers when creating new interactive courses. ARIADNE described in (ARIADNE 00) and SEMUSDI described in (Delestre et al., 1998) (SEMUSDI) are two European research projects aiming to provide distributed shared databases for pedagogical material.

FAVORITE project (Favorite02) –‘Fragmenting and Authoring Virtual Object-based documents for Reuse in Interactive Teaching Environments’– addresses the challenge of reusability of structured educational materials in distributed learning environments. The specificity of the FAVORITE project comparing to previously mentioned environments is to give a major interest to the logical structure which greatly facilitates the reuse of content. Optimizing course production by facilitating the reusability of structured courses is a challenging task. In fact, finding potentially reusable education material in distributed environment and reconciling data conforming to different rich schemas is not an easy task.

Such task generally requires a prohibitive effort to first make structured courses accessible to users and second adapt these courses to users’ context. In this vision, the key point is the mediation between these heterogeneous course models. As a consequence, complex mappings and reasoning about those mappings are necessary for comparing and combining those models, and for integrating data described using different schemas. Integration aims first to facilitate access to multiple distributed structured courses by providing global views that abstract and unify the original course sources. And second to facilitate the task of adaptation through conflict resolution techniques, which are responsible for extracting semantic and structural information needed to perform adaptation between schemas.

The paper is organized in the following way. The second section briefly summarizes the limits of current approaches to support reuse of pedagogical
material. Section 3 points the requirements of good integration strategy, and describes the global architecture adopted to integrate heterogeneous XML Schemas. Section 4 details the main aspects of the integration process. Finally, we conclude the paper with a discussion and future work.

2. Educational material reuse: problems and challenges

Currently, the research community has started to focus on pedagogical material reuse and sharing. In this context, two key issues have to be addressed: accessibility and adaptation. The first concern is defined as the identification of relevant and potentially reusable educational material. The second is defined as the ease with which it is adapted to teacher context and needs.

To implement reuse, two major techniques have been proposed, by-reference or by-value. By-reference means that the same educational resource is referenced from two or more documents without transformation or adaptation. By-value means that the educational resource is taken from one document and reused in another one. In the latter case, the reused resource needs to be transformed to fit its new context (Garzotto et al., 1995) (Paolini et al., 1993). Since the FAVORITE project deals with providing collective shared repositories for structured pedagogical materials to assist a large number of authors and teachers when creating new courses, we consider the by-value reuse method.

In the last few years, several methods have been proposed to address the problem of educational resources’ reusability, but none answer both requirements of accessibility and adaptation. These approaches can be classified into two categories. One solution lies in using metadata to enhance accessibility. Another is to structure educational resources’ content in such a way that it facilitates reuse and adaptation. The FAVORITE project addresses both requirement of accessibility and adaptation. Our approach is designed especially for teachers who need to save time and effort in putting together structured courses in order to produce new material for teaching.

2.1. Educational material reuse by means of metadata

One solution to enhance reusability lies in using metadata. The idea behind using metadata in education is to segment existing course material (e.g., slides, text books, animations, videos) into so called learning objects, each learning object represents a self-contained unit, which is appropriately annotated with metadata.

A lot of effort is devoted to the definition of educational metadata. Three metadata standards are emerging to describe educational resources: IEEE LOM (LOM 00), ARIADNE (ARIADNE 00) and IMS (IMS 00). Those meta-models define how learning materials can be described in an interoperable way. All the metadata elements necessary to describe a resource can be classified into several
categories, each offering a distinct view on a resource. For example the LOM allows one to describe such items as type of the object, author, owner, title, format, etc. Moreover, LOM also includes pedagogical attributes, such as teaching or interaction style, difficulty, typical learning time and prerequisites. Since no information is included in this standard on how to represent metadata in a machine-readable format, the IMS Global Learning Consortium developed a representation of LOM in XML.

Using this approach, projects like (Caumanns00), (Seeberg et al., 1999), (Vrabic et al., 2001), (Universal), (Delestre et al., 1998) and (Hollfelder et al., 2001), intend to achieve a high degree of reusability.

