Event Processing in Big Data: Creating Fast Data

Big Data is a term that has been widely hyped by the IT media and vendors. Given the huge amounts of data that businesses are accumulating -- terabytes, even petabytes of data -- and the competitive need to get more insight into market trends and corporate performance, that's not surprising. Almost every company is faced with the need to manage, and analyze, "big data". In this article we explore what is *Big Data* and how organizations can take advantage of it using event processing technologies.
Let's start by considering a common use-case in which a sales manager needs to identify the company's top-selling products in each of the primary markets where the product can be purchased for extended periods, such as the last month. It sounds simple, but it's actually a very hard problem to solve. One of the difficulties is the volume of data that needs to be processed for such an analysis. In this case, the volume of data could vary between thousands, to millions, or even billions of sales transactions. Regular tools such as relational databases and visualization software typically don't scale to such volumes of data. Also, the data can be both structured and unstructured, something which is not easily handled by the existing data management and analysis tools.

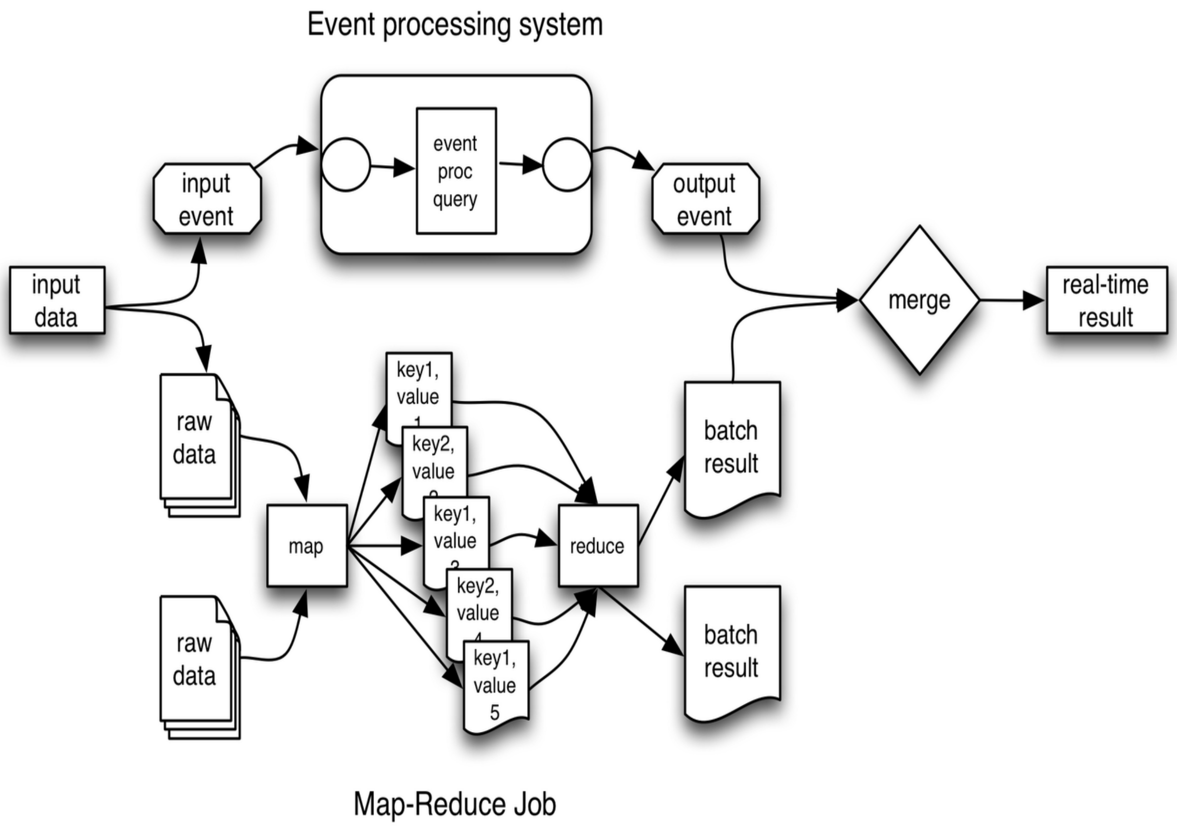
How to Make Big Data Manageable
People have tried managing Big Data in different ways. One common way to deal with large volumes of disparate data is to partition and replicate the data across different servers to create more manageable data sets. However, creating multiple copies of the data makes it more difficult to maintain the consistency of the information, as updates to one data set may not copy over to the others.
That might happen, for instance, if an application that uses the data updates an entry in the sales data but that update fails to be copied to the other replicated data. If this happens, what should be done? It is even worse if other applications that access the data have already used some of that data in other transactions. Another problem with this approach is how should the data be re-partitioned when it changes.

As a result, people came up with a simpler approach, distribute the data (that is, the big data) as protected (immutable) data. For example, store the sales transactions as they happen in some durable storage, and do not let applications change this raw data anymore. Keeping the data immutable avoids the problem of consistency, and the distribution mitigates the problem of volume, however it does mean that a new tool is needed to query this distributed data. As you recall, the original problem is to find the top-selling products in this data.

Fortunately, there is a well-known distributed algorithm that works on top of immutable distributed inputs and that can solve similar queries on distributed systems. . This is the Map-Reduce algorithm, which is implemented in some products including the open-source Apache Hadoop.
Using Apache Hadoop, it's possible to create a Map-Reduce job that uses the big data sales transactions as input and then returns the top-selling products as the result.
However, due to its distributed nature and obviously depending on the size of the input data, the Map-Reduce job returns the top-selling items in batch results every *n* minutes or more likely every *n* hours.
Not every situation can afford to wait several minutes or hours for an answer. For instance, fraud detection needs to be done immediately, as transactions are underway, not after thousands of fraudulent transactions have been completed.
Another example is an extension of the original use-case being discussed where a retailer wishes to have up-to-the-minute knowledge of the worst-selling products, for example, so that the store boost sales of products by handing out discount cards to customers as they enter the store. The longer the wait for the data, the less likely it will be still relevant.

Event Processing Adds Real-Time to Big Data
This is where event processing comes to the rescue. As previously seen, big data is managed as immutable data. Another way of looking into immutable data is to consider it as events. This makes sense as events represent actions that happened and therefore also do not change by nature.
Event processing systems can subscribe to this flow of sales transactions. In other words, each new sales transaction is used both as input to the Map-Reduce jobs, as well as an input event to the event processing system. Therefore, as the Map-Reduce job is processing the large amount of data, the event processing system is processing the latest events, and merging the results from the Map-Reduce job together with its results, providing a real-time answer with

up to the minute precision to the big data queries.



The combination of event processing technology together with Map-Reduce allows you to query a huge volume of data in real-time, something which was unprecedented. This is being called as *Fast Data*. In the next edition, we will look into details of how the event processing practitioner can accomplish this.

About the Author:

Alexandre Alves has over twelve years of experience developing software for large companies, such as BEA, IBM, and Oracle. He has worked with network management, CORBA, JEE, Web-Services, OSGi, BPEL, CEP, and middleware technologies in general. He is the co-author of the WS-BPEL 2.0 specification, co-author of BPEL for Java specification, author of the "OSGi in Depth" book, and a steering committee member of the Event Processing Technical Society (EPTS). He holds a MS in Computer Science from SJSU, has several patents on software, and currently works as the architect for Oracle Event Processing.