CHAPTER 8

QUERY PROCESSING FOR THE HETEROGENEOUS INFORMATION SOURCES USING METADATA DICTIONARY APPROACH

The querying process for the HIS (Arch-Int, Li, Roe, and Sophatsathit, 2003; Arch-Int, Sophatsathit and Li, 2003) aims to enable users to pose their queries over the virtual schema instead of the physical source schema so as to obtain relevant answers from the HIS. The querying process of the HIS encompasses two main processes, namely, the accessing process of the HIS and the integrating process of the results from HIS. The accessing and integrating process of the HIS can be accomplished through the metadata dictionary support. The accessing process is responsible for generating a global transaction associated with the user's request. The global transaction is then simplified, and decomposed into sub-transactions for accessing the real data in the physical information sources. The decomposition process focuses on mapping the virtual properties and concepts of the global transaction to physical properties and concepts of the sub-transactions via a mapping algorithm. In contrast, the integrating process focuses on consolidating the XML results obtained from executing each sub-transaction on a physical information source, whereby forming a unified XML-based data corresponding to the user's request. This unified XML-based data contain the relevant answers corresponding to user's request. This chapter also provides a means to handle data replication and query result validation and correctness being returned to the users. The algorithms for decomposition and integration process are given in the Appendix.

The query process is accomplished by acquiring the information from the metadata dictionary. Since the metadata dictionary is represented in XML format which is a tree-like structure, the metadata dictionary contents are organized in a conventional tree structure as illustrated in Figure 8.1. Searching the metadata dictionary can be accomplished in the same manner as a tree traversing. The proposed query process enables the semantic heterogeneity to be solved at query time for local and remote processing. Details on how it is carried out are described below.

8.1 The Accessing Process of the Heterogeneous Information Sources

The accessing process of the HIS starts at the presentation layer of the reference architecture proposed by Arch-Int and Sophatsathit, 2002, as shown in Figure 8.3. In this step, any virtual concept that is a subconcept of another concept inherits all of the virtual properties from its superconcepts. These virtual properties are thus presented to the user as the properties of the subconcept. For example, a user can view the virtual properties of `Instructor` originating from `Instructor` and `Staff`. The user can pose a query in a unified-query format encircling the virtual schema provided by the user interface agent or in standard SQL format. There are three steps involved in accessing the HIS, namely, global transaction creation, simplification, and decomposition.

(1) Global Transaction Creation: A global transaction is a visual user requirement represented in standard SQL format that consists of virtual concepts and properties of the virtual schema as illustrated in Figure 8.3. Upon submission of a user query that may be in any arbitrary complex form, the request will be sent to the user interface agent to form a global transaction, which is a normalized query form constructed by means of the metadata dictionary. The query normalization eliminates type mismatch, semantic mismatch, and redundant predicates (Özsu and Valduriez, 1999) from the global transaction. A formal description of a global transaction is given in Definition 8.1.

Definition 8.1 Let Q be a global transaction defined as a triple $<S, C, P>$, where $S = \{vc_i.vp_{ij} \mid \forall i=1\ldots n, \forall j=1\ldots m\}$ is a finite set of the target virtual properties vp_{ij} (or attributes) of the virtual concepts vc_i that are in the SELECT clause, and the property value of vp_{kc} that is defined over domain D_c; $C = \{vc_i \mid \forall i = 1\ldots n\}$ is a finite set of the target virtual concepts (or entities) in the FROM clause; and $P = Qp \cup Jp$ is a finite set of predicates in the WHERE clause that consists of two kinds of predicate: (*i*) the set $Qp = \{c_i \mid \forall i = 1\ldots n\}$ of qualifying predicates and (*ii*) the set $Jp = \{j_i \mid \forall i = 1\ldots m\}$ of join predicates, such that,

- A qualifying predicate $c_k \in Qp$ is defined as having $vc_k.vp_{kc}\ \Theta\ value$, where $\Theta \in \{=, <, >, \neq, \leq, \geq\}$ and $value \in D_c$ are defined in the qualified virtual property $vc_k.vp_{kc}$, and

- A join predicate $j_k \in Jp$ is defined as having $vc_k.vp_{kc} = vc_m.vp_{mc}$, where k $\neq$ m, and $vp_{kc} = vp_{mc}$.

Examples of a qualifying predicate is `Staff.st_id` = "11111" and a join predicate might be `Staff.dept_id = Department.dept_id` or `Instructor.st_id = Administrator.st_id`.

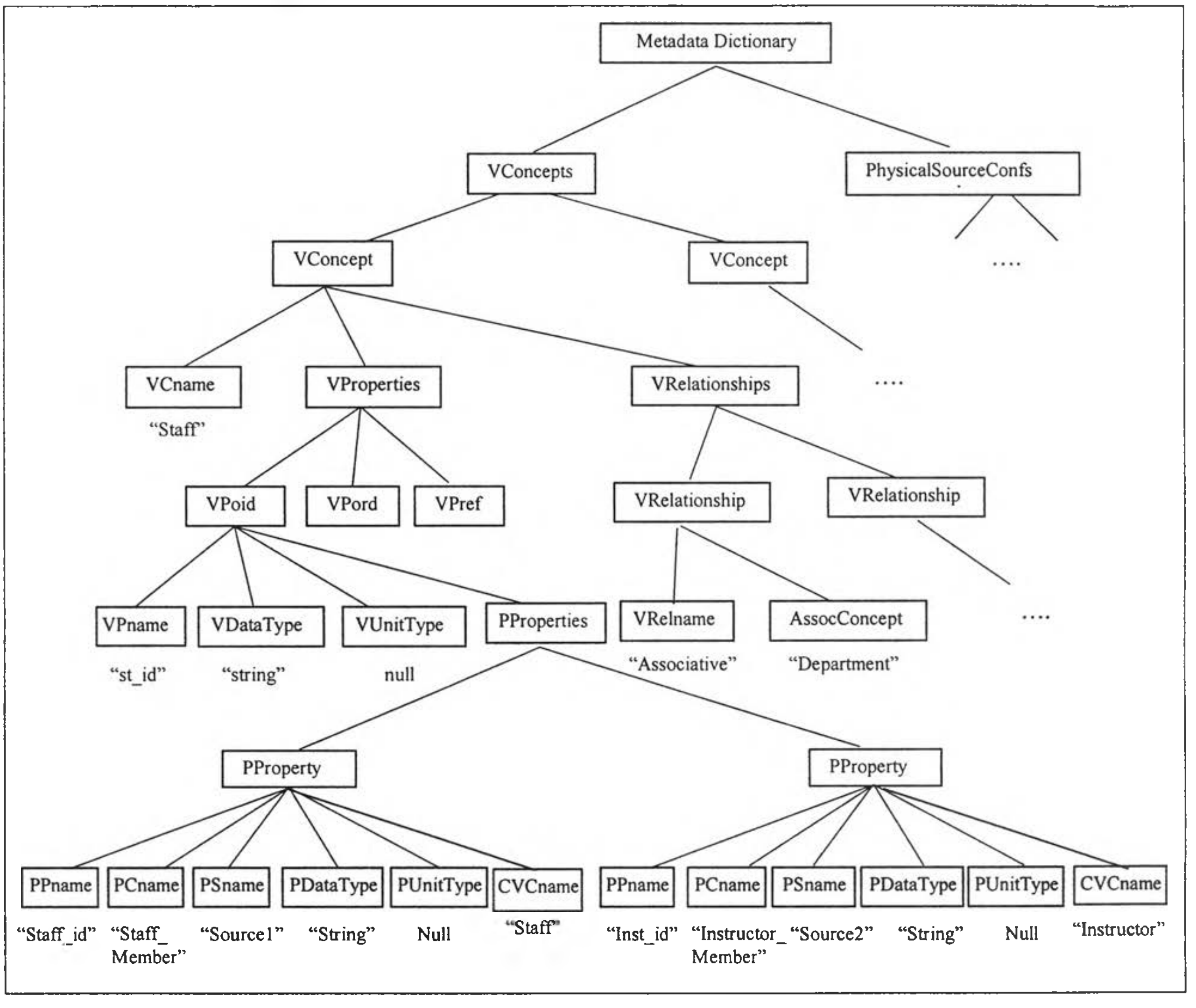


Figure 8.1 An example of the metadata dictionary contents represented by a labeled tree.

