BInXS: A Process for Integration of XML Schemata*

Ronaldo dos Santos Mello[1] and Carlos Alberto Heuser[2]

[1] Universidade Federal de Santa Catarina,
Depto. de Informatica e Estatistica, Cx. Postal 476,
Florianopolis, SC, Brasil 88040-900
ronaldo@inf.ufsc.br
[2] Universidade Federal do Rio Grande do Sul,
Instituto de Informatica, Cx. Postal 15064,
Porto Alegre, RS, Brasil 91501-970
heuser@inf.ufrgs.br

Abstract. This paper presents a detailed integration process for XML schemata called BInXS. BInXS adopts a *global-as-view* integration approach that builds a global schema from a set of heterogeneous XML schemata related to a same application domain. This bottom-up approach maps all element and attribute definitions in XML schemata to correspondent concepts at the global schema, allowing access to all data available at the XML sources. The integration process is semi-automatically performed over conceptual representations of the XML schemata, which provides a better understanding of the semantics of the XML data to be unified. A conceptual schema is generated by a set of conversion rules that are applied to a schema definition for XML data. Once this conceptual schema is the result of a meticulous analysis of the XML logical model, it is able to abstract the particularities of semistructured and XML data, like elements with mixed contents and elements with alternative representations. Therefore, the further unification of such conceptual schemata implicitly deals with structural conflicts inherent to semistructured and XML data. In addition, BInXS supports a mapping strategy based on XPath expressions in order to maintain correspondences among global concepts and data at the XML sources.

1 Introduction

The XML format has been extensively used to represent data as well as to interchange data among users and applications, specially through the Web [7]. Several application domains, like e-commerce [1, 3] and bibliographic references [2, 4], provides XML information on the Web. Considering such increasing availability of XML data, schema integration mechanisms are required to provide an unified access to several heterogeneous XML sources on the Web related to a same application domain.

An XML data is a semistructured data [8]. Thus, the integration of XML schemata is more complex than the integration of database schemata because semistructured schemata are irregular, allowing the definition of heterogeneous instances in a same

* This work was partially supported by CAPES Foundation.

O. Pastor and J. Falcão e Cunha (Eds.): CAiSE 2005, LNCS 3520, pp. 151–166, 2005.

schema. Because of this high heterogeneity, it is difficult to find out semantic correspondences among XML data based on a structural analysis of them, as well as to solve conflicts of data representation in order to perform a unification.

Database schema integration processes usually convert the data models of the heterogeneous databases to a common data model called *canonical model* [10, 17, 31]. This canonical representation abstracts the heterogeneity of the data models, reducing the complexity of the integration activity. Considering the specific integration of XML schemata, there is a choice between: *(i)* to convert the XML data model to a canonical model that is able to abstract the high structural heterogeneity of each XML schema or; *(ii)* does not perform such conversion. Alternative *(i)* requires a conversion process and mappings from one model to the other. However, the complexity of the further integration is reduced. Alternative *(ii)* does not require the conversion, but has to deal with the complexity inherent to the integration of XML schemata.

Several related work on semistructured or XML schema integration apply alternative *(i)* [11, 20, 21, 23, 24, 28, 30]. However, their main drawback is that the adopted canonical model does not consider all the particularities of the XML data model. Consequently, they do not deal with some kinds of conflicts that raise when XML schemata are unified, like elements with mixed content (text and structure) and elements with alternative representations.

This paper presents a process for XML schema integration called **BInXS**[1]. BInXS also follows alternative *(i)*, proposing a conceptual canonical representation to a schema for XML data. Such canonical representation results of a detailed analysis not only of the XML data model, but also of XML instances in order to improve the understanding of data semantics. The further schema unification applied on these canonical schemata takes implicitly into consideration the resolution of conflicts related to XML schemata, like the ones exemplified before. The main advantage of such approach is that the integration is applied on a conceptual basis, i.e., on high level and detailed abstractions of XML schemata. A global conceptual schema is generated at the end of the integration process. This global schema is useful in the context of a mediation system [12] that provides access to XML sources on the Web.

This paper is organized as follows. Section 2 gives an overview of the integration process followed by BInXS. Section 3 describes the conversion of an XML schema to a conceptual schema. Section 4 describes how the global schema is defined from the unification of conceptual schemata. Section 5 discusses some related work. Section 6 is dedicated to the conclusion.

2 BInXS Overview

BInXS is a *semi-automatic* and *bottom-up* process for semantic integration of XML schemata [25]. It is *semi-automatic* because user intervention is needed in order to validate the semantic intention of data during the integration process. Semantic integration processes are not fully automatic because the definition of a precise meaning for a data

[1] **BInXS** is an acronym for **B**ottom-up **In**tegration of **X**ML **S**chemata.

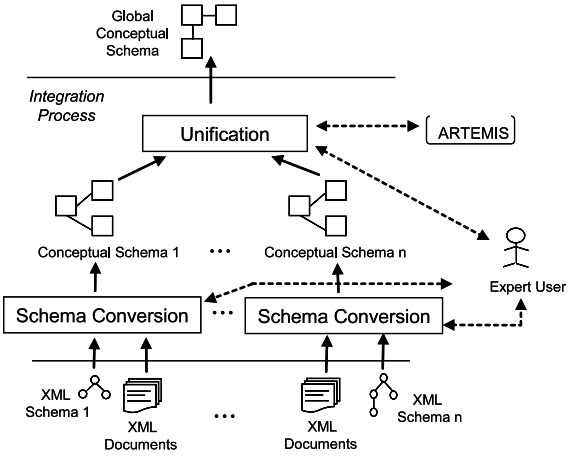

Fig. 1. BInXS integration process

is a very subjective matter. BInXS is also a *bottom-up* process because it generates a global schema from a set of XML schemata, being classified as a *global-as-view* integration approach [18]. Such global schema abstracts the high heterogeneity of the XML data sources and considers the semantic intention of all of these sources.

