Demo: Altibase DSM: CTable for Pull-based Processing in SPE*

Jaemyung Kim, Vladimir Verjovkin, Sergey A. Fedorov,
Younghun Kim, Dae-Il Kim, Sungjin Kim
Altibase Corporation
182-13, Guro-dong, Guro-Gu
Seoul, 152-790, Korea
{ jmkim,verjovkin,fedorov,
yhkim,newdaily,sjkim }@altibase.com

Sang-Won Lee
School of Information & Communications Engr.
Sungkyunkwan University
Suwon, 440-746, Korea
swlee@skku.edu

ABSTRACT

We demonstrate streaming applications of the Altibase Data Stream Middleware (DSM), the distributed Stream Processing Engine using the Publish/Subscribe Communication Model. Altibase DSM has the Cached Table that is a key-value store supporting not only insert but also update and delete operations. The demos include the Sex-offender Tracking and Bus Arrival Information Systems.

Categories and Subject Descriptors

D.2.11 [**Software Architectures**]: Domain-specific architectures

General Terms

Design,Performance,Algorithm

1. INTRODUCTION

Recent advances in wireless technologies and mobile devices such as smartphones and tablet computers, and new applications including location-based and social network services generate a significant amount of data. These services involve online analytic processing, which requires a Stream Processing Engine (SPE) due to its performance benefits. Streaming applications are naturally distributed, and with increasing loads, the demand for distributed architecture [3, 5] has emerged. Advanced network infrastructures such as Gigabit Ethernet and InfiniBand (RDMA) also play a vital role in heavily loaded distributed systems.

While traditional DBMS uses pull-based processing, SPE uses push-based processing. Although SPE benefits from its push-based model, it often refers to static relational data from DBMS for meaningful analytic processing. Fetching data from DBMS can degrade the entire performance due to its frequent occurrence. Conversely, output stream events also need to be converted to relational tables or accessed through legacy applications designed for pull-based processing. Therefore, it entails additional implementation cost to connect two systems with different processing paradigm.

We implement the Altibase Data Stream Middleware (DSM), a distributed SPE with Publish/Subscribe Communication Model, because each distributed Event Processing Agent (EPA) can be loosely coupled. Altibase DSM includes the Cached Table (CTable) feature, a key-value store supporting not only insert but also update and delete operations. It can be solely used as a relational table and convert an event stream object into a CTable with a mapping function. For example, to minimize the latency in accessing a database table, CTable is able to replicate a table from the Altibase Hybrid Database (HDB). When DSM starts up, it transmits all HDB source table records. During operation, the HDB transaction log of DML can be captured and converted into DML operations of DSM. On the DSM side, to perform event processing, which refers to the DBMS table, the CTable replicated with a HDB's table can be joined with standard event streams and windows.

CTable can be regarded as a transition technology which reduces the introduction costs of SPEs to legacy systems. Unbounded streaming events are bound in the CTable with a unique up-to-date key constraint. DSM supports SQL-like languages with a DBC-style interface such as ODBC and JDBC. Furthermore, to facilitate integration with the legacy system, DSM additionally supports pull-based query in an open-fetch-close manner when the query includes only CTable(s) on the FROM clause. We explain the distributed architecture of the DSM and the Cached Table in Section 2 and demonstrate machine to machine (M2M) applications such as sex-offender tracking and bus arrival information in Section 3.

2. ALTIBASE DSM AND CTABLE

In this section, we describe the distributed architecture of Altibase DSM using the Publish/Subscribe communication model and Cached Table. This paper mainly describes the CTable and its applications, and briefly explains our distributed architecture.

The event-processing tasks of SPE can be represented as an Event Processing Network (EPN). EPN is an event flow, which is a collection of staged executions of Event Processing Agents (EPAs). Because Altibase DSM is designed to be distributed, it can have an EPN that spans multiple nodes (Figure 1).

While Borealis [1] tightly manages distributed nodes and uses the Medusa distribution logic, and TCP connections between EPAs and TCP multiplexing for optimization, we use the Publish/Subscribe Communication Model (Pub/Sub Model) for loosely coupled EPN programming, UDP, and multicast technology for scalability. EPAs are participants of the Pub/Sub Model that directs the flow of messages from senders to receivers based on receivers' data interests (Topic) [4]. To query distributed events, the event stream of the Altibase DSM is mapped on the Topic of the Pub/Sub Model. Using this mapping, physically separated stream queries (EPAs) can send and receive the events without socket programming.

*This work was supported in part by MKE, Korea under ITRC (NIPA-2011-(C1090-1121-0008)) and also supported in part by Seoul Metropolitan Government 'Seoul R&BD Program (PA090903)'

DEBS'11, July 11–15, 2011, New York, New York, USA.
ACM 978-1-4503-0423-8/11/07.