Reusability seen as a matter of metadata for learning objects presents several limitations. First, it addresses only the problem of accessibility. In fact, good annotation makes easier the task of finding adequate resources. Second, even with good metadata available, reuse works in practice only for entities of rather low granularity, such as images (Teege 02). Third, the availability of several metadata standards and vocabularies raises the problem of incompatibility between disparate metadata descriptions across communities. Finally, most of those metadata standards lack a formal semantics, which can be found in the content itself.

2.2. Structuring educational content

It is commonly accepted that availability of structured educational documents greatly facilitates the processibility of content. In this respect, a number of research works are dedicated to the analysis of raw or semi-structured educational documents in order to structure or re-structure them to facilitate their reuse through existing applications. The key motivation behind structuring educational material is to decouple style from content. Storing course contents in elements that reflect semantics (e.g. introduction, chapter, exercise) allows the reuse of data in different contexts according to user needs. (Wiest et al., 2001), demonstrates how authors (teachers) benefit from the use of structured documents:

–Authors can concentrate on content not on layout;
–Structural integrity can be easily checked;
–Indexes, summaries, glossaries etc. can be generated automatically;
–Content modules can be reused in other contexts.

There are many implementations of XML DTDs for structuring educational content (IMS 00), (LTSC), (EML). In (Wiest et al., 2001), they enrich an XML-compliant, lightweight variant of the DocBook DTD with additional elements and attributes that cater educational needs, different media types and hyperlink capabilities. A DTD and an XSD were designed in the framework of the Milca project (Stuhrenberg 01) to structure the content of learning objects. Targetteam project (Teege 02) identifies *Abstractness* as an additional requirement for
reusability. This is achieved by employing an appropriate XML Schema, which is called TeachML.

Structuring courses facilitates their accessibility. In fact, based on the structure, indexes and summaries can be generated automatically to accelerate access to structured courses.

In majority of these works adaptation is seen as a matter of layout, which is generally achieved by predefined stylesheets. The Milca project argues that storing the content of a Learning Object as XML data allows for an easy transformation into several output formats, such as HTML for online browsing or the Portable Document Format (PDF) for offline viewing. In Targeteam project, adaptation does not occur at the level of layout, adaptation means modifications on the content level. This is done by the means of predefined Stylesheets written in XSLT. XSLT is a language, itself written in XML, encoding any transformation of XML documents.

One of the important limitations of these methods is that the course content is defined with respect to one predefined schema. This makes the task of adaptation easy. However, the content of lectures are diverse, and different instructors can give the same course over and over again with different contents and structures. Thus, we cannot conform the lecture content to a rigid predefined schema. In this context, adaptation become difficult and the investment required is generally prohibitive. The usual procedure requires an analysis of both the semantics and the structures of the source and target course models. Manual coding then follows this analysis. A partial step towards in reducing the effort in such procedure is to use special software to assist the writing of the XSLT. These tools are very useful in describing and carrying out the process of transformation. However, they still require developers to manually indicate mappings for each source and target pair. What is significantly lacking is to semi-automatically discover such mappings based on schema matching techniques and so generate automatically the transform script.

3. XML Schema integration strategy to enhance reusability of structured courses

In order to solve problems and limitations of current methods to enhance both accessibility and adaptation, we argue in favour of a multi-layered integration strategy. Our approach facilitates access to structured courses by abstracting and unifying the original data sources into a global view. Adaptation is achieved by automatic view generation conforming to user needs in term of content and structure.

Like most schema integration approaches, we decompose integration into a multi-layered architecture pre-integration (where input schemas are transformed to make them more homogenous (both syntactically and semantically), comparison (devoted to the identification and description of interschema relationships) and
integration (which is the final step that solves interschema conflicts and unifies corresponding items into an integrated schema) (Batini et al., 1986).

In this section, we first present a short survey of some of the techniques used for integrating heterogeneous data. Second, we state the requirements to build an integrated global schema according to our proposed strategy.

3.1. Approaches for Data integration

Reconciling distributed data in different formats from autonomous heterogeneous sources is not a new problem. It has been a major focus in the last two decades by the database community (Abiteboul et al., 1999), (Bergamaschi et al., 1999), (Castano et al., 1999), (Haas et al., 1999), (Levy et al., 1996), (Mena et al., 1996), (Papazoglou et al., 1998). Many methods provide integrated access to structured data sources (like in relational and object-oriented databases) (Wiederhold 93). Others, support integration of both structured and semi-structured data (Papakonstantinou 99).