BInXS has two phases, as shown in Figure 1. The first phase, called *Schema Conversion*, maps each XML logical schema to a correspondent conceptual schema. BInXS adopts a conceptual canonical model because it provides a high level abstraction for the XML data. Besides, a same conceptual schema may abstract several XML logical schemata of a same application domain. The unification of conceptual representations of XML data reduces the complexity of the integration process because it is much simple to find out semantic similarities among conceptual schemata, which straightly represent real world facts and their relationships. The *Schema Conversion* phase is detailed in section 3.

Not only XML schematic information are analyzed in this first phase, but also data in XML documents. Such data analysis is necessary to define a more accurate conceptual schema, helping on the definition of relationship cardinalities and relationships derived from XML element references, for example. The intervention of an expert user is expected to validate automatic-generated conceptual schema concepts in order to obtain a definitive conceptual schema. Mapping information from conceptual schema concepts to XML elements or attributes are also generated and kept in a catalog.

The second phase, called *Unification*, takes a set of conceptual schema generated from the previous phase and performs their semantic integration, creating a global conceptual schema. An external tool, called ARTEMIS, is used to find out semantic affinities between concepts in different schemata. User intervention is considered again to eventually choose one among several alternative semantic meanings for a global concept or relationship representation, or to validate an automatic-generated preliminary global schema. Section 4 details this phase.

3 Schema Conversion

The *Schema Conversion* phase is based on a set of rules that consider the concepts of the XML model, analysis of XML documents, and user expertise [27]. The conversion process has three steps: *Pre-processing*, *Conversion* and *Restructuring*.

The *Pre-processing* step takes an XML schema (a DTD or XSD specification) and modifies its definition in order to generate a more well-structured and simplified schema to be further converted. Examples of schema modifications are: removal of elements that are not semantically relevant (for example, an `author-list` element as a component of an element `book`, acting as an intermediate element between `book` and `author` elements); and the replacing of nested components by a new element type (called *virtual element*) that abstracts the set of component elements[2]. Some of these modifications require user intervention, like the first example.

The *Conversion* step takes a pre-processed XML schema and applies a set of conversion rules on it, generating a *preliminary conceptual schema* and mapping information. Section 3.3 presents these rules. The *Restructuring* step takes a preliminary conceptual schema and performs manual and automatic modifications on it to produce a more semantically correct and simplified conceptual schema (a *definitive conceptual schema*). Examples of manual modifications are: definition of suitable names for automatic-generated concepts, and the validation of default cardinality constraints for relationships. An example of automatic modification is the removal of redundant relationships.

The considered XML and conceptual models are presented in the following, for sake of understanding of the conversion rules. It is necessary to introduce again the XML logical model in this paper because BInXS deals with several features of this model that are not fully considered in related work.

3.1 XML Model

The XML logical model defines *elements* and *attributes*. An element is composed by a *start-tag*, a *content model* and an *end-tag*. The content model defines what is enclosed between the *start-tag* and the *end-tag*. An *attribute* describes a property of an element. Its value is specified at the *start-tag* of the element. Figure 2 (a) shows an XML document. e_3 and e_{15} are elements and a_1 is an attribute of e_3. Figure 2 (b) shows the correspondent schema to this XML document [3]. A terminology to the concepts of the XML model is presented in the following. Examples are taken from Figure 2(b).

A **composite element** is an element with attributes and/or an element that has a content model defined by one of two XML grammatical constructs: sequence and choice. A *sequence* $(\mathtt{ec}_1, \mathtt{ec}_2, \ldots, \mathtt{ec}_n)$ defines *n* ordered component elements, with $n \geq 1$. A *choice* $(\mathtt{ec}_1|\mathtt{ec}_2|\ldots|\mathtt{ec}_m)$ define *m* alternatives for component elements,

[2] e_1Group_1 in Figure 2 (c) is a conceptual abstraction of a virtual element. See sections 3.1 to 3.3.

[3] Figure 2 (b) is a logical abstraction of a schema defined through a DTD (*Document Type Definition*) or an XSD (*XML Schema Definition*) [5].

