Dr. K. Mani Chandy, Simon Ramo Professor of Computer Science, California Institute of Technology and
W. Roy Schulte, Vice President and Distinguished Analyst, Gartner Inc.

July17, 2007

How Event Processing Improves Business Decision Making

Introduction

Almost everyone wants to have an adaptive enterprise, a company that runs faster and smarter because it has "situational awareness," can quickly "sense-and-respond" to opportunities and threats, and can "track-and-trace" important items as they go through their life cycles. What do these three management concepts have in common? They are all based on event processing.

Situational awareness implies having an up-to-the-minute understanding of all critical aspects of the environment and your own internal operations. Simply put, it is "knowing what is going on so you can figure out what to do" (Adam 1993). ***Sense-and-respond*** has a similar flavor, and in fact, the term is sometimes used interchangeably with situational awareness. However sense-and-respond emphasizes timely response to specified opportunities and threats whereas situational awareness connotes seeing the big picture, a holistic view of all factors.

Track-and-trace addresses a different aspect of business operations. It focuses on recording the past and present of identified items as they move between physical locations or through logical steps in a business process. For example, logistic systems track-and-trace movements of physical goods in supply chains. Supply chain management systems tell users where a shipment was and where it is now, predict when it will arrive at a destination, and can prove that it was delivered. The idea of track-and-trace can also apply to the movement of insurance claims, customer orders, payments and other types of intangible, information-based items as they go through their life cycles.

All three of these concepts are examples of operational business intelligence (or military intelligence) in action. Each is intended to decrease the time gap between when something is knowable or predictable and when a person, device or organizational unit takes action on it. Situational awareness, sense-and-respond and track-and-trace are aspirations, sought-after modes of operation. ***Event processing*** is how those aspirations can be achieved. It encompasses a specific set of concepts, design pattern, best practices and software tools.

To address the growing importance of this field, Gartner has organized a new conference on Event Processing and Business Activity Monitoring (BAM) to be held from September 19 - 21, 2007 in Orlando Florida (see www.gartner.com/it/page.jsp?id=502259&tab=overview). It is aimed primarily at

architects, software engineers, business analysts and IT managers. This article provides background information for those who will attend the conference and others who want to learn more about the subject.

Understanding Event Processing

Event processing has two separate, although related, aspects:

1. **Using individual events in asynchronous, push-based business applications.** In this application architecture, a value chain is implemented as a staged Event-driven Architecture (EDA) process in which each stage executes immediately when one properly-formed event message of the right type arrives. Staged EDA makes processes run more quickly and coherently, while also being easier to modify as business requirements change. The architecture is explored in a related article, *"What is Event Driven Architecture (EDA) and Why Does it Matter?"*

2. **Using Complex-event Processing (CEP) for operational business intelligence.** The second major benefit of event processing derives from using CEP systems to find patterns in business event data. Such systems deal with multiple events, often of disparate types coming from disparate event sources. CEP systems look at the relationships between the events to "connect the dots" and extract previously-unavailable insights to enable faster and better operational decisions. Such systems may use Business Activity Monitoring (BAM) dashboards and similar end-user information delivery channels. This article focuses on this second aspect of event processing.

Definitions

An ***event*** is the fact of something happening, such as a bank transaction, stock trade, customer order, address change, shipment delivery or buying a house. Computers can't manipulate events because they are abstractions so an application or device must create a machine-readable report of the event, an ***event object*** that computers can transmit or use in calculations. The fact that Fred Smith withdrew $ 100 from his bank account at 10 AM today is an event. The computer record associated the withdrawal transaction, perhaps in the form of an XML message, is the event object. ***Event processing*** is defined as computing that performs operations on event objects, including creating, reading, transforming or deleting them. An ***event processor*** is any software module that sends, receives or manipulates event objects.

Characteristics of Event Processing

EDA is a style of software architecture that deals with "pushing" event objects. Not all event processing is EDA. Event objects can be manipulated in pull and scheduled modes of operation as well as in push mode. Event objects may be stored in information repositories for later data mining, and event objects may be packaged into remote procedure calls (RPC). Push is necessary for EDA, but there is more to EDA than just

that. This subject is discussed in more detail in related article "*What is Event Driven Architecture (EDA) and Why Does it Matter?*" To summarize, an application uses EDA only if the following conditions are met:

1. **Event objects are pushed.**

   Event objects are sent from an event source to the event consumer in asynchronous messages at times determined by the event source.