Semi-structured data arise in many application areas. The emergence of XML further increases the availability of semi-structured data. See (Abiteboul 97) for an excellent survey on semi-structured data. In this section, we present a short survey of some of the techniques for integrating semi-structured data.

An important distinction in building semi-structured data integration systems is whether to take a warehousing or a virtual approach. In the warehousing approach, data imported from multiple sources is materialized into a centralized repository (DIXSE system) (Rodriguez 01), and the queries are applied to the warehoused data. The main advantage of data warehousing is the performance in answering queries (since the queries are applied to the warehoused data). However this approach requires that the warehouse be updated when data changes, which is not appropriate when handling a large number of sources or when the sources frequently change.

In the virtual approach instead, the data remains in the original sources, and only a logical view is provided. Queries to the data integration system are decomposed at run time into queries on sources. Virtual approach is more appropriate for handling a large number of sources or when the sources change frequently. For these reasons, most of the recent research in semi-structured data integration has focused on the virtual approach. Data integration in these systems is done using logical views, and applying query-rewriting techniques that translate queries posed to the views into specific source queries. TSIMMIS (Chawathe et al., 1994), Lore (McHugh et al., 1997), and MIX (Bars et al., 1999) are examples of these systems. A framework for supporting integrated views using a combination of these approaches is also possible (Hull et al., 1996).

Two major points can differentiate systems that relay on virtual approach. The first one consists of the data model used to represent the global view. The second
concern the mapping techniques to define correspondences between views at global level and data sources. Broadly speaking, two general approaches have been proposed: Global As Views (GAV) (Papakonstantinou 96) and Local As Views (LAV) (Duschka et al., 1997). In the first approach (GAV), the mediated schema is defined in term of sources’ schemas. While in the LAV approach, the descriptions of the sources are given in the opposite direction. The main advantage of the GAV approach is that query reformulation is very simple. However, adding sources to the mediated schema is non-trivial. In contrast, in the LAV approach, query reformulation is harder but adding new sources is quite straightforward. (Friedman 99) describes the GLAV language that combines the expressive power of GAV and LAV to integrate web resources.

Concerning XML data integration, a lot of work has been recently proposed (Castano et al., 2001), (Mello et al., 2001). Computation of similarity between different DTDs involves schema-matching techniques. Schema matching is studied mostly in Entity Relationship models (Kashyap et al., 1996), (Larson et al., 1989), (Milo et al., 1998), (Bernstein 01). In contrast, research in schema matching for XML DTDs is just gaining momentum. It is handled in some systems through machine learning techniques (Doan et al., 2001). In other systems element- and structure-level matching is performed such as in Cupid (Madhavan et al., 2001), SPL (Su et al., 2001) and Xclust (Lee et al., 2002).

3.2. Requirements of XML schema integration strategy

Recently, most of the XML based integration systems deal with XML DTDs. However, DTDs have some limitations in modelling information resources. One important restriction of DTDs is, that they are poor in terms of types (e.g. user defined types are not provided) and in terms of conceptual abstractions (Batini et al., 1992) (generalization / specialization is not supported). Due to the rich content of educational material, DTDs are no more sufficient and are replaced by XML Schema standard (XMLSchema01). In fact, XML Schemas add user defined types and allow generalization / specialization by means of substitution groups and subtyping mechanisms. For this, we propose a novel integration strategy to unify XML schemas that describe courses content.

The first requirement we state for our integration strategy is to deal with the syntactic, the semantic and the structural dimension of XML schemas. Syntactic dimension concerns the terms used to declare elements, types, etc. XML Schema allows a partial inference of the semantics of data they describe through cardinality constraints, type inheritance, substitution groups and namespaces. Finally aggregation relationships describe the structural dimension.