(2) Global Transaction Simplification: If a global transaction contains virtual properties selected from both of superconcept and subconcept, a join predicate between these concepts is called for to link the concepts. Such a global transaction can be simplified by substituting its superconcept with the subconcept and removing the join predicates. A formal description is given in Definition 8.2.

Definition 8.2 Given a global transaction Q containing the selected virtual properties $vc_i.vp_{im}$ and $vc_j.vp_{jn}$, where $i \neq j$, that form a *Superconcept(vc_i, vc_j)* and there exists a join predicate $vc_i.vp_{ik} = vc_j.vp_{jk}$ where $vp_{ik} = vp_{jk}$. The global transaction Q can be simplified to Q' by substituting the superconcept vc_i with the subconcept vc_j, denoted $vc_i \rightarrow vc_j$, and removing the join predicate from Q.

The example of the global transaction in Figure 8.2 (a) contains virtual properties `st_id` and `st_name` of a superconcept `Staff`, a virtual property `position` of a subconcept `Instructor`, and a join predicate `Staff.st_id = Instructor.st_id`. Since all of the virtual properties of `Instructor` are inherited from `Staff`, the global transaction in Figure 8.2 (a) can be simplified to be a normalized transaction as illustrated in Figure 8.2 (b) by replacing all properties of `Staff` with the corresponding properties of `Instructor` as well as the join predicate. This simplification is essential for subsequent processing.

```
SELECT Staff.st_id, Staff.st_name,
Instructor. position
FROM Staff, Instructor
WHERE Staff.salary > 10000
and Staff.st_id = Instructor.st_id
```

(a) A global transaction before the simplification process

```
SELECT Instructor.st_id,
Instructor.st_name, Instructor. position
FROM Instructor
WHERE Instructor.salary > 10000
```

(b) A global transaction after the simplification process

Figure 8.2. An example of the global transaction simplification.

(3) Global Transaction Decomposition: After global transaction simplification is complete, the global transaction is sent to the managing agent where global transaction decomposition is initiated. This process transforms the global transaction into sub-transactions by substituting each virtual concept and property in the global transaction with the corresponding physical concept and property of the local physical sources obtained from the metadata dictionary. The formalization is given in Definition 8.3.

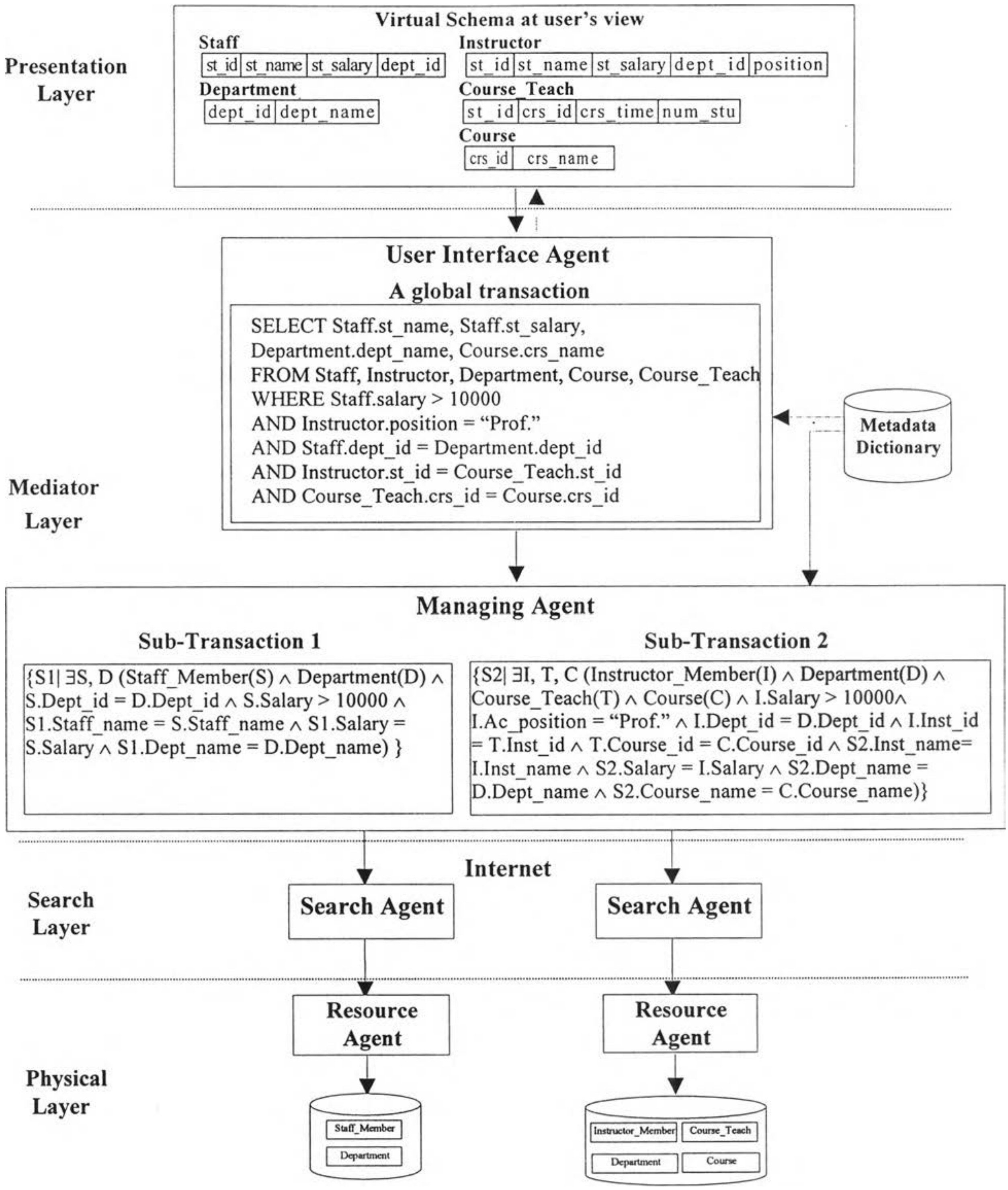


Figure 8.3. Accessing process of the heterogeneous information sources.

Definition 8.3 A simplified global transaction Q' can be transformed to sub-queries or sub-transactions $q'_1,\ldots, q'_n$ over the physical schema such that $q'_1,\ldots, q'_n$ encompass potential answers relevant to the user's query. The transformation process maps the virtual schemas in Q' to the physical schemas assigned in $q'_1,\ldots, q'_n$.

Let V be a virtual schema and P be a physical schema. The mapping relation σ_d from V to P is the function σ_d: $V \rightarrow P$, such that $\sigma_d\ (vc_k, vp_{kc}) = \{p_{kci} \mid p_{kci} = <PSname, PCname, PPname>\}$, where *PSname, PCname, and PPname* are physical source name, concept name, and property name, respectively, and $i = 1..n$.

For example, the virtual property `st_id` of the simplified global transaction Q' is replaced by `Staff_id` of `Staff_Member` and `Inst_id` of `Instructor_Member` to form *sub-transaction1* and *sub-transaction2*, respectively. A sub-transaction will subsequently access data from the designated physical information source.