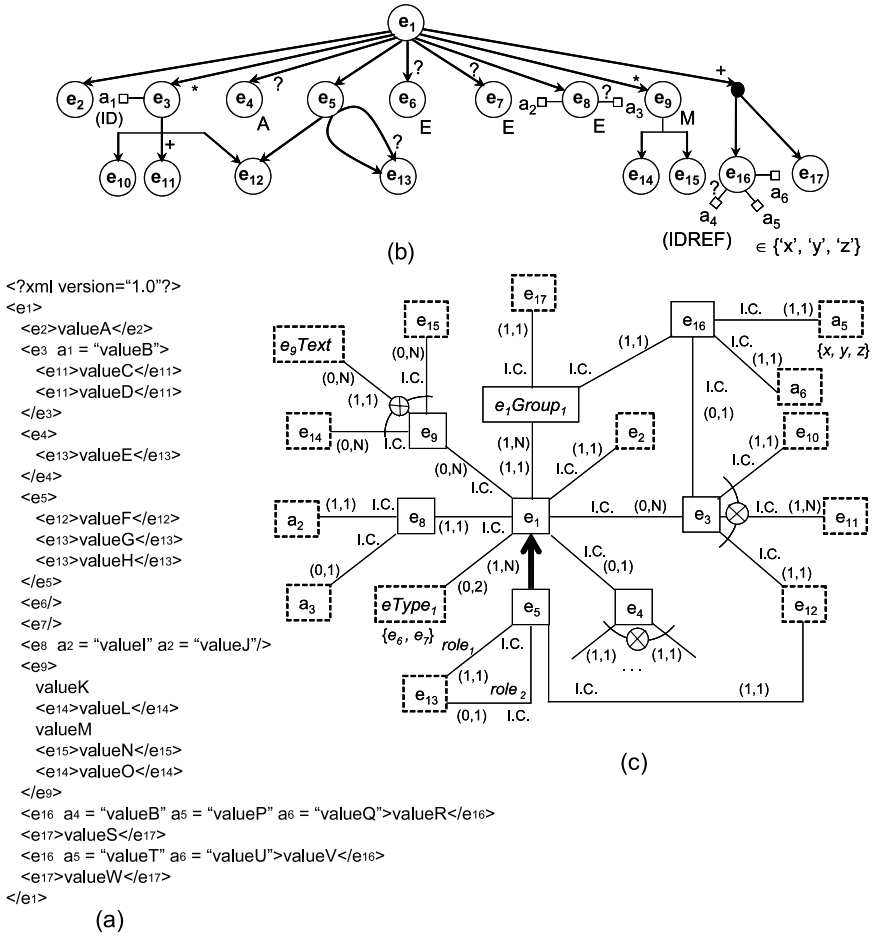

Fig. 2. An XML document (a), an XML schema for the document (b), and a conceptual schema for the XML schema (c)

with $m > 1$. The regular expression operators '?', '*' and '+' indicate the allowed number of occurrences of a composite element, denoting, respectively, 0 or 1 occurrence, 0 to n occurrences, and 1 to n occurrences. e_1, e_3, e_5, e_8 and e_{16} are examples. e_3 is an element defined by a choice and e_5 is an element defined by a sequence. A **nested component** is a *sequence* or *choice* specification that is embedded into the content model of a composite element. The composite element e_1 has a nested component comprised by e_{16} and e_{17}.

A **simple element** has a content model defined by a single value. e_2, e_{10} to e_{15}, and e_{17} are examples. An **empty element** has no content model, i.e., its content model is *empty*. e_6 and e_7 are examples (labelled by 'E'). A **free element** allows any kind of schema element in your content model. It corresponds to an `ANY` element in a DTD or XSD specification. e_4 is an example (labelled by 'A').

A **mixed element** has a content model that is a mix of values and component elements, i.e., it **is a composite element** with the following restrictions: *(i)* its content model is defined by a choice; *(ii)* its components may repeat from zero to *N* times; *(iii)* there is a special component (a *valued component*) without a name. e_9 is an example (labelled by 'M'). The elements e_{14} and e_{15}, as well as the implicit valued component, may occur from zero to several times into e_9 content.

An **attribute** is an optional or required property associated to an element, like a_2 and a_6. An attribute has a data type and may act as an element identifier (an `ID` attribute, like a_1). A **reference attribute** for an element e_x **is an attribute** that establishes a reference from an e_x element instance to one or more values of `ID` attributes of instances of elements. a_4 is an example (labelled by '`IDREF`' or '`IDREFS`').

3.2 Conceptual Model

BInXS adopts a graphic variant of the ORM/NIAM (*Object with Roles Model/Natural language Information Analysis Method*) conceptual model as the canonical model [19]. Figure 2 (c) shows an example of a conceptual canonical schema.

The ORM/NIAM model is based on two types of concepts: lexical and non-lexical concepts. A *lexical concept* models information that has an associated value (a dotted rectangle). a_5 and e_2 are examples of lexical concepts. A lexical concept has a data type (*string* or *integer*, for example), and an optional enumeration of allowed values, as shown in the concept a_5. A *non-lexical concept* models information that is composed by other information (a solid rectangle). e_1 and e_8 are examples of non-lexical concepts. The model supports binary *association relationships* (with optional roles) with cardinality constraints, and *inheritance relationships*. An association relationship is defined between the concepts e_1 and $e_1 Type_1$, and an inheritance relationship is defined between e_1 and e_5, being e_5 an specialization of e_1. It is still possible to model *mutually exclusive relationships*, like the relationships of e_3 with e_{10}, e_{11} and e_{12}.

The ORM/NIAM model was chosen to be the canonical model because it has a more straight correspondence with the XML logical model: non-lexical concepts are suitable to model composite elements, and lexical concepts are suitable to model simple elements and attributes. Besides, simple elements and attributes (valued information) may be associated to several composite elements in an XML schema. Such situation is also possible in the ORM/NIAM model, i.e., a lexical concept may have relationships with several non-lexical concepts. However, this is not possible in the ER model [9], for example, where valued information can only be modelled as an attribute, which is an exclusive property of an entity or relationship.

3.3 Conversion Rules

The *conversion rules* are the core of the *Schema Conversion* phase. They are summarized in the following[4].

[4] For sake of paper space, correctness and completeness of the conversion rules are not discussed. This is a focus of future work.

Rule 1 (Simple Element Conversion). A simple element E_S generates a *lexical concept* E_l with name E_S. The data type of E_l is the data type defined to the simple element, if exists; or *string*, otherwise.