The second requirement concerns the global schema also called mediated schema. The global schema has to meet the three criteria described in (Batini et al.,
Completeness, minimality, and understandability. In order to meet the first, all the element in the source schemas should be present in the global schema. Minimality means that each element should be defined only once within the global schema (e.g. avoid redundancy). Finally, to satisfy the third criteria, understandability, the global schema should be formulated in such a way that supports easy browsing and querying.

3.3. FAVORITE project architecture

The FAVORITE project focuses in the virtual approach to integrate heterogeneous XML schemas describing teaching materials. Our global architecture consists of three layers, outlined in figure 1.

![Schema integration architecture](image)

**Figure 1. Schema integration architecture**
The data layer consists of several distributed educational databases (also called educational pools). In each pool, structured courses (XML documents) are stored conforming to specific data models (XML Schemas). We distinguish between two kinds of databases:

– **Native XML Databases (NXD):** Databases that store XML in native form, thus maintaining the full XML structure and associated metadata. Data access is implemented through XML and related standards only, e.g. XPath, DOM, SAX, or XSLT. Sample products are dbXML (dbXML), Virtuoso (Virtuoso), Tamino (Tamino).

– **XML-Enabled Databases (XEDB):** Databases with extensions for transferring data between XML documents and themselves. Examples are Informix (Informix), SQL Server 2000 (SQLServer), Oracle 8/9i (Oracle). In this context, XML is not stored in its native form and additional work has to be done in order to convert XML documents into internal database structure.

The integration layer deals with solving heterogeneities between XML schemas. The aim of the integration process is the development of a global schema which integrates and subsumes the local schemas in such a way that users are provided with a uniform and correct view of the global data. For this, many algorithms and matching techniques are required. This layer is the main focus of this paper, and will be detailed in next section.

Once a global schema is provided, browsing and querying techniques, as well as query-rewriting techniques that translate queries formulated against the global schema into specific source queries are needed. This aspect concerns the application layer.

4. XML Schema integration process

The integration process is the set of activities required to unify schema sources into a global schema. By studying the process and analysing different approaches from the literature we observed a set of commonalities and assembled them into five modules described in figure 1. These modules describe the phases that we consider fundamental and distinct in the integration process:

– **Abstractness:** This module focuses on raising all schemas to be integrated onto the same representation level, coping with syntactical heterogeneity;

– **Semantic modeling:** This module is concerned with the real world view and understanding of data. It typically includes only significant “entities” which have a meaning in the application domain, along with their “relationships”. A semantic model represents the information independently of all physical considerations (implementation details).

These two modules perform schemas’ pre-processing in order to facilitate the integration. We believe that this is a critical step that affects the quality of the
generated global schema. Our goal is to capture all XML Schemas’ semantic generally inferred by typing, type inheritance, substitution groups, namespaces and element cardinality.

–Comparison: This module focuses on analysing and comparing schemas to determine the correspondences among concepts and detect possible conflicts.

–Conflict resolution: Once conflicts are detected, this module focuses on resolving such conflicts by providing adequate functions and techniques.

–Merging and restructuring: The result of this module is a global schema unifying heterogeneous schema sources. The global schema has to meet the three criteria of Completeness, minimality, and understandability mentioned in section 3.2.

4.1. Abstractness

The XML Schema standard has been criticized for its complexity, and several attempts at a simpler and more formal description are underway (Brown et al., 2001), (Ludascher et al., 2002). XML Schema complexity is due to the manner of declaring schema information and not to the actual schema meta-model, which is simple and straightforward.

To make XML Schemas more homogenous syntactically, we propose an abstract Model for XML Schemas, which is an attempt to simplify and organize XML Schema declarations.

In this phase, we do not aim at a complete formalization of all details of XML Schema, but rather at capturing the essential features required for semantic modelling performed by the next module. The advantage to use an abstract model is twofold. First, our approach of “abstract” representation is not limited to XML Schema but could also be based on other type aware XML schema languages. Second, on the contrary to other formal descriptions, it is written with XML syntax, which enables us to benefit from XML technologies to manipulate it.

An abstract XML Schema is composed of disjoint three sets: E, SG, and T (declared in this order) of element declaration, substitution group declaration, and type declaration.