The decomposition process of the global transaction is described in two steps as follows:

(1) *Mapping*. The virtual concepts and properties in the SELECT clause are mapped to the corresponding physical concepts and properties and, in turn, to the physical sources in which each physical concept resides. Some formal definitions of the terminology associating with a labeled tree are defined in Definition 8.4 and 8.5 to represent the basic structure of the XML metadata dictionary.

Definition 8.4 A *labeled tree*, T, is defined as a pair $<t, n>$, where t is a finite sub-tree consisting of one or more nodes, n is a finite set of labeled nodes of t.

A tree, t, has a root node denoted *root(t)*, with its children $v_1, \ldots, v_k$, $k \geq 0$. If (v, w) is an edge in t, then v is called the parent of w, and w is a child of v. A labeled node represents the begin-end tag in the XML data model. The attributes are denoted by tag elements of an XML document. A node consists of the attributes and ID (or key) whose value (true/false) is stored in that node.

Definition 8.5 A labeled node, *w*, is a quadruple <*l*, *d*, *k*, *p*>, where *l* is a label, *d* is a function that returns a value of the leaf node, *k* is a key function that returns "true" value if that node is a key and "false" if that node is not a key, and *p* is a set of pointers that point to the child nodes accompanied by a labeling function *l(w)* returning a *label* to node *w*.

The mapping process is carried out by means of a mapping algorithm, as illustrated in Appendix C, to acquire physical information from the metadata dictionary.

(2) *Sub-transactions creation.* Each sub-transaction is successively created from the following processes:

2.1 Grouping process. The virtual concepts/properties and the corresponding physical concepts/properties with the same physical source are grouped together in accordance with the following formulation.

Let $S = \{PSname_i \mid i = 1 \ldots n\}$ be a finite set of physical source names from the mapping process. A physical source name $PSname_k \in S$ is defined as a finite set of virtual concepts/properties and the corresponding physical concepts/properties such that $PSname_k = \{p_i \mid i = 1 \ldots n\}$, where p_k is defined as a 5-tuple $<vc_k, vp_{kc}, PCname_k, PPname_k, CVCname_k>$.

For example, the virtual properties/concepts in the global transaction of Figure 8.3 are mapped to physical information and grouped by *PSname* in the form of

S = {"Source1", "Source2"}, where

Source1 = {<"Staff", "st_name", "Staff_Member", "Staff_name", "Staff">,
<"Staff", "st_salary", "Staff_Member", "Salary", "Staff">,
<"Department", "dept_name", "Department", "Dept_name", "Department">}

and

Source2 = {<"Staff", "st_name", "Instructor_Member", "Inst_name", "Instructor">,
<"Staff", "st_salary", "Instructor_Member", "Salary", "Instructor">,
<"Department", "dept_name", "Department", "Dept_name", "Department">,
<"Course", "crs_name", "Course", "Course_name", "Course">}

2.2 Substitution process. In order to generate sub-transactions corresponding to the user's query, each `CVCname`$_k$ of $PSname_k$ is used to generate the initial sub-transactions.

The generation process can be divided into two cases, namely, no replicated data and replicated data.

- *No replicated data*: For each $PSname_k \in S$, if the subcomponent $CVCname_k$ matches with $vc_i \in C$ of the global transaction then generates a sub-transaction for accessing each $PSname_k$ by substituting the virtual concepts/properties in each $PSname_k$ with the corresponding physical concepts/properties to form a sub-transaction, denoted by $PSname_k$: $<vc_k \rightarrow PCname_k, vp_{kc} \rightarrow PPname_k>$. The physical properties constitute the requested information in the SELECT clause, and the physical concepts represent the target information sources to be accessed in the FROM clause.

The number of sub-transactions to be generated are taken from the corresponding virtual concept name (*CVCname*) for each physical source name. Since the *CVCname* in both `Source1` and `Source2` match with all the virtual concepts in the FROM clause of the global transaction, a corresponding sub-transaction is generated for each source by substituting the virtual properties/concepts by the physical properties/concepts, that is,

`Source1`: <“Staff” → “Staff_Member”, “st_name” → “Staff_name”>
<“Staff” → “Staff_Member”, “st_salary” → “Salary”>
<“Department” → “Department”, “dept_name” → “Dept_name”>

`Source2`: <“Staff” → “Instructor_Member”, “st_name” → “Inst_name”>
<“Staff” → “Instructor_Member”, “st_salary” → “Salary”>
<“Department” → “Department”, “dept_name” → “Dept_name”>
<“Course” → “Course”, “crs_name” → “Course_name”>

The initial sub-transactions are illustrated in Figure 8.4.

Source1	**Source2**
SELECT Staff_Member.Staff_name, Staff_Member.Salary, Department.Dept_name **FROM** Staff_Member, Department	**SELECT** Instructor_Member.Inst_name, Instructor_Member.Salary, Department.Dept_name, Course.Course_name **FROM** Instructor_Member, Department, Course_Teach, Course

Figure 8.4 Two initial sub-transactions generated from the substitution process.

- *Replicated data*: for two or more sources containing replicated data only one sub-transaction is generated from one of the replicated source. For example, if there is a physical source named `Source3` that replicates with `Source2` as illustrated in Figure 8.5, the information mapping of the three physical sources will be generated as follows:

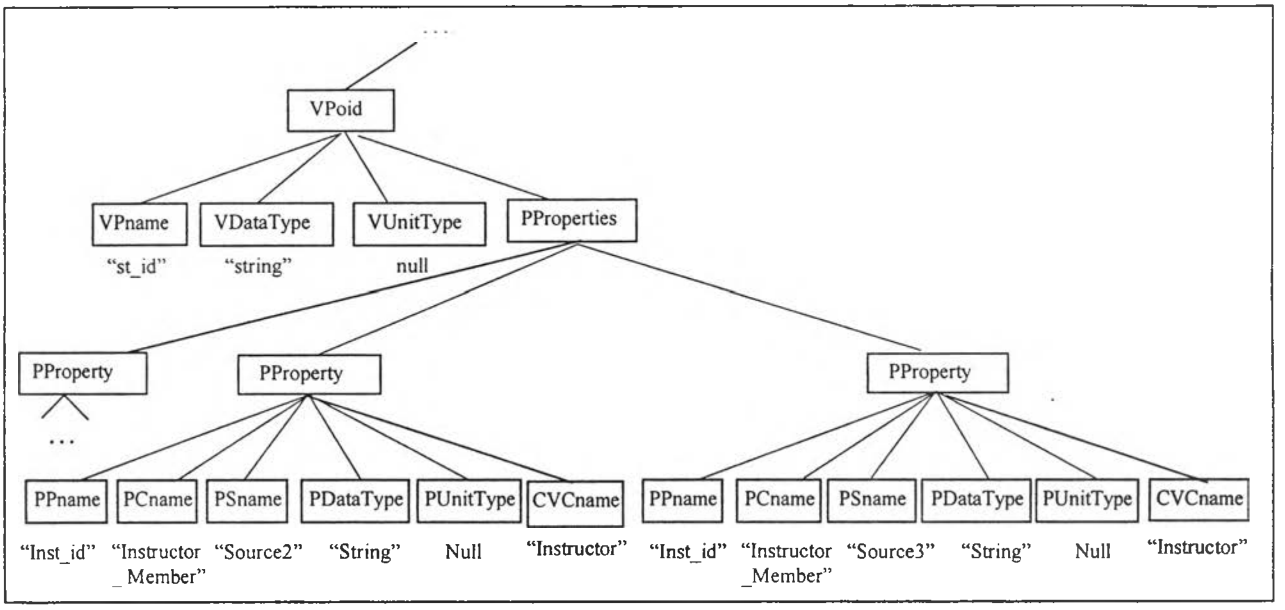


Figure 8.5 A portion of metadata dictionary illustrating replicated data.

