SOFTWARE–HARDWARE SYSTEMS

GRAPH QUERIES FOR DATA INTEGRATION
USING XML

V. G. Tulchinsky, A. K. Yushchenko, and R. A. Yushchenko

A new visual language, which is called ER-QBE and uses a conceptual model semantics, is proposed for the development of XML views. ER-QBE is based on graph queries that are trees of parametric SQL queries. The ER-QBE language is characterized by formally proved completeness, its expressiveness is higher than that of well-known alternatives, and it supports both relational and object-relational databases.

Keywords: XML language, database (DB), representations of DBs, QBE language, XML representation, semantic network.

INTRODUCTION

XML [1] is a popular document markup language that allows one to structure information using an arbitrary collection of instructions. The structure of XML documents is treelike. Main XML applications include Web services, Internet applications, data storehouses, and documents circulation systems. An advantage of XML is its text representation that does not require special programs for creation, examination, and editing. At the same time, as a rule, the XML format is more cumbersome in comparison with binary formats with similar content.

In contrast to HTML, XML is not a language for coding definite information. It is used for the representation of arbitrary hierarchies of data expressed in text form. Each concrete XML application generates a formally defined language or a dialect. There are many standardized XML dialects, for example, MathML, RSS, GpaphML, XHTML, etc. Such dialects are also specifications of the schema of an XML document, for example, DSD, DTD, XSD, and XDR [2]. An XML parser gives an error message when the content of an XML document does not correspond to its declared structure. The specification of a schema can be built into an XML document (in this case, it describes itself) or can be stored in the form of a file (this is convenient if different documents must have an identical structure). In practice, the following intermediate solution is often used: a schema specification built into an XML document refers to terms defined in other files. The World Wide Web Consortium (W3C) offers tools for XML data processing that provide high-performance capabilities in integrating data of different XML documents, searching for concrete information, and transforming XML trees.

At the present time, most data are stored in relational DBs. To simplify the development of Internet applications, interprogram interaction, and data integration, the export of DB data into XML is topical. Manufacturers of DBMSs offer special tools for solving this problem. The extension SilkRoute [3] of the DBMS DB2 (IBM) efficiently computes instantaneous representations described in the language RXL. Some other development of the same group, namely, XPERANTO [4], is oriented toward object-relational bases. Starting from Version 2000, Microsoft SQL Servers makes it possible to specify both instantaneous and virtual (dynamic) XML representations [5]. To this end, the SQL extension called FOR XML was developed and then used as the basis for the SQL/XML standard [6]. At the same time, efforts were undertaken to manipulate relational data using the query languages XML and XML-QL [7] and more flexible XQuery [8] developed by W3C. To represent DB tables in XPath queries in the form of virtual XML documents, the language XDR is used whose completeness and expressiveness are worse than those of RXL but that allows one to sufficiently easy describe...
the mapping of a table schema into the corresponding virtual XML document schema. However, the above standard tools are subjected to the following constraints:

- in XPath/XDR, DB tables are represented by individual virtual XML documents and, hence, the missing semantics should be written directly in a transformation query, which significantly complicates the query;
- all the data items exported into an XML document by means of SQL/XML must be represented by one join, which reduces the efficiency of realization of complicated XML representations and increases resource consumption;
- traditional query development automation tools do not support XML representations, which complicates their programming and increases the probability of errors.

In [9, 10], it is proposed to reconstruct an extended entity-relationship model from a schema according to templates and, based on this model, to describe an XML tree in the form of a subschema or in terms of the language of embedded relations. A drawback of the representation of XML in the language of embedded relations is the necessity of a special nonrelational SQL extension. In [11], the following simple heuristic solution is proposed: the description of an XML tree by a tree of SQL queries. Graph queries [12, 13] combine the advantages of the proposed solutions by providing a visual technology for the development of query trees on the basis of a conceptual model.

1. GRAPH QUERIES

A graph query (GQ) is a means of selection and visualization of related data fragments forming a network structure. Such data structures are widely used in information systems at the level of conceptual models (for example, entity-relationship diagrams or object-relationship diagrams). The selection and visualization of network fragments (subgraphs) in a GQ are based on traversal trees. Main distinctive features of a GQ, namely, the use of the semantics of a conceptual model and a result in the form of a tree, directly relate them to the problem of formation of XML representations.

1.1. A structured semantic network (SSN) is a special data model that allows one to abstract from concrete distinctive features of network conceptual models. SSNs are based on semantic networks, i.e., on a well-known knowledge representation model. In contrast to data models used in the business sphere, semantic networks are oriented toward the systematization of general character knowledge. The types of nodes can be instances and classes. SSNs are characterized by a strict typification of nodes and edges of a network. In the case of SSNs, a semantic network subgraph consisting of nodes-classes and edges that connect these nodes is considered as an individual network that plays the role of a schema; this brings SSNs closer to classical data models. A schema forms the skeleton of an SSN.

Structurally, an SSN is a colored digraph whose nodes are entities of a model of an object domain (MOdB) and edges represent relationships between entities. Its complement is a schema in the form of a colored digraph of classes. An edge connecting nodes-classes is called a binary relation, and an edge connecting nodes-instances is called an assertion. Entity-relationship and object-relationship models are easily reduced to SSNs, which allows one to immediately extend further results to data described by ER diagrams, diagrams of classes, or UML sequences. XML schemas also specify SSNs. (In all the above cases, it makes sense to assign entities/objects/elements to SSN nodes and XML attributes to SSN attributes.)

Questions of reconstruction of an absent conceptual model from the schema of a relational or an object-oriented database are considered in [10, 14, 15].

1.2. Grammatical representation of SSNs. We specify a nonempty set of root nodes of an SSN \( T_0 \subseteq E \). We introduce a set of nonterminal symbols \( N_E \) and define a biunique mapping \( n:E \leftrightarrow N_E \).

Let us consider a grammar \( \Gamma = (N = N_E \cup \{A\}, T = E \cup L, A, P = \{P_E \cup P_E^S \cup P_L \cup P_E^S\} \) in which to each instance are assigned terminal and nonterminal symbols and to each root node of the SSN and to each assertion of the SSN correspond productions of four types, namely, \( P_E = \{p_1^E : A \rightarrow s \mid s \in T_0, S = n(s)\}, P_E^S = \{p_2^E : A \rightarrow s \mid s \in T_0\}, P_L^* = \{p_3^E : S \rightarrow r \mid r = (\alpha, s, t) \in L, S = n(s)\} \), and \( P_L^* = \{p_4^E : S \rightarrow rT \mid r = (\alpha, s, t) \in L, S = n(s), T = n(t)\} \). We call such a right linear grammar an SSN grammar.

The set of strings of the language of the SSN grammar \( \Gamma \) is identical to the set of the SSN traversal paths outgoing from the root nodes of the SSN.

A grammar \( \Gamma_2 \) is directly coupled with a grammar \( \Gamma_1 \) if the productions of the grammar \( \Gamma_2 \) are marked by labels from among the terminal/nonterminal symbols of the grammar \( \Gamma_1 \) and, at the same time, for at least some strings derived in \( \Gamma_1 \), all the symbols of each string are labels of productions of \( \Gamma_2 \) and uniquely determine the sequence of productions that derive