2. **Components process events on arrival.**

   An event consumer responds as soon as it gets an event object.

3. **Event objects do not specify operations.**

   An event object does not specify the operation that a consumer of the object must perform upon receiving the object. This decreases the logical coupling between sources and consumers of event objects. The source simply sends a message reporting that an event has happened. The logic to decide what to do about the event is embodied in the consumer (or consumers). A developer can modify or add consumers without changing the event source in any way.

Event processing is not about making traditionally-designed processes run faster, nor about re-engineering processes by eliminating steps, combining steps or doing steps in parallel (although those are all good things to do). Rather, event processing involves the use of specific design principles that differ from those used in most conventional transaction processing and business intelligence (BI) applications.

EDA Design Patterns

EDA is a software architecture that includes several patterns for designing systems that receive, manipulate and transmit event objects. A few of the design patterns are outlined below.

1. Event-at-a-time EDA

A basic EDA application executes a business function each time it receives one event object, whereas a CEP application operates on multiple event objects. For example, a step in a basic, staged EDA process may route an event object to consumers determined by the contents of the object. A basic EDA application handles each event object largely independently of other event objects. By contrast, a CEP step may correlate many event objects generated at many points in time by many event sources. For example, a CEP step may generate new events depending on analyses of sequences of events from suppliers, customers and manufacturing processes to determine whether service-level agreements are likely to be violated.

Some stages of a process may use one-event-at-a-time logic while other stages use CEP techniques. More CEP applications are being developed as threats and opportunities are identified by increasingly complex patterns of activity over time by multiple disparate elements. A CEP application may merely save an event object until other event objects arrive so that operations can be carried out on collections of objects.

2. Event Enrichment

The event enrichment pattern consists of an event processor that uses information in databases and other repositories to transform an event object into one with more accurate or more extensive information. The event enrichment pattern is applied to one event at a time so it can apply to basic EDA steps or used in combination with other EDA patterns. For example, an event object describing the event of a customer email requesting information about a product may be enriched with additional information about the customer.

Event objects generated from data sources outside the enterprise may be noisy. For example, event objects dealing with a competitor's prices may be created by polling the competitor's Web site, extracting textual information, and then extracting prices from paragraphs of English text. Noisy or "dirty" information in event objects may need to be "cleaned" by using prior knowledge of the context in which the objects are generated. For instance, a string "Toytoa" in an event object may be replaced by "Toyota" in applications dealing with cars. Similarly, the pair of numbers – 100,000 and 5,000 – probably deals with odometer readings in miles and price in dollars rather than the other way around.

3. Event Stream Processing (ESP)

An ***event stream*** is a sequence of event objects, typically arranged in order of arrival time. An event stream is often thought of as a special relation or table in a relational database context where the only operations that modify the table are those that append rows. Each row in the table represents one event object. Typically, each row includes a timestamp specifying when the object was created. A row is appended to a stream table when an object arrives. A regular table, of the type found in a normal database, can be obtained from a stream by specifying a window in time: all rows in the stream table with timestamps in the specified window form the regular bounded table. Relational operations in SQL and its extensions can be performed on these tables. In a sense, the data in an EDA system changes but the query stays persistent. This is an inversion of the arrangement found in typical database applications where the data is persistent and the query is transient. Typically, EDA systems read messages from input queues of event objects resident in external message-oriented middleware (MOM) or within the database itself. Event streams often contain extraneous event objects that are irrelevant to the task at hand. One of the duties of an event processor is to filter out the useless event data, separating the "wheat from the chaff" by using rules that are specific to the business function. Commercial event processing software tools are designed to filter (and

otherwise process) event data quickly because some event streams contain thousands or tens of thousands of messages per second.

4. Detecting aggregate trends in multiple streams

Some event processing applications require responses to be executed when aggregates in multiple time sequences satisfy certain trends. Event processors in these applications may compute aggregates across time series in multiple event streams. For example, automated trading systems on Wall Street keep track of the volume-weighted average price (VWAP) for a stock over a moving window of time usually in one or five minute chunks. Trading systems select the "ticks" (trade report events) for a particular stock, sum the total sales prices and divide by the number of shares traded in each time window on a rolling basis. An event processor in the trading platform may emit a new event containing the VWAP figure. The new event, a complex event, is emitted when a specified pattern of other events is detected; the other events (in this case, the ticks) are called **members** of the complex event or member events. One ***complex event*** summarizes the significance of many member events, each which may be a simple event or another complex event. By contrast, a ***simple event*** is atomic in the sense that no one has any reason to subdivide or look at a finer-grained subset of it. (Any event, no matter how small and inconsequential, could always be separated into yet-smaller events but there may be no reason to do so.) ***Complex-event Processing (CEP)*** is defined as processing that performs operations on complex events, including creating, reading, transforming or abstracting them.