S = {"Source1", "Source2", "Source3"}

Source1 = {<"Staff", "st_name", "Staff_Member", "Staff_name", "Staff">,
<"Staff", "st_salary", "Staff_Member", "Salary", "Staff">,
<"Department", "dept_name", "Department", "Dept_name", "Department">}

Source2 ={<"Staff", "st_name", "Instructor_Member", "Inst_name", "Instructor">,
<"Staff", "st_salary", "Instructor_Member", "Salary", "Instructor">,
<"Department", "dept_name", "Department", "Dept_name", "Department">,
<"Course", "crs_name", "Course", "Course_name", "Course">}

Source3 ={<"Staff", "st_name", "Instructor_Member", "Inst_name", "Instructor">,
<"Staff", "st_salary", "Instructor_Member", "Salary", "Instructor">,
<"Department", "dept_name", "Department", "Dept_name", "Department">,
<"Course", "crs_name", "Course", "Course_name", "Course">}

Thus, for each $PSname_k \in S$, if there exists two or more sets of $PSname_j$, $j = 1 \ldots m$, such that $PSname_1 = \ldots = PSname_m$, only one sub-transaction is generated from one of these sources. A sub-transaction similar to the one illustrated in Figure 8.4 is generated from this substitution process.

2.3 Generating the constraints: The virtual concepts/properties in the WHERE clause of a global transaction are also mapped to the associated physical concepts, properties and sources through the mapping algorithm. Two kinds of predicates in WHERE clause are considered, namely, qualifying predicates and join predicates.

2.3.1 Qualifying predicates. For each group with the same physical source, the qualifying predicates of the global transaction are replaced by the physical properties and concepts to form a set of constraints for use in a sub-transaction, that is, for each $PSname_k$, ($vc_k.vp_{kc}$ Θ value) → ($PCname_k.PPname_{kc}$ Θ value). For example, a qualifying predicate `Staff.st_id` = "11111" of `Source1` is replaced by `Staff_Member.Staff_id` = "11111" and `Instructor_Member.Inst_id` = "11111" to form the qualifying predicate in sub-transactions of `Source1` and `Source2`, respectively.

2.3.2 Join predicates. For each $PSname_k$, the join predicates of the sub-transactions are taken into consideration.

- If $PCname_k$ and $PCname_m$ correspond with vc_k, and vc_m, respectively, and reside in the same source, the join predicates of the global transaction are replaced by the same pairs of the physical properties and concepts, that is, $(vc_k.vp_{kc} = vc_m.vp_{mc}) \rightarrow (PCname_k.PPname_{kc} = PCname_m.PPname_{mc})$, where $k \neq m$, and $PPname_{kc} = PPname_{mc}$. For example, `Staff.dept_id = Department.dept_id` is replaced by `Staff_Member.Dept_id = Department.Dept_id` in a sub-transaction of `Source1`, since `Staff_Member` and `Department` refer to the same physical source.
- If $PCname_k$ and $PCname_m$ correspond with vc_k, and vc_m, respectively, but reside in different sources, there is no join predicate to be generated in the sub-transactions, and each individual sub-transaction operates in its respective physical source. For

example, the corresponding physical concept names of a join predicate `Instructor.st_id` = `Administrator.st_id` in the global transaction refer to `Instructor_Member` and `Administrator_Member`, which reside in `Source2` and `Source3`, respectively, there is no join predicates to be generated in the sub-transactions of `Source2` and `Source3`. This means that the returned results from these sources will be combined during the integration process that will be described in the next section.

All constraints obtained from the above procedures are combined to form the complete constraints of each sub-transaction as illustrated in Figure 8.3. Each sub-transaction, together with the physical source configurations that are necessary for accessing the HIS, is then packed and sent along with each search agent to the resource agent at the destination physical source. The actual information retrieval will be carried out by the resource agent.

8.2 The Integrating Process of the Heterogeneous Information Sources

Due to the different physical information sources that govern their own query languages in manipulating data represented in different data models, query language conflicts stemming from such differences must be eliminated. To eliminate these conflicts, each sub-transaction is transformed into the appropriate data manipulation language, regulated by each proprietary information source via the interface wrapper of the resource agent. The results obtained from the execution of each sub-transaction are converted to a canonical data model represented in an XML-based format via the interface wrappers. These XML results are transmitted to the managing agent, where the integration process takes place. The managing agent utilizes information obtained from the metadata dictionary to integrate XML results into unified XML-based data that consists of XML document and XML-DTD. The unified XML-based data is generated from the conceptual virtual schema of the global transaction according to a formal procedural definition (8.6) and is forwarded to the user interface agent, where the presentation formatting is configured at the presentation layer.

Definition 8.6 Given sub-transactions $q_1,\ldots, q_n$ generated from a global transaction Q, let $R(q_1),\ldots, R(q_n)$ be the results returned from each sub-transaction which are represented as

XML-based data. The unified XML-based data, denoted *UXML*, is the final result derived from integrating these XML results, such that

$$UXML = \underset{i=1\ldots n}{\Delta} R(q_i)$$

where operator Δ denotes the integration process that can be either a merge or join operation of the XML-based data and the mapping of the physical concepts/properties to the virtual concepts/properties corresponding to the user's request.

The integration process can be classified into two categories as follows.

8.2.1 Single Source Integration

If the XML results returned to managing agent are obtained from a single source or a single sub-transaction, the transformation process will map the corresponding physical properties and data values of the XML results to the virtual properties and data values in the form of unified XML-based data as defined in Definition 8.7. An algorithm for single source integration is given in Appendix D.

Definition 8.7: Single source integration.
Let $R(q_a)$ be the returned results obtained from executing a sub-transaction q_a of a single source a, represented by a labeled tree, such that $R(q_a) = \{A_i \mid \forall i=1\ldots n\}$ is a finite set of records at the leaf nodes of the tree, where each record $A_i = \{<PPN_x, PPD_x> \mid \forall x = 1\ldots m\}$ is a finite set of physical property name PPN_i and its data value PPD_i pair.

The unified XML-based data *UXML* is generated from mapping $R(q_a)$ to *UXML* such that $R(q_a) \rightarrow UXML$ and $UXML = \{X_i \mid \forall i=1\ldots n\}$, where $X_i = \{<VP_j, VPD_j> \mid \forall j = 1\ldots m\}$ is a finite set of virtual property VP_i and its data value VPD_i pair. The VP_i and VPD_i are obtained from mapping $PPN_i \rightarrow VP_i$ and $PPD_i \rightarrow VPD_i$, respectively.

The following is an example of the XML returned results of the global transaction in Figure 8.2 (b) that are sent from a single source as illustrated in Figures 8.6 (a) and (b). For example, in Figure 8.6 (b), there are two records returned from `Source2`. The unified XML-based data generated from integrating the above two records as illustrated in Figures 8.7 (a) and (b) becomes a two element array X[I], I =1…2 that take the form

X[1] = {("st_id", "11111"), ("st_name", "David"), ("position", "Prof.")}

X[2] = {("st_id", "12211"), ("st_name", "John"), ("position", "Asst.Prof.")}

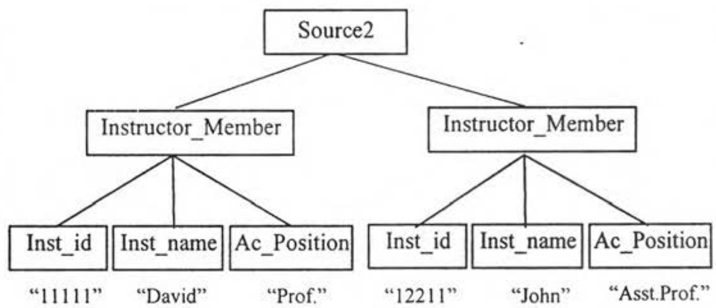
The set *UXML* is generated from joining the array of X[I] as follow:

$$UXML = \bigcup_{I=1,2} X[I]$$

That is, *UXML* = {{("st_id", "11111"), ("st_name", "David"), ("position", "Prof.")}, {("st_id", "12211"), ("st_name", "John"), ("position", "Asst.Prof.")}}

```
<!DOCTYPE  Source2 [
<!ELEMENT Source2(Instructor_Member)+>
<!ELEMENT Instructor_Member (Inst_name,
Ac_Position)>
<!ATTLIST Inst_id   ID        #REQUIRED>
<!ELEMENT  Inst_name          (#PCDATA)>
<!ELEMENT  Ac_Position        (#PCDATA)>
]>
```

(a) XML-DTD of returned results.