To each element of E, is associated a unique type in T. We do not allow inlining element declaration (e.g. sequence declaration within element declaration). figure 2 describes an example of XML schema rewritten according to our abstract model. The inlining sequence declaration in course1.xsd is transformed into a complex type in course2.xsd.
4.2. Semantic modeling

Before comparing and integrating heterogeneous schemas, the usual procedure is to convert first these schemas into a canonical conceptual model that makes schemas more homogenous at a semantic level. The canonical conceptual model is concerned with the real world view and understanding of data. It typically includes only significant “concepts” which have a meaning in the application domain, along with their “relationships”. A conceptual model represents the schema information independently of any implementation considerations.

In the context of XML sources integration, the canonical model is usually a simple labelled tree that represents elements and aggregation relationships. This simple labelled tree representation doesn’t capture all the semantics of XML schemas inferred by cardinality constraints, type inheritance, substitution groups and namespaces.

Several attempts are done to represent XML Schemas at a semantic level. (Passi et al., 2002) translates XML Schema into an object-oriented model of nodes. Each element is then represented by an object containing its name, its namespace, its data type, its cardinality and a set of aggregation relations with other elements. Although, this proposed model is richer than the usual tree representation, it doesn’t capture all XML Schema features such as abstract types and subtyping relationships.

Figure 3 shows an extract of an XML schema, where the complex type Report is an abstract type. Abstract types may not have direct instances, but their subtype (Internal_report) may. Using an object-oriented representation where each node describes an element, this XML schema will be equivalent to a node report having two aggregation relationships with nodes title and subject. This implies that all the information present in the Internal_report type is lost.

Since a major part of XML schema semantics is expressed though typing relations, we decide to adopt an object-oriented model as our canonical conceptual model, in which complex Types and user-defined simple types (defined as restriction of predefined simple types) are represented as objects and are considered as concepts. Figure 3 illustrates our representation of Report and Internal_report types, and reflects the subtyping relation between them.

Elements declared within a concept are considered as properties (such as introduction within Head concept in figure 4).

Aggregation relationships are represented with edges labelled with element names (in figure 4 an edge labelled head represents the aggregation relationship between concepts Course and Head).

Subtyping relationships between types are indicated by edges labelled E for extensions and R for restriction. Every type in the schema graph not explicitly associated with its supertype stands in subtyping relationship with type ANY (not illustrated in the example of figure 4).
The example of figure 4 represents the canonical conceptual model of the XML schema course.xsd described in the annexe.

```xml
<xs:element name="course">
  <xs:complexType>
    <xs:sequence>
      <xs:element name="head" type="Head"/>
      <xs:element name="chapter" type="Chapter"/>
      <xs:element name="end" type="End"/>
      <xs:element name="title" type="xs:string"/>
      <xs:element name="alternateTitle" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
</xs:element>
```

Course1.xsd

-----Elements' declaration
<xs:element name="course" type="Course"/>

-----Types' declaration
<xs:complexType name="Course">
  <xs:sequence>
    <xs:element name="head" type="Head"/>
    <xs:element name="chapter" type="Chapter"/>
    <xs:element name="end" type="End"/>
    <xs:element name="title" type="xs:string"/>
    <xs:element name="alternateTitle" type="xs:string"/>
  </xs:sequence>
</xs:complexType>

Course2.xsd

Figure 2. Example of XML schema rewritten according to our abstract model

4.3. Comparison

This module establishes similarities between concepts identified by the previous module. It performs what we call mappings discovery. Several different similarity measures have been proposed in literature. We adopted a multi-strategy process that calculates similarities between concepts at different levels. The first strategy focuses on the calculation of linguistic similarity. The most basic similarity measure between concepts is to match their names to resolve any homonyms, synonyms, etc. Constraints on the cardinality must also be taken into consideration. In general, given two concepts' names, their term similarity or name affinity in a domain can be provided by thesauri. For this purpose, WordNet thesaurus [WordNet] is used.
Subsequently, the next step calculates \textit{property similarity}, that is similarity between concepts based on their properties.