5. Finding intricate patterns in multiple event streams

In the VWAP example above, a complex event is computed from multiple instances of the same event type (stock ticks). CEP systems may also derive complex events from patterns among multiple classes of (simple or complex) member events. For example, an event processor may analyze sequences of VWAP objects and sequences of stock-market advisory newsletters to generate events that signal buy or sell opportunities. Another example of an intricate pattern is the pattern of waves over time at multiple points in the ocean floor that signals a tsunami.

Event processing applications may create event hierarchies by using complex events as inputs to derive even higher-level complex events. Despite the "complex" label, it is usually easier to understand and respond to a high level complex event than it would be to work directly with a large volume of low level, simpler events. For example, if a person says that they bought a house, you quickly grasp what happened, although a "house purchase" is a complex event. It would be harder to discuss the situation if instead they described all of the member events, such as the house tours, phone calls, emails, meetings and bank transactions that took place in the course of buying the house. Similarly, a business person can assimilate the significance of a sudden increase in customer orders more quickly by receiving an alert message (a complex event) that says "today's sales volume is 30 percent above average" rather than getting a detailed listing of the individual transactions (see www.complexevents.com or *David Luckham's book*

"The Power of Events" Addison-Wesley Professional, May 2002, ISBN 0-201-72789-7 for more explanation of event hierarchies and complex events).

There are no hard and fast rules between the different patterns of event processing. Many applications have some steps that use one event at a time and others that are CEP steps. ESP applications use CEP in the sense that Join operations in SQL operate on multiple tables or equivalently on multiple sequences of event objects of (possibly) different classes. Nevertheless, to some fuzzy degree, ESP connotes processing high volumes of fast-arriving event objects in a few event streams, whereas CEP connotes finding more obscure patterns in more-diverse event types across the enterprise. The meanings of the labels are still evolving, and it is best to explore the specific pattern underneath a label.

Patterns of Response

After an event processor has detected a condition of interest, such as a threat or an opportunity, it takes one or more actions. A few common patterns of response are identified below.

1. **Cascaded events**: An event processor may respond to patterns that it detects by emitting events in turn; it generates (complex) event objects that are input to other event processors, thus resulting in a cascade of events. Often, the network of event processors is feed-forward, with increasingly complex events in the event hierarchy being formed. In some cases, the network may have feedback cycles too.
2. **Feeding databases**: An action commonly taken by event processors is to write a new event into an event stream in a database. The event stream may be part of an ESP process or may be analyzed in an offline business intelligence application.
3. **Inserting objects into queues, invoking applications and initiating business processes**: An event processor may create an event object and insert the object into a queue, such as a JMS queue, or it may invoke an application through Web services or some other API.
4. **Updating BAM dashboards**: Many event processing applications update BAM dashboards when values of performance indicators change or when critical events are detected.
5. **Initiating human-centric business processes**: Some event processing systems implement sophisticated alert management facilities that determine the optimum team of responders to deal with a given event. The application determines which people are currently online, alerts teams of people who are online and have the best credentials to respond to the specific event, and establishes collaboration tools such as teleconferencing and calendaring for the teams.

Patterns of Sensing

Event objects generated by sensors may have well-defined schemas, may have unknown and changing schemas or have no schema. RSS feeds of text and images have valuable information but do not provide data in well-defined schemas. Event objects generated by business processes within the enterprise may have clear schemas that allow for immediate

manipulation of the event objects without first attempting to "clean" or "understand" the objects. Stock ticker feeds from Bloomberg or Reuters, for example, provide schematized information with clear semantics. Events at competitors, government agencies, and the public are, however, playing an increasing role in determining critical conditions for the enterprise.

Event objects may be pushed to an event application by components within the enterprise and by some suppliers. In other cases, the application may have to poll external information repositories periodically, identify what changed between successive accesses, and then generate event objects capturing the changes. In summary, some (but not all) of the more common patterns of sensing by event processing systems are as follows.