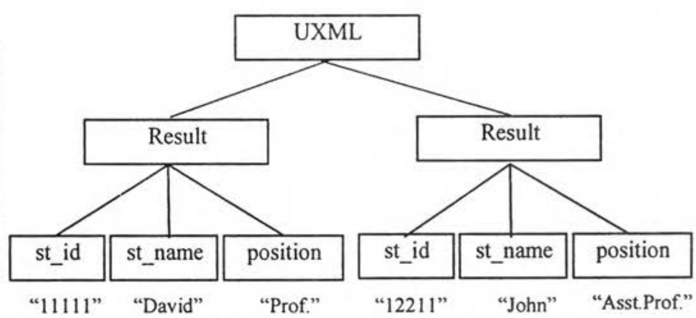


(b) XML tree structure representing XML document conforming to the XML-DTD in (a).

Figure 8.6 The XML returned results from Source2 to be sent to the managing agent.

```
<!DOCTYPE  UXML [
<!ELEMENT UXML(Result)+>
<!ELEMENT Result (st_id, st_name, position)>
<!ELEMENT st_id               (#PCDATA)>
<!ELEMENT  st_name            (#PCDATA)>
<!ELEMENT  position           (#PCDATA)>
]>
```

(a) XML-DTD of the unified XML-based data.



(b) XML tree structure representing XML document conforming to the XML-DTD in (a).

Figure 8.7 The unified XML-based data generated from the managing agent.

8.2.2 Multiple Sources Integration

To provide a flexible integration of the XML results obtained from multiple sources, a key or ID denoting each XML result is required for proper identification of the designated XML record. Each record $A_c \in R(q_a)$ contains key properties and non-key properties. Let Ka_c be a finite set of key properties of the record A_c such that $Ka_c \subseteq A_c$, and Xa_c be a finite set of non-key properties of the record A_c such that $Xa_c \subseteq A_c$ and $Ka_c \cap Xa_c = \phi$.

For each tree R_i, the physical concepts/properties in each record will be mapped to virtual concepts/properties by acquiring the mapping information from the metadata dictionary. The mapping of physical concepts/properties of each tree to virtual concepts/properties of the unified XML-based data is defined in Definition 8.8.

Definition 8.8: Multiple source integration

Given the returned results $R(q_a)$ and $R(q_b)$ being sent to the managing agent, let $A_c \in R(q_a)$ be a record in $R(q_a)$ and $B_d \in R(q_b)$ be a record in $R(q_b)$. Each $PPN_k \in A_c$ and $PPN_m \in B_d$ is searched for its corresponding virtual property in the metadata dictionary. If any PPN_k and PPN_m are children of the same parent virtual property and contain the same data values, these terms will be treated as synonymous terms and combined with the parent virtual property. In other words, $PPN_k \sim PPN_m$ iff $ChildOf(PPN_k, VP_t) \wedge ChildOf(PPN_m, VP_t)$, and $PPD_k = PPD_m$ such that PPN_k and PPN_m and their data values are integrated into a pair of $<VP_t, VPD_k>$ in the unified XML-based data.

For example, the `Staff_name` in `Source1` and `Inst_name` in `Source2` are synonymous since they are children of the same virtual property `st_name` and both contain the same data value. These synonymous terms are combined into `st_name` in the unified XML-based data.

The multiple sources integration process can be classified into two cases, namely, merging and joining the XML results.

(1) Merging the XML results. Given the individual result of a sub-transaction previously decomposed from the join predicates of a global transaction, each result holds the corresponding physical concepts/properties residing in the same source. The merging process will combine all the properties and data values of the physical property in each record from the labeled tree R_i, despite some differences in the key properties of each record. The process begins by mapping the physical property and data value from each record of the labeled tree R_i to a pair of virtual property and data value of the virtual property. For each record, if the mapping key properties and data values of each record are the same, these records are merged into the unified XML-based data. The records that have different mapping key properties and data values from other trees are also merged into the unified XML-based data with slight variation treatments. The merging process is defined in Definition 8.9.

Definition 8.9 Merging of the XML results.

Given the returned results $R(q_a)$. Let A_a be a record in $R(q_a)$ and $K_a = \{<VP_i, VPD_i> \mid \forall i = 1\ldots n\}$ be a finite set of virtual property VP_k and data value VPD_k obtained from mapping the key property PPN_k of A_a to VP_k, and PPD_k of that key property to VPD_k.

Let $X_a = \{<VP_j, VPD_j> \mid \forall j = 1\ldots m\}$ be a finite set of virtual property and data value pair. The VP_c obtained from mapping the non-key property PPN_c of A_a to VP_c, and VPD_c obtained from mapping PPD_c of that non-key property to VPD_c.

Given the returned results $R(q_b)$. Let B_b be a record in $R(q_b)$ and $K_b = \{<VP_i, VPD_i> \mid \forall i = 1\ldots n\}$ be a finite set of a pair of virtual property VP_k and data value VPD_k obtained from mapping the key property PPN_k of B_b to VP_k, and PPD_k of that key property to VPD_k.

Let $X_b = \{<VP_j, VPD_j> \mid \forall j = 1\ldots m\}$ be a finite set of virtual property and data value pair. The VP_c obtained from mapping the non-key property PPN_c of B_b to VP_c, and VPD_c obtained from mapping PPD_c of that non-key property to VPD_c.

There are four possible cases for merging consideration of each record of the labeled tree R_i, that is,

(i) If $(K_a == K_b) \wedge (X_a == X_b)$ then

Add $(K_a \cup X_a)$ to *UXML*, if $(VP_c \text{ in } K_a) \in S$ of the global transaction, or

Add X_a to *UXML*, if $(VP_c \text{ in } K_a) \notin S$ of the global transaction.

(ii) If $(K_a == K_b) \wedge (X_a \subset X_b)$ then

Add $(K_b \cup X_b)$ to *UXML*, if $(VP_c \text{ in } K_b) \in S$ of the global transaction, or

Add X_b to *UXML*, if $(VP_c \text{ in } K_b) \notin S$ of the global transaction.

(iii) If $(K_a == K_b) \wedge (X_b \subset X_a)$ then

Add $(K_a \cup X_a)$ to *UXML*, if $(VP_c \text{ in } K_a) \in S$ of the global transaction, or

Add X_a to *UXML*, if $(VP_c \text{ in } K_a) \notin S$ of the global transaction.

(iv) If $(K_a \neq (\forall K_b \in R(q_b)))$ then

Add $(K_a \cup X_a)$ to *UXML*, if $(VP_c \text{ in } K_a) \in S$ of the global transaction, or

Add X_a to *UXML*, if $(VP_c \text{ in } K_a) \notin S$ of the global transaction.

The process of comparing and merging of the XML results yields the unified XML-based data which is carried out by the algorithms given in Appendix E. Note that there will not be duplicated record being added to the UXML by virtue of set principles.