Rule 2 (Empty Element Conversion). An empty element E_E generates a *lexical concept* $eType_i$, where *i* corresponds to the i-esimal converted empty element. The data type of $eType_i$ is set to *string* and its enumeration is set to $\{E_E\}$.

Rule 3 (Free Element Conversion). The conversion of a free element E_A proceeds as follows:

1. a *non-lexical concept* E_{nl} with name E_A is generated;
2. given *n* the number of lexical or non-lexical concepts *NL* that corresponds to XML elements, for *i* from 1 to *n*: generate an association relationship R_i between E_{nl} and NL_i with a direct cardinality *(0,1)* and an inverse cardinality *(0,N)*;
3. all previously defined relationships are set as *mutually exclusive*.

Rule 4 (Attribute Conversion). The conversion of an attribute a_x of a composite element E_C proceeds as follows:

IF a_x is a **reference attribute** and an analysis of XML documents indicates that all references of a_x instances points to instances of a same (target) element type E_T

THEN generates an association relationship between E_C and the non-lexical concept corresponding to E_T with a direct cardinality *([0—1],1)*, depending if the atribute is optional or not; and define the inverse cardinality through analysis of XML documents or assume *(1,N)* as default

ELSE generates a lexical concept E_l with name a_x. The data type of E_l is the data type defined to a_x, if exists; or *string*, otherwise. If a_x has an enumeration, it is transfered to E_l.

Rule 5 (Composite Element Conversion). The conversion of a composite element E_C proceeds as follows:

1. a non-lexical concept E_{nl} with name E_C is generated;
2. given $\{ec_1, ec_2, ..., ec_n\}$ the set of component elements of E_C, for each component element ec_i (1≤i≤n):
   IF ec_i is not an **empty element** and it is possible to infer an $<E_C$ *hyperonym* $ec_i>$[5] relation with the aid of a lexical database
   THEN generates an inheritance relationship R_i between E_{nl} and the concept correspondent to ec_i
   ELSE generates an association relationship R_i between E_{nl} and the concept correspondent to ec_i with a direct cardinality based on the defined regular expression operator; and an inverse cardinality defined through analysis of XML documents or assumed as *(1,N)* as default;
3. **IF** E_C is a **mixed element**
   THEN generates:
   (a) a lexical concept with a name *'E_C' + 'Text'*, and a data type *string*;

[5] t_1 *hyperonym* t_2 means that t_1 is a more general term than t_2.

(b) an association relationship between E_{nl} and *EText* with a direct cardinality *(0,N)* and an inverse cardinality *(1,1)*;

4. **IF** R_i associates E_{nl} with a lexical concept L_a generated from an **empty element** and there is another lexical concept L_b also generated from an empty element and an association relationship R_j between E_{nl} and L_b with the same direct cardinality **THEN** merges L_a and L_b into a lexical concept L_u, and merges R_i and R_j into an association relationship R_u between E_{nl} and L_u, adjusting properly the direct cardinality. The set of enumerations of L_a and L_b are also unified;
5. **IF** there is more than one relatioship R_1, R_2, ..., R_k between E_{nl} and a concept C_x **THEN** defines default names $role_1$, $role_2$, ..., $role_k$ to each respective relationship;
6. given $\{a_1, a_2, ..., a_m\}$ the set of attributes of E_C, for each attribute a_i ($1 \leq i \leq m$) generates an association relationship between E_{nl} and a_i with a direct cardinality *(1,1)* or *(0,1)*, depending if a_i is required or not, respectively; and an inverse cardinality defined through analysis of XML documents or assumed as *(1,N)* as default;
7. **IF** the content model of E_C is defined by a *choice* **THEN** set all previously defined relationships as *mutually exclusive*.

Figure 2 (c) is the preliminary conceptual schema generated by the application of the conversion rules on the XML schema in Figure 2 (b)[6]. The concepts e_2 and a_2, for example, are created by **Rule 1** and **Rule 4** applied to the element e_2 and the attribute a_2, respectively. **Rule 4** is also applied to the reference attribute a_4, defining an association relationship between the concepts e_3 and e_{16}. **Rule 3** applied to the element e_4 generates a same name concept and their mutual exclusive relationships with other element-derived concepts. **Rule 2** applied to the elements e_6 and e_7 generates the concepts $eType_1$ and $eType_2$, that are further merged into a single concept $eType_1$ by the application of **Rule 5** to the element e_1. It means that an empty element of a composite element E_C is considered a property (or *qualification*) of E_C, being represented as a lexical concept associated to it with a fixed value. Empty elements with the same direct cardinality are merged into a single lexical concept, with a set of allowed values.

Besides generating a concept e_9, **Rule 5** applied to the mixed element e_9 generates a new concept e_9Text that abstracts its valued components, and a set of mutually exclusive relationships that comprises the relationship to e_9Text and all relationships to the concepts generated to its component elements, considering that the content model of e_9 is defined by a choice. **Rule 5** applied to the element e_5 generates two association relationships from the e_5 concept to the e_{13} concept. Because of this, two default role names are defined to these relationships. In the *Restructuring* step, these names may be changed by the user, or the relationships may be merged if the user assumes that they have the same semantic meaning.

3.4 Mapping Strategy

Mapping information are defined during the *Conversion* step to each generated concept or relationship in the conceptual schema. BInXS adopts *XPath 1.0 expressions* to specify mappings to an XML schema [6]. *XPath* was chosen because it is a W3C recommendation for searching elements and attributes in an XML document.

[6] 'I.C.' denotes an automatic-generated default cardinality.

