제29회 한국정보처리학회 춘계학술발표대회 논문집 제15권 제1호 (2008. 5.)

비즈니스 이벤트 스트리밍 대한 연속 질의 처리

박영욱*, 홍봉희*, 박재관*, 김기홍*
*부산대학교 컴퓨터공학과
e-mail : ywpark@pusan.ac.kr

Continuous Query over Business Event Streams in EPCIS Middleware

Yong-Xu Piao*, Bong-Hee Hong*, Jeak-Wan Park*, Gi-Hong Kim*
*Dept. of Computer Science & Engineering, Pusan National University

Abstract

In this paper, the study focus on continuous query in EPC Information Services(EPCIS) middleware which is a component of RFID system. We can consider EPCIS as a data stream system with a repository. In our work continuous query is implemented in two query execution model. One is standing query model another is traditional query execution model in which continuous query run over database periodically. Furthermore a balance strategy is presented. It is used to determine which continuous query implementation model is suitable for the query. Finally we conclude our work and issue some research topic for future work.

1. Introduction

Recently, RFID(Radio Frequency Identification) technique is widely used in many areas. Such as supply chain system, yard management and healthcare applications. RFID system sensed information continuously. A lot of data are generated in RFID system. To cooperate RFID system with other application, many originations processes on RFID standards. EPCglobal, which is a standard association devoted to RFID systems, has presented EPCglobal Architecture Framework[1]. The EPC architecture consists of RFID Reader, ALE, Capturing Application and EPCIS components. Thereinto, ALE[2](Application Level Event) component collecting and filtering RFID stream data. EPCIS[3](EPC Information Services) is responsible for providing EPC-related information services. The goal of EPCglobal is to develop an integrated Middleware supporting EPC-related sharing. As part of this effort, EPCIS provides information service using four kinds of business events and query interfaces which defined in EPCIS specification. EPCIS receive the business events continuously and stored them for future query. Then user can obtain the EPC-related information using query interfaces.

There are two query interfaces in EPCIS specification. Poll interface is One-time query, Subscribe interface is continuous query interfaces. Continuous query implementation is important research issue. Many researchers have focused their attention on continuous query technique study. Most researchers has been devoted to development query indexing for data stream system; rather less attention has been paid to continuous query implementation on data stream with a repository. Little work discusses continuous query over append-only database. For continuous query Two important properties are mentioned in Kun-Lung Wu et al[4] . Low storage cost and excellent search performance is critical in continuous query processing. To provide information services in RFID system efficiently. EPCIS have to reduce response time and main memory requirement.

This paper states continuous query processing in EPICS. There are two works in continuous query processing. First, continuous query is implemented in two query execution model. One is standing query model another is traditional query execution model in which continuous query run over database periodically. The second is that a balance strategy is presented. It is used to determine which continuous query implementation model is suitable for the query.

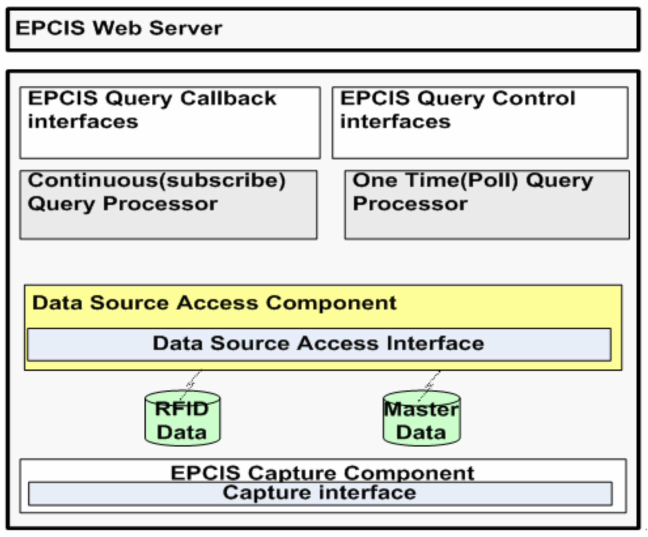
The remainder of this paper is organized as follows: section 2 describes the EPCIS architecture. In section 3 query processing is discussed. . While, An efficient balance strategy for choosing an appropriate query execution model is proposed in section 4. At last in section 5 conclusion and future work is given.

2. Architecture of EPCIS Middleware

RFID system senses data continuously using RFID readers. So many data are gengerated that cause the overload in user application. To avoid this problem, middleware is necessary to filter the non-meaningful data for reducing data volume. In EPCglobal solution, the architecture is organized as three layers. First layer, ALE is used to collect tag data and filtering duplicate and non-useful data. Second layer, Capturing application receives EPC data from ALE and generated business events. In last layer, EPCIS stored business events and provide query interfaces for user application.