An example of integrating the XML results that are sent from multiple sources `source1` and `source2` based on the global transaction in Figure 8.3 are illustrated in Figures 8.8 and 8.9. The XML results are represented by the labeled trees R_1 and R_2 as illustrated in Figures 8.8 (a) and (b), respectively. From this example, `Source1` returns two records of data values, whilst `Source2` returns one record. The tree R_1 and R_2 contains the sets Ka_i and Xa_i, $\forall i = 1 \ldots n$, such that each Ka_i is a finite set of virtual property and data value pair, which in turn are mapped from the physical key property and its data value. On the other hand, the set Xa_i is a finite set of virtual property and data value pair, which are mapped from a physical non-key property and its data value. Hence, the first record of tree R_1 contains a finite set of key Ka_1 and non-key Xa_1, that is,

$Ka_1 = \{$ ("st_id", "22211") $\}$, and

$Xa_1 = \{$("st_name", "Anna"), ("st_salary", "11000"), ("dept_name", "Personnel")$\}$

The second record contains a finite set of key Ka_2 and non-key Xa_2, that is,

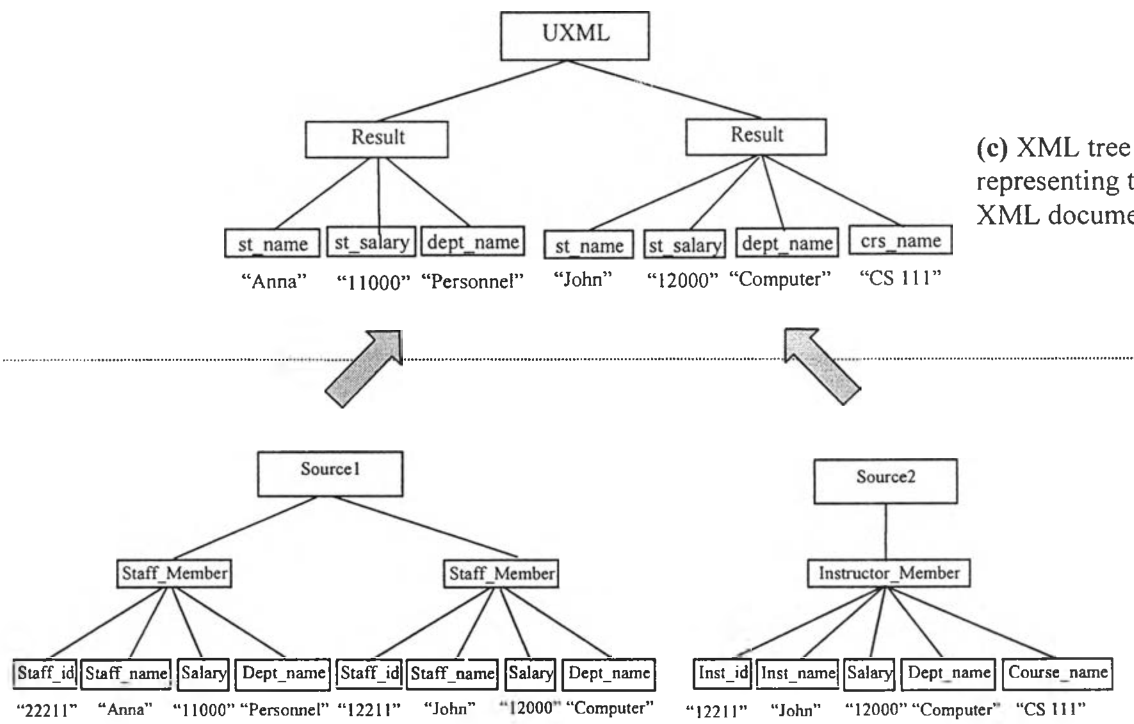
$Ka_2 = \{$ ("st_id", "12211") $\}$, and

$Xa_2 = \{$("st_name", "John"), ("st_salary", "12000"), ("dept_name", "Computer")$\}$.

For the tree R_2, only one record contains a finite set of key Kb_1 and non-key Xb_1, that is,

$Kb_1 = \{$ ("st_id", "12211") $\}$, and

$Xb_1 = \{$("st_name", "John"), ("st_salary", "12000"), ("dept_name", "Computer"), ("crs_name", "CS 111")$\}\}$.



(c) XML tree structure representing the unified XML document.

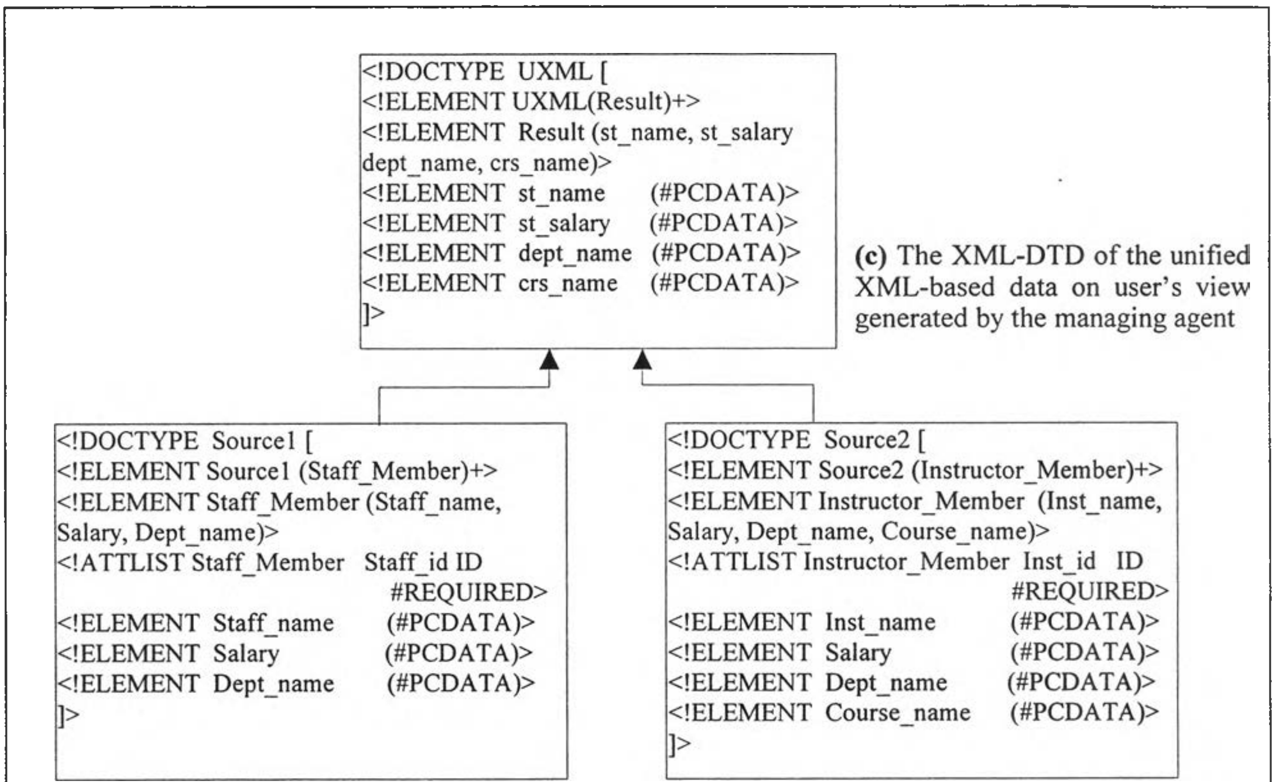
(a) XML tree structure (R_1) representing XML document sending from source1.

(b) XML tree structure (R_2) representing XML document sending from source2.

Figure 8.8 Multiple sources integration by merging the XML documents into the unified XML document.

