An XML Framework for Integrating Continuous Queries, Composite Event Detection, and Database Condition Monitoring for Multiple Data Streams

Susan D. Urban[1], Suzanne W. Dietrich [1, 2], and Yi Chen[1]
Arizona State University
[1]School of Computing and Informatics
Department of Computer Science and Engineering
Tempe, AZ 85287-8809
[2]Department of Mathematical Sciences and Applied Computing
Phoenix, AZ 85069-7100
s.urban@asu.edu dietrich@asu.edu yi@asu.edu

Extended Abstract

Current, data-driven applications have become more dynamic in nature, with the need to respond to events generated from distributed sources or to react to information extracted from incoming data streams. Event processing and stream processing have traditionally developed as two separate areas of research. Event processing has its roots in research with active rule processing (Widom and Ceri, 1996) as well as distributed systems (Muhl et al., 2006), with a focus on composite event specification languages and execution issues for detecting, broadcasting, and consuming streams of events. More recently, data stream processing has developed as a new form of data management, with a focus on the continuous execution of queries over data generated from sensors or other sources that emit streams of data that must be quickly analyzed (Golab and Ozsu, 2003; Arasu et al., 2003). Research on data streams mainly focuses on continuous (and potentially infinite) sequences of data, investigating query processing techniques that can be "localized" to recently received streaming data using sliding windows to handle the temporal aspects of the stream. Continuous queries for streaming data are similar to past work with condition monitoring for persistent data (Rosenthal et al., 1989). Condition monitoring has been used in the context of condition-action rules in rule processing environments to determine data changes, known as deltas, that affect the truth value of the condition and to incrementally evaluate the query of the condition for efficiency.

There are many complementary and synergistic relationships among these different but related areas of research. In fact, it is our belief that the future of dynamic, data-driven applications is found in the integration of techniques from composite event processing, continuous query processing for data streams, and condition monitoring over persistent data. As motivation for the integration of these research areas, consider the following application scenarios from the medical and financial domains:

Medical Scenario: Medical monitoring devices often suffer from a condition know as alert fatigue, where too many false alarms are generated. As a result, clinicians may ignore the alarms that are generated by such devices. Since a patient can be given an agent intravenously that causes transient physical properties, the monitoring process can be improved through the use of contextual information. For example, chemotherapy patients are sometimes given chemotherapeutic agents that will cause the white blood cell counts, red blood cell counts, or platelet counts to drop two to four weeks after the treatment. The decreases in these counts, however, should not last for more than six weeks. Alert fatigue can potentially be avoided by coupling the monitoring of streaming data from physiological monitors with appropriate contextual queries over a patient's electronic medical record.

Shopping and Credit Card Monitoring Scenario: Monitoring consumer activity can provide a wealth of information to suppliers, retail businesses, banks, and credit card companies. Suppliers could monitor streams of information about sales of specific items from clients. Coupled with client history and rules about seasonal sales, this information can be used to plan for future supply chain demand. Department stores can monitor the types and frequency of customer purchases to offer customers special deals at the point of sale or to assist call centers and service desks with information about customer history and preferences. Businesses, banks, and credit card companies can also work together to monitor sales, complaint, and return activity that could indicate fraudulent behavior.

These are two different applications, and yet we can extract and generalize several similar requirements from each scenario. Both applications involve the analysis of multiple streams of data. In one case, the data is arriving from sensors attached to a patient; in the other the data is arriving from multiple streams of events that are generated by application software. Both applications require extracting information (i.e., events) from one or more, possibly distributed, data streams. The extracted information must be analyzed and possibly combined with other events in meaningful ways.

In both applications, however, the application context is extremely important. In the medical scenario, we can avoid alert fatigue by combining the analysis of sensor data with application rules that are coupled with information stored in a patient's persistent, electronic medical record. In the shopping and credit card monitoring scenario, we can assist call processing centers, employees at service desks, or point-of-sale cashiers in rewarding good customers or detecting fraudulent behavior by analyzing events in the context of queries and application rules against customer preferences, history, and credit card data. Furthermore, there are temporal aspects to each application scenario, where the timeframe for the analysis of the data and the composition of events is important.

As illustrated by the above application scenarios, ideally, research in the areas of event and stream processing must converge to provide a more effective way of extracting meaningful events from multiple, distributed data streams. In response to this need, our research is addressing the development of *Distributed Event Processing Agents (DEPAs)* that support streaming, event-driven applications in the context of application knowledge. DEPAs provide a conceptual language for the expression of composite events, an underlying XML framework for filtering and processing composite events, and a distributed execution environment for processing events from multiple streams.

The DEPA conceptual language framework is based on our past work with the Composite Event Definition Language (CEDL) (Biswas, 2005; Urban et al., 2006), which provides expressive filtering capabilities on both primitive and composite events. The event operators of CEDL include the OR, AND, SEQ, and TIMES operators (Chakravarthy and Mishra, 1994; Gatziu and Dittrich, 1993). CEDL uses the *recent* (latest) and *continuous* selection modes from (Chakravarthy and Mishra, 1994) for use with the AND, OR, and SEQ event operators. The *cumulative* selection mode is automatically provided with the use of the TIMES event, where all parameter values of the same event type are formed into a collection associated with a single occurrence of the TIMES event. The unique aspect of CEDL is the support it provides for filtering of primitive events as well as filtering of composite events and their associated aggregate values and timelines (Urban et al., 2006). CEDL supports the specification of basic parameter filters on primitive events; basic parameter filters on composite events, with the ability to compare parameter values from the different events that compose a composite event; time filters that limit the lifetime of composite event detection; and indexed, aggregate and quantifier filters on cumulative composite event parameter values. Composite events can also be composed in a nested fashion to create more complex composite events.