The mapping of a concept C_g is defined as an *absolute path expression* in *XPath*, i.e., a complete path from the root element to the C_g correspondent element or attribute in the XML schema. Given the conceptual and XML schemata in Figure 2 (c) and Figure 2 (b), the mapping of the concepts e_{14} and a_2 are denoted respectively by the expressions `'/e9/e14'` and `'/e8/@a2'`.

The mapping of a relationship is defined as a *relative path expression* in *XPath*. Such expression says how to navigate between related concepts in an XML schema. Mappings are defined for both relationship directions in order to allow the translation of any traversal over the conceptual schema graph. In Figure 2 (c), the *XPath* expressions `'e14'` and `'..'` denote, respectively, the mapping of the relationship between the concepts e_9 and e_{14} in the directions $e_9 \rightarrow e_{14}$ and $e_9 \leftarrow e_{14}$.

A query language for conceptual schemata called *CXPath* (*Conceptual XPath*) was defined in the context of the BInXS approach. A *CXPath* query is an *XPath*-like query that starts at a concept and traverses the schema graph in any direction in order to reach a desired related concept. With the proposed mapping strategy, the translation of a *CXPath* query does not become complex because the translation process will basically replace the concepts as well as the relationship traversals in a *CXPath* expression by their correspondent mappings in *XPath* to the schema of an XML source XS_i. Once unified, these *XPath* expressions define a complete *XPath* query to be executed at XS_i[7].

4 Unification

Once defined a set of conceptual schemata from local schemata[8], the *Unification* phase performs their semantic integration, generating a *global schema* [26]. To each global concept or relationship are associated the mappings to all respective local concepts or relationships that it represents. These mappings are kept in a global catalog. This phase follows the traditional database schema integration steps: *Schema Comparison*, *Merging* and *Restructuring* [10, 17].

The *Schema Comparison* step defines groups of synonym concepts coming from different local schemata called *affinity clusters*. An affinity cluster belongs to one of the following types: a *lexical cluster*, that holds only lexical concepts; a *non-lexical cluster*, that holds only non-lexical concepts; and a *mixed cluster*, that holds lexical and non-lexical concepts. The definition of these clusters is supported by an external tool called ARTEMIS [14], and it is out of the scope of this paper.

The *Merging* step is the core of the *Unification* phase. It generates concepts and relationships of a *preliminary global schema* through the merging of concepts in a same affinity cluster. Such merging is based on semi-automatic unification rules that are applied on the context of three unification cases: **LxL** (*lexical unification*), **NLxNL** (*non-lexical unification*), and **NLxL** (*mixed unification*). The next sections detail these cases.

Once performed the *Merging* step, the resulting *preliminary global schema* is validated in the *Restructuring* step through a set of automatic, semi-automatic and manual

[7] For sake of paper space, *CXPath* and the translation process are not detailed. See [13].
[8] From now on, XML schemata are called local schemata.

actions to generate the *definitive global schema*. An example of semi-automatic action is the definition of new inheritance relationships between global concepts with some common properties coming from different local schemata. Such relationships are defined with the aid of terminological databases and further user validation. An automatic action is the generalization of association relationships defined to all specialized concepts in an inheritance hierarchy. Manual adjustments include names of new concepts and relationship cardinalities.

4.1 Lexical Unification

The **LxL case** merges the concepts of a lexical cluster, generating a lexical concept L_G at the global schema. It corresponds to the merging of all XML valued content with affinity in different local schemata: simple elements, empty elements (considered properties), attributes and valued components of mixed elements. Specific rules determine the name, data type and allowed values of L_G.

Figure 3 shows the unification of two local schemata: `S1` and `S2`. This example is used to illustrate all unification cases. Several lexical clusters (denoted by *(L)*) are defined between local concepts, like 1 and 2. Cluster 1 generates the global concept *Style*, with a name chosen by the user between the names in the cluster. As both of the local concepts have enumerations, they are also unified. Cluster 2 generates the global concept *University*, whose name is the one with more incidences in the cluster.

4.2 Non-lexical Unification

The **NLxNL case** merges the concepts of a non-lexical cluster, generating a non-lexical concept NL_G at the global schema. It corresponds to the merging of all XML element types that are composed by other elements or attributes: composite elements, mixed elements and free elements.

To merge relationships, an iterative matching of pairs of concepts in the cluster is provided, until one single concept (NL_G) exists in the cluster. Basically, at each iteration it is analyzed if two relationships have *affinity*. Consider two concepts c_i and c_j in the same cluster. A relationship r_i of c_i has affinity with a relationship r_j of c_j if: *(i)* r_i and r_j have the same type (association or inheritance) and; *(ii)* both of them associate c_i and c_j with concepts in the same affinity cluster AC_L. If so, a merged relationship r_{ij} is generated from NL_G to the concept that represents AC_L in the global schema. User intervention is required when an association relationship r_i has affinity with more than one relationship of c_j (or vice-versa). Such situation raises when c_j has two or more association relationships with a same concept, and these relationships have roles. In this case, the user must decide if r_i has affinity with one of the c_j relationship or not. If a c_i or c_j relationship has no affinity with other relationships, it is considered an optional NL_G relationship.