The integration process will join the records obtained from each tree that have the same set of key Ka_c and Kb_k. For the first record of each tree R_1 and R_2, since $Ka_1 \neq Kb_1$ and `st_id` is not designated in the global transaction, that is, `st_id` $\notin S$, only Xa_1 is added to the set *UXML*. For the next record of tree R_1, since $Ka_2 = Kb_1$ and $Xa_2 \subset Xb_1$, thus Xb_1 is added to the set *UXML*. Therefore, the unified XML-based data becomes

UXML = { {("st_name", "Anna"), ("st_salary", "11000"), ("dept_name", "Personnel")},
{("st_name", "John"), ("st_salary", "12000"), ("dept_name", "Computer"),
("crs_name", "CS 111")}}



(c) The XML-DTD of the unified XML-based data on user's view generated by the managing agent

(a) The resulting XML-DTD generated from Source 1.

(b) The resulting XML-DTD generated from Source 2.

Figure 8.9 Multiple sources integration by merging the XML-DTD of each source into the unified XML-DTD.

(2) Joining the XML results. This process occurs when the join predicates of a global transaction are decomposed into join predicates of individual sub-transaction that consists of the corresponding physical concepts/properties residing in the different sources. In this process, only the records with the same physical key properties and data values are joined to form the unified XML-based data. The joining process is defined in Definition 8.10.

Definition 8.10 Joining of the XML results.

Given the results $R(q_a)$ and $R(q_b)$. Let A_a be a record in $R(q_a)$ and B_b be a record in $R(q_b)$, K_a, K_b, be the set of mapping key properties of records A_a and B_b. Let X_a, and X_b be the set of mapping non-key properties of records A_a and B_b, respectively, as given in Definition 8.9. The joining of each record in among the labeled tree R_i occurs when the set K_a is matched with the set K_b, that is,

If ($K_a == K_b$) then

Add $(K_a \cup Xa \cup Xb)$ to *UXML*, if $(VP_c \text{ in } K_a) \in S$ of the global transaction, or

Add $(Xa \cup Xb)$ to *UXML*, if $(VP_c \text{ in } K_a) \notin S$ of the global transaction.

The process of comparing and joining of the XML results to form the unified XML-based data is carried out by the algorithms given in Appendix F.

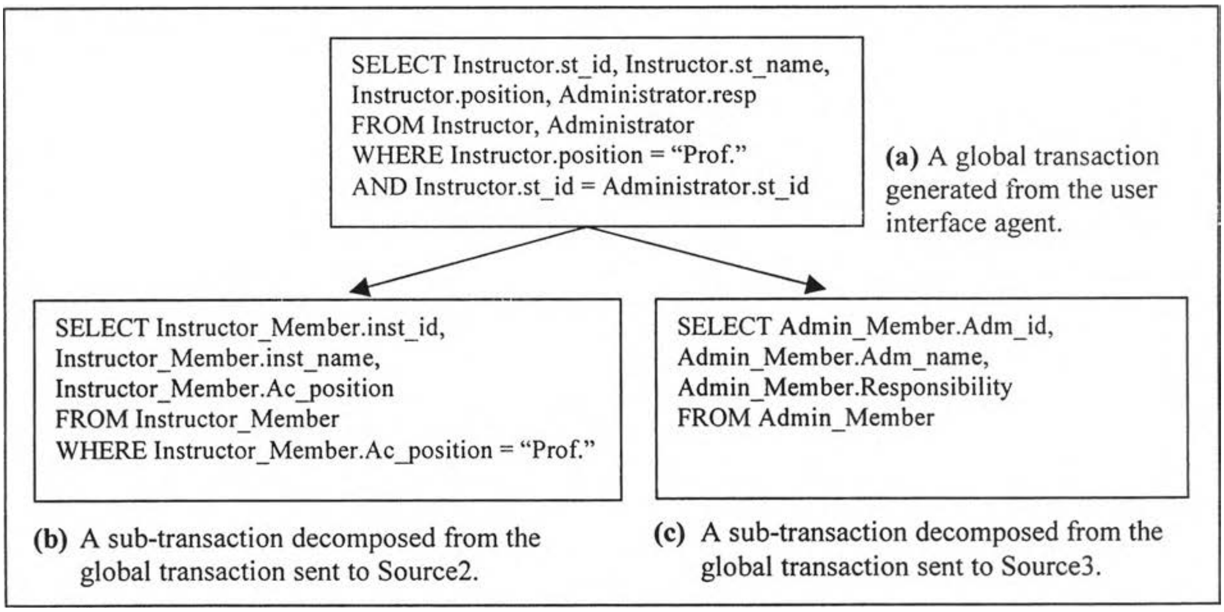


Figure 8.10 An example of the global transaction decomposition into sub-transactions.

The example in Figure 8.10 (a) depicts a global transaction that selects properties from the virtual concepts `Instructor` and `Administrator`. This example illustrates a partial IS-A relationship, where some (not all) instructors are administrators and some administrators are instructors. However, these concepts are sub-concepts of the concept `Staff`. If the physical concepts of these virtual concepts reside in different sources, the sub-transactions will be generated without the join predicate of these physical concepts as shown in Figures 8.10 (b) and (c). The returned results from each sub-transaction will subsequently be joined into a unified XML result.

Examples of integrating the XML documents and DTDs that are sent from multiple sources `Source2` and `Source3` based on the global transaction in Figure 8.10 (a) are illustrated in Figures 8.11 and 8.12. The XML results are represented as the labeled trees R_1 and R_2 in Figures 8.11 (a) and (b), respectively. In this example, both `Source2` and `Source3` return two records shown below.

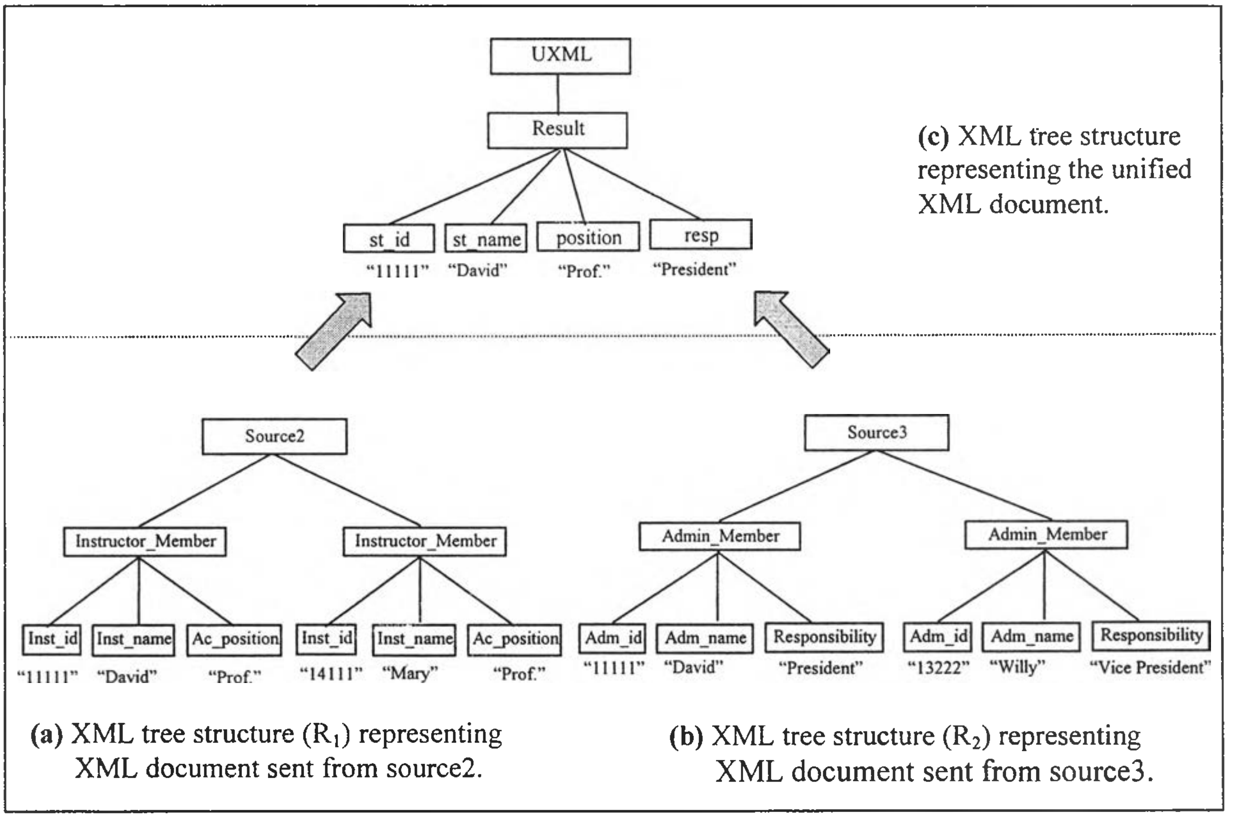


Figure 8.11 Multiple sources integration by joining the XML documents into the unified XML document.