As the underlying query execution framework, we are extending XQuery to create Composite XQuery (CXQ), which supports the expressive event filtering features of CEDL. An advantage of extending XQuery in the design of CXQ is that input data streams and databases can export an XML view, and queries on these various types of data can be expressed in XQuery. Therefore we can seamlessly query streams in relational tuple form and streams in XML form, as well as persistent relational databases that add application context in the filtering process. The CXQ engine is being designed as an extension of ViteX, a streaming XPath engine (Chen et al., 2006; Chen et al., 2005; Chen et al., 2004; Chen et al., 2002). The CXQ engine will integrate continuous queries with condition monitoring over persistent databases based on our past work with materialized view maintenance and condition monitoring in the deductive data model (Harrison and Dietrich, 1992; Harrison, 1992) and condition monitoring in the object-oriented data model (Sundermier et al., 1997; Sundermier, 1999). Our work also includes the development of an architecture for communication among multiple DEPAs for sharing events, filters, and context data sources.

Our approach to the integration of stream processing and event processing is essential to the support of applications such as those in the medical field for health monitoring, the financial domain for executive dashboards, supply chain and other B2B applications for monitoring consumer activity, or for autonomic behavior within computer systems and embedded systems. With XML becoming a standard for data representation, our research will provide extensions to XQuery that will allow the uniform framework of time-based composite event detection to streams of data and events in different formats. On a broader scale, our research will enhance the analysis of data and event streams through the use of database context filters and through the correlation of multiple streams, thus providing a way of extracting more meaningful, application-oriented events from streams of data in the support of dynamic, data-driven applications.

References

Arasu, A., B. Babcock, S. Babu, M. Datar, K. Ito, I. Nishizawa, J. Rosenstein and J. Widom. (2003) STREAM: The Stanford Stream Data Manager. *In Proc. of the 2003 ACM SIGMOD Intl. Conf. on Management of Data*.

Biswas, I. (2005) *A Composite Event Definition Language and Detection System for the Integration Rules Environment.* M.S. Thesis, Department of Computer Science and Engineering, Arizona State University.

Chakravarthy, S. and D. Mishra. (1994) SNOOP: An Expressive Event Specification Language for Active Databases, *Knowledge & Data Engineering Journal*, vol. 14, no. 10, (October), 1-26.

Chen, Y., S.B. Davidson and Y. Zheng. (2002) XKvalidator: A Constraint Validator for XML. *In Proceedings of 11th ACM Conference on Information and Knowledge Management* (CIKM), pp. 446-452.

Chen, Y., G. A. Mihaila, S. B. Davidson and S. Padmanabhan. (2004) EXPedite: A System for Encoded XML Processing. *In Proceedings of 13rd ACM Conference on Information and Knowledge Management* (CIKM), pp. 108-117.

Chen, Y., S.B. Davidson and Y. Zheng. (2005) ViteX: A Streaming XPath Processing System. Demonstration description. *In Proceedings of 21st International Conference on Data Engineering (ICDE)*, pp. 1118-1119.

Chen, Y., S.B. Davidson and Y. Zheng. (2006) An Efficient XPath Query Processor for XML Streams. *In Proceedings of 22nd International Conference on Data Engineering (ICDE)*.

Gatziu, S. and K. Dittrich. (1993) Events in an Active Object-Oriented Database System. *In Proceedings of the 1st International Workshop on Rules in Database Systems*, (Springer, September).

Golab, L. and T. Ozsu. (2003) Issues in Data Stream Management, *ACM SIGMOD Record*. 32(2).

Harrison, J. (1992) *Condition Monitoring in an Active Deductive Database*, Ph.D. Dissertation, Arizona State University, Department of Computer Science and Engineering, Summer.

Harrison, J. and S. W. Dietrich. (1992) Maintenance of Materialized Views in Deductive Databases: An Update Propagation Approach. *Proceedings of the Deductive Database Workshop in conjunction with the Joint International Conference and Symposium on Logic Programming*, Washington, D. C., November 1992, pp. 56-65.

Muhl, G., L. Fiege,and P.Pietzuch. (2006) Distributed Event Based Systems. Springer (Germany).

Rosenthal, A., U. S. Chakravarthy, B. Blaustein, and J. Blakely. (1989) Situation monitoring for active databases. *In Proceedings of the 15th international Conference on Very Large Data Bases* (Amsterdam, The Netherlands). Very Large Data Bases. Morgan Kaufmann Publishers, San Francisco, CA, pp.455-464.

Sundermier, A., T. B. Abdellatif, S. W. Dietrich and S. D. Urban. (1997) Object Deltas in an Active Database Development Environment. "In Proceedings of the 5th international Conference on Deductive and Object-Oriented Databases" (December 08 - 12, 1997). F. Bry, R. Ramakrishnan, and K. Ramamohanarao, Eds. Lecture Notes In Computer Science, vol. 1341. Springer-Verlag, London, 211-228.

Sundermier, A. (1999) "Condition Monitoring in an Active Deductive Object-Oriented Database." M.S. Thesis, Arizona State University.

Urban, S., Ingrid Biswas, Suzanne W. Dietrich. (2006). Filtering Features for a Composite Event Definition Language. *Proceedings of the International Symposium on Applications and the Internet*, Phoenix, Arizona.

Widom, J. and S. Ceri. (1996) *Active Database Systems: Triggers and Rules for Advanced Database Processing*, Morgan Kaufmann Publishers, San Francisco.