Main roles of EPCIS is storing the Business events and providing information services using standard interfaces. Normal data stream system processes unbounded stream data in real time then the data are dropt by system. In EPCIS Middle, all business events are stored in EPCIS repository.

Our EPCIS implementation is based on EPCIS Specification[3]. It implements all interfaces defined in EPCIS specification and provides repository to store business events. Figure 1 shows details of the EPCIS architecture. It consists of capture component, database access component, one-time query processor component and continuous query processor component. Capture component provide a path between EPCIS repository and lower infrastructure. Capture interface is used to receive RFID related business events. This interface is implemented under http and Message queue binding protocol. One-Time query processor and continuous query processor components implements the query interfaces by which user application can obtain EPC-related information. One-time query can be treated as traditional query. Just access database once then return query results. Continuous query run query periodically. Next section will describe detail of query processing. Data Source Access component provide a set interface to access and operate on database.

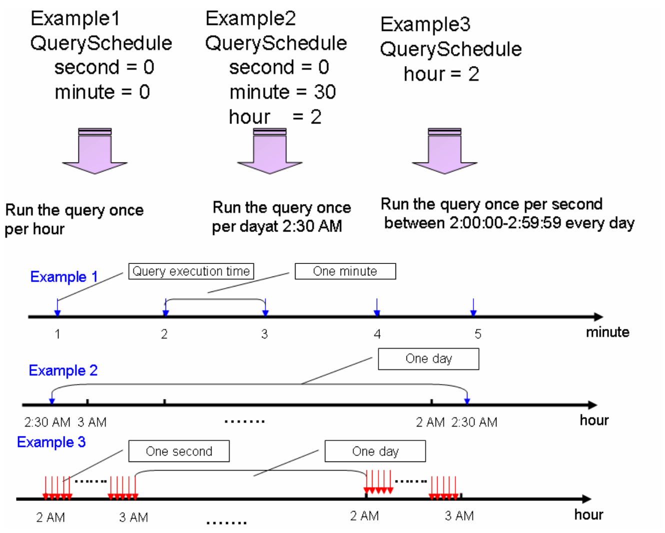

EPCIS Web Server
EPCIS Query Callback interfaces
EPCIS Query Control interfaces
Continuous(subscribe) Query Processor
One Time(Poll) Query Processor
Data Source Access Component
Data Source Access Interface
RFID Data
Master Data
EPCIS Capture Component
Capture interface

(Figure 1) EPCIS Architecture

3. Continuous Query Processing

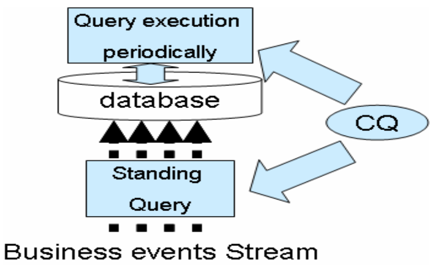
In this section, we describe query processing technique. Two query models have to be implementation. One is one-time query model in which query access Database once and return query result. Another is continuous query model in which query issue once then query result will be delivered continuously. In previous work many continuous query optimization is proposed in data stream system. As mentioned in Section 2. EPCIS is a stream data stream with repository. All the business events are stored in repository. Common stream data system only can process data when the data coming. There no historical data for future processing.

In EPCIS, continuous query is established using subscribe interface defined by EPCglobal. Subscribe interface concludes subscriptionID, query parameter and subscription controls. subscriptionID is identifier of the query. Query parameters, which are a list of name&values, provide the query predicates of query. There are 33 name&values in EPCIS Specification[3] these name&values covers all query conditions for standard business event fields and extension fields. The common continuous query define query execution schedule using time interval. Subscription controls is used to define return query schedule. i.e. when the query should be executed and when query result be delivered to user application. In contrast to common continuous queries, subscription execution schedule use six business time fields define query execution schedule. they are second, minute, hour, dayOfMonth, month and dayOfWeek. Some examples presents in Fig 2. First query schedule example has set second and minute fields to 0. This means run the query once per hour, at the top of the hour. Second example set second field to 0, set minute field to 30 and set hour field to 2. As the subscription schedule showed, the query runs once per day, at 2:30 AM. The third example only set hour field to 2. Third query execute query once per second between 2:00:00 and 2:59:59 each day. By three examples above, we can recognize that duration of twice query execution maybe related short time, maybe related long time and sometimes it is various according to the subscription controls. This requires that some queries execute frequently. Some queries execute non-frequently. Even one query execute in different frequent.