The affinity cluster 13 is an example of non-lexical cluster, that generates the global concept *Address*. The relationships *Address-Country* (`S1`), *Address-Street* (`S2`) and *Address-ZipCode* (`S2`) become optional relationships because they have no affinity with other relationships. The relationships *Address-City* in `S1` and `S2` have affinity and are unified. The cardinality constraints are adjusted to be in accordance to both local cardinality constraints. The relationship *Address-Author* in `S2` has affinity with two `S1`

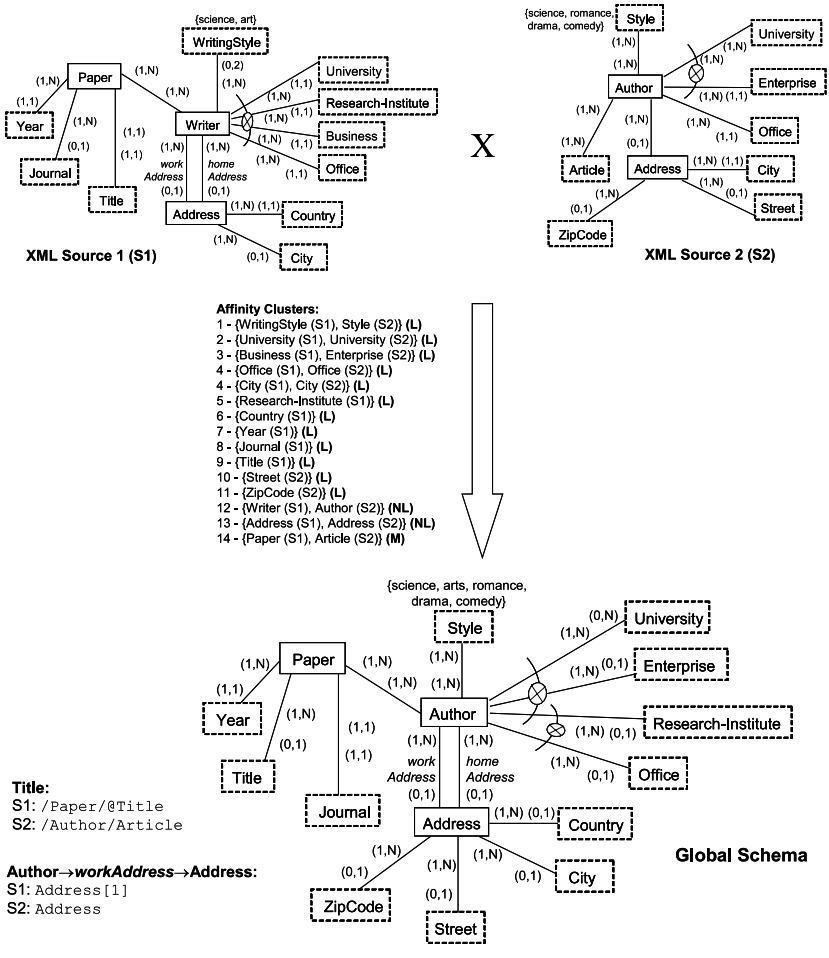

Fig. 3. An example of unification of two local schemata

relationships: *(i) Address-homeAddress-Writer* and *(ii) Address-workAddress-Writer*. Supposing that user intervention had decided by an affinity with relationship *(ii)*, it is indicated in the global schema that the relationship *Address-workAddress-Author* has mappings to `S1` and `S2`.

A mutual exclusion constraint defined to c_i relationships, for example, is directly represented at NL_G if such relationships have no affinity with c_j relationships. Otherwise, it is possible that a mutual exclusion constraint conflict exists, and a detailed analysis of c_i and c_j relationships must be performed[9]. In this case, only *valid mutual exclusions* are considered over NL_G relationships. Basically, a *valid mutual exclusion*

[9] This conflict is implicitly related to the problem of unifying XML elements with alternative representations.

is the one that comprises: *(i)* c_i disjoint relationships ri_1, ..., ri_n, and c_j disjoint relationships rj_1, ..., rj_n with affinity and; *(ii)* other c_i and c_j relationships that have no affinity but are disjoint of ri_1, ..., ri_n and rj_1, ..., rj_n, respectively. A subset of c_i and c_j relationships in a local mutual exclusion constraint without relationship affinity, or at most with one relationship with affinity, is also a valid mutual exclusion (case *(iii)*). Cases *(ii)* and *(iii)* preserve local mutual exclusion constraints at the global level.

The unification of the affinity cluster 12 in Figure 3 raises a mutual exclusion constraint conflict among the relationships of the local concepts *Writer* and *Author*. The conflict resolution performs as follows: the relationships with *University* and *Enterprise* are mutually exclusive in both local schemata. Therefore, an exclusion constraint me_i is defined between them in the global schema (case *(i)*). The relationship *Writer-Research-Institute* in `S1` has no affinity with `S2` relationships but is mutually exclusive of the two concepts mentioned above. Therefore, it is included in me_i in order to maintain the `S1` constraint (case *(ii)*). Besides, the relationship subset that comprises *Writer-Research-Institute* and *Writer-Office* is still mutually exclusive in `S1`. As *Writer-Office* is the only relationship with affinity, an exclusion constraint is defined on them at the global schema (case *(iii)*).

Again, observe that the global relationships *Author-University*, *Author-Enterprise* and *Author-Office* are defined as optional *Author* relationships. Such definitions avoid that, for example, *Author-University* and *Author-Office* always occur simultaneously at the global level, considering that their correspondent relationships in `S1` are mutually exclusive. Such analysis is also performed during the resolution of mutual exclusion conflicts.

4.3 Mixed Unification

The **NLxL case** merges all the concepts of a mixed cluster, generating a global non-lexical concept NL_G. It corresponds to the merging of structured and valued information with affinity in different local schemata.