A further step calculates \textit{structural similarity}, \textit{e.g.} the similarity of the context in which the concepts occur in their respective conceptual model. In this step, we take in consideration subtyping relations and substitution group. We adopt an approach similar to the one described in (Madhavan \textit{et al.}, 2001). A bottom-up similarity intends to propagate similarity from lower concepts to upper concepts. Complementary, a top-down similarity enhances the similarity of lower concepts in function of parent concepts similarity. The rational is that concepts with highly similar ancestors occur in similar context. The detailed algorithm is out of the scope of this paper.

```xml
<xs:schema>
  <xs:element name="report" type="Report"/>
  <xs:complexType name="Report" abstract="true">
    <xs:sequence>
      <xs:element name="title" type="xs:string"/>
      <xs:element name="subject" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
  <xs:complexType name="Internal_report">
    <xs:complexContent>
      <xs:extension base="Report">
        <xs:sequence>
          <xs:element name="author" type="xs:string"/>
          <xs:element name="internal_decision" type="xs:string"/>
        </xs:sequence>
      </xs:extension>
    </xs:complexContent>
  </xs:complexType>
</xs:schema>
```

XML schema example

![Diagram of conceptual model]

The associated \textit{canonical conceptual model}.

\textbf{Figure 3. Canonical conceptual model example}
4.4. Conflict resolution

During the *comparison* stage of integration process, heterogeneity between concepts from different schemas is identified. Almost all factors for heterogeneity can be classified into *semantic* issues (e.g., interpreting and relating information in different schemas), and *Schematic* issues (e.g., heterogeneity in schemas’ definition). N*aming conflict* (different names for equivalent concepts or same names for different concepts), *constraint conflict* (includes for example occurrences, default values, etc), *structural conflict* (includes properties’ occurrence, sequence of properties, etc.), and *data type conflict* are schematic conflicts examples.

The *conflict resolution* module is responsible for establishing correspondence between concepts. It extracts semantic and structural information needed to perform adaptation between schemas.

To resolve such conflicts, we provide an extension of XQuery language (XQuery 01), called *Schema_Xquery* (*Xquery for XML Schemas*) that supports the design of global schema, and define relationships between concepts of global schema and the concepts of local schemas. Using a query language for database integration is not a new approach. Several query languages are proposed in the literature to address this problem such as (Lakshmanan *et al.*, 96), (Sattler *et al.*, 00). This approach present several advantages compared to the classical rule-based approach, where a set of rules and heuristics has to be maintained to resolve conflicts.
As identified in (Lakshmanan et al., 96), some of the key features required for a schema definition language for database federation are the following. (1) The language must be easy and expressive enough. (2) The language should have a syntax that is independent from the schema with which the databases are structured. (3) The language must provide full data manipulation and view definition capabilities. (4) To promote interoperability, the language should permit the restructuring of one database schema to another.

The proposed Schema_Xquery language meets the above criteria, reviews the syntax of XQuery and provides additional operations and functions to manipulate heterogeneous schemas. (1) The syntax used is based on XQuery language syntax. We impose this requirement, first because the importance and the popularity of XQuery and second to take benefits from existing XQuery processors. (2) Schema_XQuery retains the flavour of Xquery, while supporting querying both data and schemas.

For this additional functions that do not exist in XQuery were added such as SCHEMA () and CONCEPT (). The latter functions are equivalent to DOCUMENT function in XQuery but parse the schema graph instead of an XML document (figure 5). (3) Schema_Xquery supports creating views and the ability to dynamically create output schemas. (4) Schema_Xquery supports the definition of global schema and permits the restructuring of one schema to another. For this it extends XQuery to manipulate XML Schemas with various integration operations like those used in relational and object-oriented databases. For example, operations like join, union, intersect, concat are used to resolve structural conflicts. Rename operation is used in the case of naming conflict. To resolve data type conflicts, operations for type expansion, facet redefinition and scale differences are also available.

4.5. Merging and restructuring

Based on resolution technique identified by the previous module, an Schema_Xquery request is generated automatically and adequate functions are called to resolve conflicts (e.g. In case of naming conflict, rename function is called).