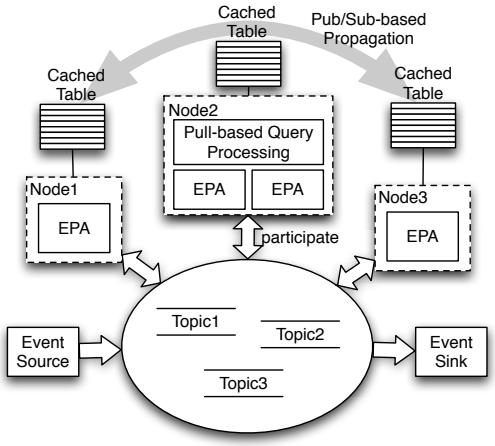

Figure 1: Publish/Subscribe-based Distribution

SPE converts streaming events into a window, which is a temporal relational table to perform online aggregation. The events from the window are then reconverted into streaming events. Bidirectional conversion between stream and relation is inevitably used for aggregation in SPE. Likewise, we believe that the manageability of relational tables in SPEs and transition between push-based to pull-based processing are required in use-cases mentioned.

As mentioned in [2], stream queries may join event streams and relational tables. Because tables are mostly stored in DBMS, many SPEs implement various input adapters like feeding data from DBMS and fetching tuples from DBMS using DBC interfaces. However, for this reason, the Altibase DSM support the relational table feature that can be updated from the DBMS tables. While a window for online aggregation is designed for append-only streaming events, the CTable can perform update and delete operations.

In addition, we found that legacy systems hardly change from pull-based to push-based processing architecture. Despite the benefits of SPE, customers and application developers complain about the difficulty in combining stream applications with legacy applications and prefer to use pull-based processing while minimizing the changes in legacy systems. To reduce this introduction cost, the CTable maintains the most recent record for each key to cache unbound streaming events, and, internally, it exploits the index structure, which is optimized for frequent update operations. CTable's recent values suit applications such as monitoring a dynamic dashboard containing stock market symbols [6]. For event-processing, DSM executes a set of queries before the events occur. Subsequently, when an event meets the query condition, DSM notifies a user (push-based processing). However, the queries containing only CTable(s) on the FROM clause are classified as 'CTable query' and are executed in pull-based processing (open-fetch-close method) through ODBC and JDBC interfaces.

3. DEMONSTRATION

We apply Altibase DSM to track sex offenders and as a bus arrival time information system.

Sex Offender Tracking System: Figure 2 is one of the machine-to-machine (M2M) communication service usages. Paroled sex offenders are obligated to stay in the district designated by law. For monitoring purposes, such individuals are obligated to wear an electronic anklet embedded with GPS and/or RFID for location tracking and a CDMA module for communication. Each offender has a different designated district, jurisdiction, and related organizations (e.g., Police Agency, Public Prosecutors' Office). This information is scattered across different databases and is joined using the location data of the offender. Altibase DSM joins the location tracking events and CTables from the remote databases.

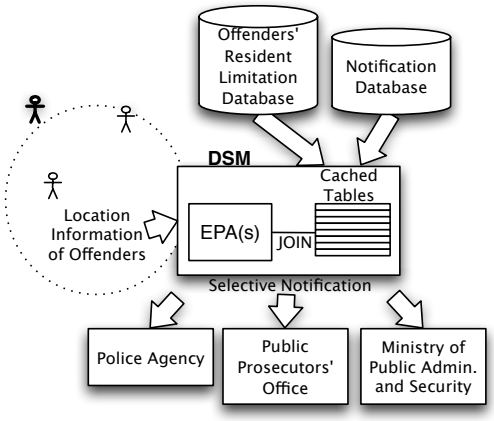

Figure 2: A Use Case for CTable

Bus Arrival Information System: In Seoul, South Korea, bus arrival information at each station and the current location of the buses for each route are provided online. Passengers are able to estimate the waiting time for a particular bus at the station. Possible implementations include 1) electric display at a bus station, 2) bus routes, and arrival information on a website, and 3) smartphone applications. The three abovementioned systems provide information to users in the pull-based processing mode, even though bus location events are processed in the push-based mode. To this end, the Altibase DSM and its CTable are applied.

4. ACKNOWLEDGMENTS

We would like to thank Dmitry Klimkin, Maxim Tomilov, and all other members of Altibase DSM Team for their efforts with development of this product. We are also grateful to Sungmin Kim, Sungil Bae, and Jae-Keoung Lim of Altibase for continuous supporting to our direction.

5. REFERENCES

[1] D. J. Abadi, Y. Ahmad, M. Balazinska, U. Cetintemel, M. Cherniack, J. Hwang, W. Lindner, A. S. Maskey, A. Rasin, E. Ryvkina, N. Tatbul, Y. Xing, and S. Zdonik. The design of the borealis stream processing engine. In *CIDR*, 2005.

[2] B. Babcock, S. Babu, M. Datar, R. Motwani, and J. Widom. Models and issues in data stream systems. In *In roceedings of the 21st ACM PODS*, pages 1–16, 2002.

[3] M. Cherniack, H. Balakrishnan, M. Balazinska, D. Carney, U. Çetintemel, Y. Xing, and S. B. Zdonik. Scalable distributed stream processing. In *CIDR*, 2003.

[4] Y. Diao and M. J. Franklin. Publish/subscribe over streams. In *Encyclopedia of Database Systems*, pages 2211–2216. 2009.

[5] J. M. Hellerstein and M. Stonebraker. *Readings in Database Systems: 4th Edition*, chapter 10. The MIT Press, 2005.

[6] A. Moga, I. Botan, and N. Tatbul. Upstream: Storage-centric load management for data streams with update semantics. Technical report, ETH Zurich, CS Dept., 2009.