For the tree R_1, the first record contains a finite set of key Ka_1 and non-key Xa_1, that is,

Ka_1 = { ("st_id", "11111") }, and

Xa_1 = {("st_name", "David"), ("position", "Prof.")}

The second record contains a finite set of key Ka_2 and non-key Xa_2, that is,

Ka_2 = { ("st_id", "14111") }, and

Xa_2 = {("st_name", "Mary"), ("position", "Prof.")}.

For the tree R_2, the first record contains a finite set of key Kb_1 and non-key Xb_1, that is,

Kb_1 = { ("st_id", "11111") }, and

Xb_1 = {("st_name", "David"), ("resp", "President")}.

The second record contains a finite set of key Kb_2 and non-key Xb_2, that is,

Kb_2 = { ("st_id", "13222") }, and

Xb_2 = {("st_name", "Willy"), ("resp", "Vice President")}.

For this example, only the first record of each tree R_1 and R_2 will be joined according to $Ka_1 = Kb_1$. Since `st_id` is the designated virtual property in the global transaction, that is, `st_id` $\in S$, therefore $Ka_1 \cup Xa_1 \cup Xb_1$ are added to the set *UXML*. The unified XML-based data becomes

UXML = {{("st_id", "11111"), ("st_name", "David"), ("position", "Prof."), ("resp","President")}}

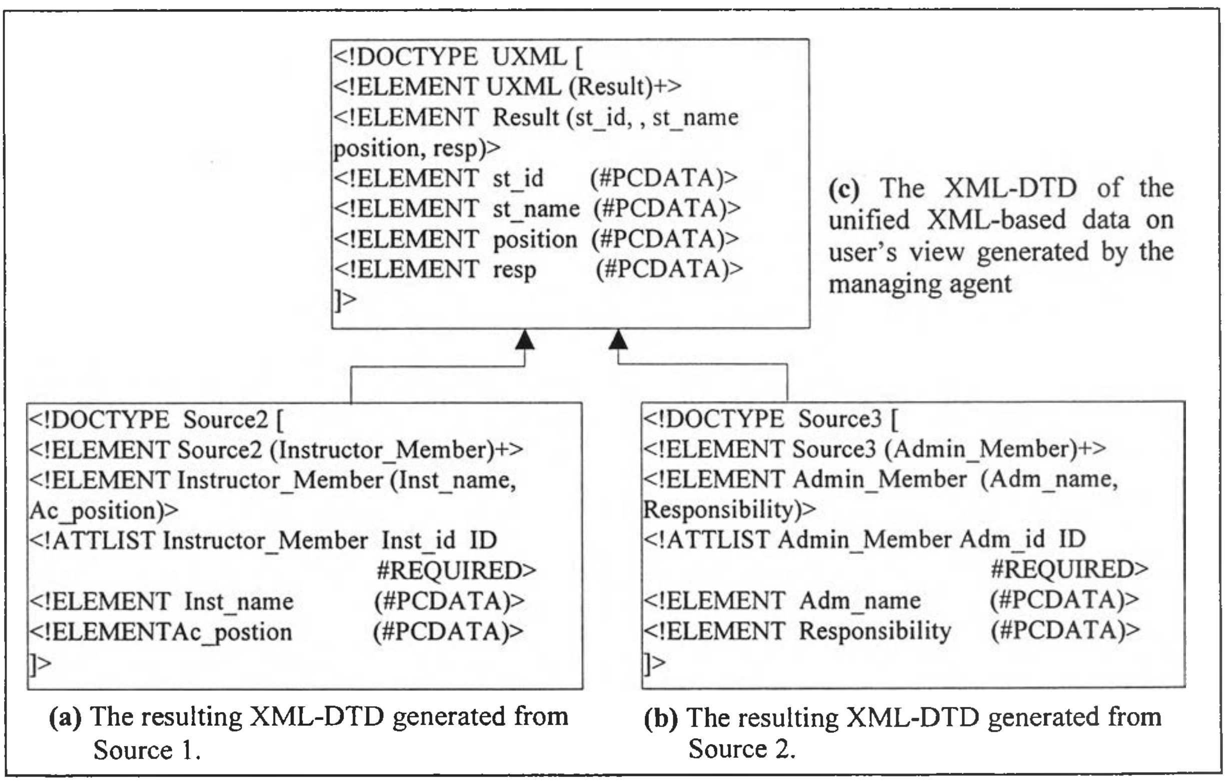


Figure 8.12 Multiple sources integration by joining the XML-DTD of each source into the unified XML-DTD.

8.3 The Query Validation

In order to ensure that the query process returns the relevant answers, query validation is required. The validation process is carried out in two steps, namely, query requirement correctness validation and result correctness validation.

8.3.1 The Query Requirement Correctness Validation

This process is carried out at the first step of the global transaction creation during the query normalization process. The objective is to match the requested virtual properties/concepts from the structural requirements of a user's query with the virtual properties/concepts residing in the metadata dictionary.

Definition 8.11 Validation of the query correctness.

Given a user's query $U(q)$ containing the set of target properties $U(p) = \{c_i.p_{ij} \mid \forall i = 1,\ldots, n, \forall j = 1,\ldots, m\}$ and target concepts $U(c) = \{c_j \mid \forall j = 1,\ldots, m\}$. Let $C = \{vc_i \mid \forall i = 1,\ldots, n\}$ be a finite set of virtual concepts in the metadata dictionary, and $P(vc_k) = \{vp_{kj} \mid \forall j = 1\ldots m\}$ be a finite set of virtual properties of the virtual concept vc_k, the set $U(q)$ is correct if $\forall c_i \in C$ and $\forall p_{ij} \in P(vc_i)$, where $\forall i = 1,\ldots, n, \forall j = 1,\ldots, m$.

8.3.2 The Result Correctness Validation

This process takes place after the unified XML-based data is generated. The result correctness aims to verify that the virtual properties of the unified XML-based data match the requested virtual properties of the global transaction. The validation algorithms are given in Appendix G.

Definition 8.12 Validation of the result correctness.

Given a unified XML-based data $UXML = \{X_i \mid \forall i=1,\ldots, n\}$, such that each $X_k = \{<VP_j, VPD_j> \mid \forall j = 1,\ldots, m\}$ is a finite set of virtual property and data value pair. Let $UDTD = \{v_i \mid \forall i=1,\ldots, n\}$ be a finite set of virtual properties in $UXML$ and $SEL = \{s_j \mid \forall j=1,\ldots, m\}$ be a finite set of selected virtual properties s_j (or attributes) in the SELECT clause of a global transaction Q. The unified XML-based data is correct if set $UDTD = SEL$.