It has been less than two years since the bulletin had a special issue on XML (September, 1999), but quite a bit has happened. Two years ago we were just beginning to consider the issues of manipulating XML as data. The issue of developing an XML query language was taking center stage, as were initial attempts to store XML data in relational databases. Since then, a working group of the World-Wide Web Consortium has been making rapid progress on developing XQuery, a standard query language for XML. Most relational database vendors already support some form of storage of XML in their products with limited capabilities for querying. In addition, several products support native XML storage (e.g., eXcelon and Tamino) and several companies have been using XML as a platform for integrating data from multiple sources (e.g., Nimble Technology, Enosys Markets).

This issue features several papers that touch on the current topics and debates with respect to XML data management. Naturally, one of the main questions facing the community today is whether XML data management should be added as another feature to relational database systems or whether we should develop new *native* database systems. The proponents of the relational approach point to the success of relational database vendors in incorporating previous non-relational features into their systems and the advantages of exploiting all the infrastructure of existing products. The advocates of native XML management systems argue that XML requires considerably different processing techniques that are impossible to support on top of relational systems, and that existing relational systems are too heavyweight to support the kinds of applications involving XML. In addition to this debate, much of the focus of the research community has been on exploring new types of query services that need to be supported in XML applications. These include processing queries where the data is being integrated from several sources on the WWW, the data is *streaming* from the sources, supporting large numbers of continual queries and subscription queries, and combining features of querying from information retrieval and databases.

The first two papers in the issue consider the relationship of XML data management and relational databases. In the first paper, Rys describes the XML support added to SQL Server 2000. In the second paper, Fernandez et al. describe Silkroute, a system that efficiently supports XML views over relational data. Readers are also referred to the Xperanto Project at IBM Almaden that has many of the same goals.

The next papers describe three of the main systems that explore native XML query processing, but an emphasis on integrating data from multiple sources on the network. Ives et al. describe the Tukwila System, which extends the key techniques from pipelined relational query processors to support XML query processing over networked bound data sources. Naughton et al. describe the Niagara System whose goal is to answer queries over large collections of external XML documents. The paper by Tufte and Maier describes a specific operator in Niagara for efficiently *merging* two streaming XML documents. The paper by the Xyleme group describes the Xyleme Project whose goal is to provide a dynamic warehouse of the XML data on the web and to support query evaluation and efficient update on the warehouse.

Finally, the paper by Kossmann and Florescu steps back from the specific query processing issues and identifies a new impedance mismatch that arises when programmers try to build applications that involve XML, relational databases and a programming language such as Java. The paper proposes an XML query language whose goal is to remove the impedance mismatch.