The unification proceeds as follows: first, all non-lexical concepts are unified into a preliminary non-lexical concept NL_P by the application of the **NLxNL case**. After, for each remaining lexical concept L_i in the cluster, the user decides by one of the following alternatives: *(i)* L_i is mapped to a global lexical concept related to NL_P, assuming that L_i has a semantic correspondence with a NL_P property; *(ii)* L_i becomes a global concept and a new non-lexical concept NL_U is defined as a mutually exclusive generalization of L_i and NL_P. Such alternative assumes that L_i corresponds to the union of two or more NL_P properties, and must be denoted as an alternative representation for NL_P at the global level; *(iii)* L_i becomes a global concept associated to NL_P, assuming that L_i has no semantic correspondence with NL_P properties.

The affinity cluster 14 in Figure 3 is an example of a mixed cluster composed by the lexical concept *Article* and the non-lexical concept *Paper*. Considering that *Article* keeps titles of articles in `S2`, it corresponds to the lexical concept *Title* associated to *Paper* in the global schema (alternative *(i)*). Then, this mapping to the concept *Title* is also kept in the global schema, as shown in Figure 3. Alternative *(ii)* could be applied if, for example, *Article* content was a complete bibliographic reference, including not only a title, but also other reference information. In this case, a non-lexical concept

GenericPaper could be defined as a mutually exclusive generalization of *Paper* and *Article*, representing an abstraction of two possible disjoint representations for a *paper*.

5 Related Work

There are several work related to the integration of semistructured data [11, 21, 23, 24] or XML sources [15, 16, 20, 22, 28, 29, 30, 32]. Some of them gives support to a manual integration process, acting only as a tool that aids the user to define global views or mappings among local schemata [24, 30]. Thus, their integration process has a low quality because they provide a weak automation level. Another point is the canonical model. There are work that deal with hierarchical models for semistructured data as the canonical model, or performs the integration straightly over the XML model [15, 16, 24, 29, 32]. As a conceptual schema is not considered, their models enforce the structural organization of data instead of data semantics.

An approach different from BInXS is followed by [24], that defines mappings among local schemata instead of creating a global schema. This alternative is not adopted by BInXS because we are considering the context of the Web, where there are a lot of available XML sources. In this context, it is preferable to define a global representation of these sources in order to provide an integrated access to them.

Close related work are [11, 20, 21, 23, 28], which also propose semi-automatic schema integration of conceptual representation of semistructured schemata. However, they do not consider all features of the XML logical model, like elements with alternative representations, mixed elements and references between elements, or do not detail the mixed unification case as BInXS does. In [22], it is proposed an ER-like conceptual model for representing XML data that considers XML hierarchical relationships between elements in the conceptual schemata. As the same related real world facts may be expressed by different hierarchies in two or more XML schemata, this model is not suitable to represent an integrated view of these schemata.

6 Conclusion

BInXS is a solution to the problem of schema integration for XML data. The focus on XML schemata is justified by the widespread use of XML protocols by users and applications to represent and interchange data, specially over the Web. The bottom-up approach followed by BInXS is suitable to the context of the Web because it provides an unified view of a lot of heterogeneous XML sources over the Web. If used as a basis for querying XML data sources, this unified view avoids that users and applications must know the schema of each XML source in order to formulate a query.

Compared to related work, the main contributions of BInXS are the following:

- *A semi-automatic conversion process of an XML schema to a conceptual schema*: this process is based on a detailed analysis of the XML logical model and XML documents in order to obtain a correspondent conceptual abstraction where data semantics is much clear. The proposed conceptual representation is able to model all

types of elements (simple, composite, mixed, etc); attributes; element-to-element association, element-to-attribute association, references between elements; inferred inheritance relationships between elements; and alternative representations for elements;
- *A semi-automatic unification process for conceptual representations of XML schemata*: this process is suitable to XML schema integration because takes into consideration the implicit merging of heterogeneous XML data, with content models that may hold a value, a structure composed by other XML data, a mix of value and structure, and have alternative representations;
- *A mapping strategy between a global schema and an XML schema*: the *XPath* language is used to define mapping expressions from conceptual data to XML data. Because *XPath* is a language for querying XML data, a query defined over the global schema is easily translated to an *XPath* query to be executed at an XML source. No similar strategy was found in related work.

As user expertise is considered in the process, a good integration quality is always expected. However, future work include the consideration of instance-based integration techniques at BInXS with the purpose of improving the quality of the results generated automatically. On combining schema and instance analysis of XML sources, it is possible to establish semantic correspondences with much precision. The consideration of semantic integrity constraints of local XML sources is also important. Such information could be available and associated to concepts and relationships of the global schema in the global catalog. Thus, if a global query q_i defines a selection predicate that is not in accordance to the semantic constraints of an XML source XS_i, q_i does not need to be translated to XS_i because no XML instances will be retrieved from there.

References

1. CXML.org. Available at: <*http://www.cxml.org*>, mar 2005.
2. DBLP Bibliography. Available at: <*http://www.informatik.uni-trier.de/~ley/db/*>, mar 2005.
3. EBisXML. Available at: <*http://www.basda.org*>, mar 2005.
4. SIGMOD Record. Available at: <*http://www.acm.org/sigs/sigmod/record/xml*>, mar 2005.
5. W3C XML Schema. Available at: <*http://www.w3.org/XML/Schema*>, mar 2005.
6. XML Path Language. Available at: <*http://www.w3.org/TR/xpath*>, mar 2005.
7. Extensible Markup Language (XML). Available at: <*http://www.w3.org/XML*>, mar 2005.
8. S. Abiteboul, P. Buneman, and D. Suciu. *Data on the Web: From Relations to Semistructured Data and XML*. Morgan Kaufmann, San Francisco, California, 2000.
9. C. Batini, S. Ceri, and S. B. Navathe. *Conceptual Database Design: An Entity-Relationship Approach*. Benjamin/Cummings Publishing Company, 1992.
10. C Batini, M. Lanzerini, and S. B. Navathe. A Comparative Analysis of Methodologies for Database Schema Integration. *ACM Computing Surveys*, 18(4):323–364, december 1986.
11. S. Bergamaschi, S. Castano, D. Beneventano, and M. Vincini. Semantic Integration of Heterogeneous Information Sources. *Data & Knowledge Engineering*, 36(1):215–249, march 2001.
12. S. Busse, R. Kutshce, U. Leser, and H. Weber. Federated Information Systems: Concepts, Terminology and Architectures. (Technical Report, 99-9), Berlin: Universitt Berlin, 1999.