(Figure 2) Examples of continuous query schedule

Since Our EPCIS use a commercial relational database as a repository. A naïve method of implementing a continuous query is to execute the query over the database periodically. We call this method periodical database access model. While using simple implementation, Two Main problems exist. First, because of no elimination in EPCIS, query execution over a high volume data that will cause high response time. Second it is difficult to obtain exact answer when data arrive rapidly. To solve problems above, standing query model[5][6][7] is adopted in most data stream system. When data arrive, query processes the data immediately. If the data is result of the queries which issued periously, data will be hold in main memory until time when query result should be delivered. Database access is not necessary. Using this continuous query implementation can response query on time. Let's consider implementation of EPCIS continuous query in standing query model. For continuous query which have related short duration between twice query execution. It presents good performance. For continuous query which have related long duration between twice query execution. Such as query run once every day. Query result must be hold in memory for a one day. In this case, it also can provide good response time, but it cause high memory cost when a lot of continuous query registered in system. As result, not all continuous queries can get good performance in standing query model. Related long time duration continuous query is implemented as periodical database access model rather than implemented as standing query model. As figure 3 shows, for continuous query, our EPCIS use different implementation model. Some continuous queries use standing query model. Some continuous queries using periodical data access model.



(Figure 3) Continuous Query processing in EPCIS

4. Balance Strategy for Continuous Query Implementation

As discussed in last section, both two continuous query implementations are not appropriate for all EPCIS continuous query. Some continuous queries are suitable to implemented using standing query model. Some continuous queries should be implemented use continuous access database model. How to determine which implementation model is good for a continuous query. We proposed a balance strategy to select an appropriate implementation model. First we should consider which factors can affect the response time and which factors cause high memory cost. Obviously, high response time is caused by query execute over high volume data in database, even efficient index is used. Reason of high memory cost raising is not only query execution duration but also data arrival rate, query selectivity and query number. If there are few queries, low data arrival rate and query with low query selectivity. Through, queries have related long duration between twice query executions. The queries can be implemented in standing query model. Whole system can keep a good performance. We use followed formula present effect of all factor.

$$R * S * T * \alpha N$$

Where R is data arrival rate, N is number of continuous query, S is selectivity of the query and T is duration of twice query execution. α is set between 0 to 1 according to experience. When $R * S * T * \alpha N >$ threshold, the continuous query which have longest duration will be implementation in query execution periodically model. When a continuous query is registered, System use this formula determines the continuous query implementation model.

5. Conclusion and future work

In this paper, we describe the implantation of continuous queries in EPCIS middleware. Continuous query can be implemented in standing query model and periodical query execution models. Standing query is suitable for continuous query which have related short duration between twice query executions. If continuous query with have long duration can run query over database periodically. To determine which query implementation model is appropriate, a balance strategy is proposed. It consider data arrival rate, number of continuous query, selectivity of query and duration of twice continuous query execution. Then there are some research issues for future work. In this work standing query implementation never is mentioned. How query index technique is achieved. In Jae Kwan Park et al[8] introduce a efficient query index for RFID data. But the query index is created only using tag data and reader. How solve the problem of high dimensionality. It is difficult to create query index on high dimensional data. These issues are very interesting for continuous query.

References

[1]EPCglobal. “The EPCglobal Architecture Framework, EPCglobal Standard Specification, 2005.

[2]EPCglobal. “The Application Level Event(ALE) Specification Version 1.0, EPCglobal Standard Specification, 2005.

[3]EPCglobal. “EPC Information Services (EPCIS) Version 1.0.1 Specification, EPCglobal Standard Specification, 2007.

[4]Kun-Lung W. Shyh Kwei C. et al. “Interval Query Indexing for Efficient Stream Processing, ACM CIKM’04.

[5]Douglas T. et al. “Continuous Queries over Append-Only, ACM SIGMOD.

[6]Jianjun C. David J D et al. “NiagaraCQ: A scalable Continuous query stream for Internet Database, ACM SIGMOD.

[7]Chandrasekaran S. et al. “TelegraphCQ: Continuous Dataflow Processing for an Uncertain world, In Proc. CIDR.

[8]Jae-kwan P. Bong Hee H. Chae Hong B. “An Efficient Method for Processing Continuous Queries on RFID Streaming Data, IEEE International Conference on Embedded and Real-Time Computing System and Applications.