Using this technique to resolve conflicts, a global schema is built through multiple Schema_Xquery queries and mappings between global schema and local XML Schema sources are kept within these queries. This technique is also used to automatically transform and adopt a reused course support to a target schema.
XQuery definition:

```
for $a in document("course.xml")/course/chapter
where $a/title = "java"
return $a/section
```

XQuery Result:

```
<section>
  <title>Programming language</title>
  ....
</section>
```

Schema XQuery definition:

```
for $a in Schema("course.xsd")/course/chapter
return $a/Concept(section)
```

Schema XQuery Result:

```
<xs:complexType name="Section">
  <xs:sequence>
    <xs:element name="title" type="xs:string"/>
    <xs:element name="theorem" type="xs:string"/>
    <xs:element name="proof" type="xs:string"/>
    <xs:element name="exercise" type="xs:string"/>
  </xs:sequence>
</xs:complexType>
```

Figure 5. Comparison between XQuery and Schema_Xquery

5. Conclusion

The importance of structured document paradigm in education domain, especially with the intensive use of XML technology has made a large amount of heterogeneously structured courses widely available in distributed repositories. In order to balance the autonomy of each community with the need of interoperability, mappings techniques between several data models have been proposed. We presented, within the framework of FAVORITE project, an XML schema integration strategy that facilitates access to structured courses by providing global view of distributed structured courses. However, it is applied in the educational domain, our approach is designed to be general and applicable in the generation of new structured documents through reusing existing ones.

We first have defined an abstract model for XML Schema, which is an attempt at a simpler and more formal description. To deal with both the syntactic and the semantic level of XML Schema, we proposed a semantic representation (Canonical conceptual model). This representation is richer than the simple labelled tree representation always adopted in the case of DTDs integration. It reflects subtyping relationships and substitution groups.

To perform mappings discovery, we adopted a multi-strategy process that calculates similarities between concepts at different levels. Linguistic similarity or name affinity is provided by WordNet thesaurus. Property similarity, and thus concept similarity, are calculated using simultaneously a bottom-up and a top-down approaches.

In order to resolve conflicts and to design the global schema, a new query language for XML Schemas (Schema_XQuery) was proposed. It extends XQuery to manipulate XML Schemas with various integration operations like those used in
relational and object-oriented databases such as \textit{join}, \textit{union}, \textit{intersect}, \textit{sort}, etc. The same query language is used to automatically transform and adopt a reused course to a target schema.

In the future, much work remains to be done. Especially, enriching the abstractness module with a \textit{normalisation} phase. The aim of this phase is to eliminate heterogeneity in schemas declarations due to the use of abbreviations, acronyms, punctuations, etc. For this purpose, three ordered tasks are required:

–\textit{tokenization} (The names are parsed into tokens using punctuation, upper case, special symbols, etc.).
–\textit{expansion} (Abbreviations and acronyms are expanded) and
–\textit{elimination} (Tokens that are prepositions, articles, etc are marked to be ignored by during similarity calculation).

An important part of our future research will be devoted to the facilities offered by the application layer based on the data model and integration strategies developed herein. Two promising approaches may be adopted for this task. The querying based approach represent the first direction and is based on the proposed query language (\textit{Schema_Xquery}). The second direction is based on navigational models. This will be based on the recent work done in the domain of hypertext navigation and information retrieval.

6. References


(ARIADNE 00) Alliance of Remote Instructional Authoring and Distribution Networks for Europe (Ariadne). http://ariadne.unil.ch/
S. Bergamaschi, S. Castano, M. Vincini, “Integration of Semistructured and Structured Data Sources”, \textit{SIGMOD Record}, 28(1), March 1999.


(Favorite 02) Fragmenting and Authoring Virtual Object-based documents for Reuse in Interactive Teaching Environments. Project Home Page: http://lithpc8.epfl.ch


Enhancing Course Reusability


IBM Informix, http://www.informix.com


A. Levy, A. Rajaraman, J.J. Ordille, “Querying Heterogeneous Information Sources Using Source Descriptions”, In Proc. of the 22nd Int. Conf. on Very Large Data Bases (VLDB Journal’96), Mumbai (Bombay), India, September 1996.


Oracle, http://www.oracle.com


OpenLink Virtuoso, http://www.openlinksw.com/virtuoso


(WordNet) http://www.cogsci.princeton.edu/~wn


Annexe: (Course.xsd)