13. S. D. Camillo, C. A. Heuser, and R. S. Mello. Querying Heterogeneous XML Sources Through a Conceptual Schema. In *22th International Conference On Conceptual Modeling (ER)*, pages 186–199, Chicago, USA, 2003. Springer-Verlag.
14. S. Castano, V. Antonellis, and S. C. Vimercati. Global Viewing of Heterogeneous Data Sources. *IEEE Transactions on Knowledge and Data Engineering*, 13(2):277–297, march 2001.
15. I. F. Cruz, H. Xiao, and F. Hsu. An Ontology-Based Framework for XML Semantic Integration. In *International Database Engineering and Applications Symposium (IDEAS'04)*, pages 217–226, Coimbra, Portugal, 2004. IEEE.
16. A. Doan, P. Domingos, and A. Halevy. Reconciling Schemas of Disparate Data Sources: A Machine-Learning Approach. In *ACM International Conference on Management of Data (SIGMOD)*, pages 509–520, Santa Barbara, USA, may 2001.
17. A. Elmagarmid, M. Rusinkiewicz, and A. Sheth. *Management of Heterogeneous and Autonomous Database Systems*. Morgan Kaufmann, San Francisco, California, 1999.
18. A. Y. Halevy. Answering Queries Using Views: A Survey. *VLDB Journal*, 10(4):270–294, december 2001.
19. T. Halphin. *Object-Role Modeling (ORM/NIAM), Handbook on Architectures of Information Systems*, chapter 4, pages 81–102. Springer-Verlag, 1998.
20. P. Lethi and P. Fankhause. XML Data Integration with OWL: Experiences & Challanges. In *International Symposium On Applications and the Internet (SAINT'2004)*, pages 160–170, Tokyo, Japan, 2004. IEEE.
21. S. Lim and Y. Ng. An Automated Integration Approach for Semi-structured and Structured Data. In *3th International Symposium on Cooperative Database Systems for Advanced Applications (CODAS)*, pages 12–21, Beijing, China, april 2001. IEEE.
22. B. F. Lscio and A. C. Salgado. Generating Mediation Queries for XML-based Data Integration Systems. In *18th Brazilian Symposium on Databases (SBBD'2003)*, pages 99–113, Manaus, AM, october 2003.
23. J. Madhavan, P. Bernstein, and E. Rahm. Generic Schema Matching with Cupid. In *27th Conference on Very Large Data Bases (VLDB)*, pages 49–58, Rome, Italy, september 2001. Morgan Kaufmann.
24. P. McBrien and A. Poulovassilis. A Semantic Approach to Integrating XML and Structured Data Sources. In *13th Conference On Advanced Information Systems Engineering (CAISE)*, pages 330–345, Interlaken, Switzerland, 2001. Springer-Verlag.
25. R. S. Mello. *Uma Abordagem Bottom-Up para a Integracao Semantica de Esquemas XML*. PhD thesis, Universidade Federal do Rio Grande do Sul, july 2002. (In Portuguese).
26. R. S. Mello, S. Castano, and C. A. Heuser. A Method for The Unification of XML Schemata. *Information and Software Technology*, 44(4):241–249, march 2002.
27. R. S. Mello and C. A. Heuser. A Rule-Based Conversion of a DTD to a Conceptual Schema. In *20th International Conference On Conceptual Modeling (ER)*, pages 133–148, Yokohama, Japan, 2001. Springer-Verlag.
28. K. Passi, L. Lane, S. K. Madria, B. C. Sakamuri, M. K. Mohania, and S. S. Bhowmick. A Model for XML Schema Integration. In *3th International Conference On E-Commerce and Web Technologies (EC-WEB 2002)*, pages 193–202, France, 2002. Springer-Verlag.
29. C. Reynaud, J. Sirot, and D. Vodislav. Semantic Integration of XML Heterogeneous Data Sources. In *International Database Engineering & Applications Symposium (IDEAS)*, pages 199–208, Grenoble, France, july 2001. IEEE.
30. P. Rodriguez-Gianolli and J. Mylopoulos. A Semantic Approach to XML-Based Data Integration. In *20th International Conference On Conceptual Modeling (ER)*, pages 117–132, Yokohama, Japan, 2001. Springer-Verlag.

31. A. P. Sheth and J. A. Larson. Federated Database Systems for Managing Distributed, Heterogeneous, and Autonomous Databases. *ACM Computing Surveys*, 22(3):183–236, september 1990.
32. X. Yang, M. L. Lee, and T. W. Ling. Resolving Structural Conflicts in the Integration of XML Schemas: A Semantic Approach. In *22th International Conference On Conceptual Modeling (ER)*, pages 520–533, Chicago, USA, 2003. Springer-Verlag.