1. Sensor devices – such as motion detectors, RFID readers and ATM machines – send signals when specified patterns (e.g., motion) are detected.
2. Web sites and devices are polled by the system which then generates events from the polled data.
3. Services, such as stock ticker feeds, send sequences of messages containing information to the system.
4. Services, such as news sources, send sequences of messages containing natural language text, image and video data; this data is analyzed by the system to generate events.
5. Business processes, workflow, database triggers and data log-mining tools generate events when specified patterns of behavior occur.
6. People generate events when they detect critical patterns.

The Growing Impact of Event Processing

Companies are starting to use event processing in business applications much more often than in the past. Event processing is not a new concept – the fundamental principles of event processing have been understood for decades and used in network management (including Telco networks), system management tools and other technical roles. A few, high-value business applications, particularly trading systems in the finance industry, have used CEP for almost twenty years (although not using that label). However, CEP was not widely used at the business application level in mainstream applications in the past. This is changing because technologies for sensing, processing, disseminating, managing and responding to events are becoming more powerful and widespread. A few of these changes are detailed below.

- Some of the design techniques and algorithms for applying rules to high volume event streams were only invented in the last 10 years, and were brought to market in commercial software products even more recently. Basic academic research on sense-and-respond, CEP and ESP at Berkeley, Brown, Caltech, Cambridge, MIT, Stanford and many other universities began in the 1990s,continues today and is transferring technology to startups and larger corporations. Publish/subscribe systems are studied in several universities and the area now has its own conferences. Gartner is now tracking more than thirty vendors that offer CEP and ESP products, most of which are less than four years old.

- Decreasing hardware and network costs have made it practical to use event processing for many aspects of business that would have previously been too expensive to automate. Sensors, including RFID, bar code and GPS-based devices, have dropped in price so mainstream companies can deploy them more widely. Lower costs have also contributed to the rising volume of automated teller machine transactions, credit card authorizations and transactions, customer orders, insurance claims and other traditional event-driven transaction processing systems.

- The spread of the Web and related standards has contributed to the emergence of many new types of event streams including Web-based auctions, Web banking, click-stream analysis of customer service and customer experience management applications, micro payment systems, Web-base news feeds (e.g., RSS and Atom) and on-line gaming systems. These standards simplify integration of event processors with sources of events and responders to events.

In most companies, the "killer application" for event processing will be some form of BAM. End users are increasingly aware of the benefits of business dashboards and other BAM mechanisms for keeping in touch with their key performance indicators (KPIs) in near real time. Most BAM solutions are event-driven (EDA) and qualify as CEP systems because they analyze and abstract multiple business events before displaying the results.

BAM is the real-time component of a holistic BI strategy. Unlike most traditional BI, which runs when an analyst submits a query or on a fixed schedule, BAM usually implements the push-oriented, EDA style of architecture. Traditional BI deals with data at rest in a data warehouse or data mart (such databases may include event objects that have been stored for deferred use, so these systems do event processing but are not event *driven*). Traditional BI generally aims at satisfying the needs of analysts or managers, and is strategic in focus. By contrast, BAM aims primarily at operational people making minute-to-minute, tactical decisions. Traditional BI is used by data mining experts whereas BAM dashboards are for everybody in the enterprise. Traditional forms of BI can be described as finding a needle in a haystack – finding valuable information in stored data – whereas BAM uses EDA coupled with human intelligence to figuratively find a needle in a moving conveyor belt of hay.

Conclusion

People and companies have always wanted situational awareness, sense-and-respond nimbleness and track-and-trace insights, but traditional application systems and most forms of BI do not provide these capabilities. The key to enabling near-real-time operational business intelligence is event processing. Event processing has become practical for a much wider range of applications than in the past because of the growing understanding of best practices in EDA application design, more than a decade of research on event processing, the emergence of off-the-shelf commercial event-processing and BAM tools, the spread of industry standards for communication

(particularly the Web) and the drop in cost of sensor devices, computers and networks. Companies now have access to streams of business events that did not previously exist. Large companies experience literally trillions of ordinary business events every day, and a growing number of these are captured and represented as event objects in a form that can be exploited for their information value. Companies that understand event processing well, and use the appropriate design techniques and tools, have a distinct competitive advantage over those that are unaware of the information value of the events that are in their midst.

At the Gartner Event Processing and BAM conference in September 19-21, 2007, we will explore the best practices and software tools associated with event processing at length. The conference has a special emphasis on case studies. About 13 case study speakers will participate, along with 3 experts from academia and a number of Gartner analysts. We will also be joined by 16 vendors of event processing products who will demonstrate the range of capabilities available on the market today. See www.gartner.com/it/page.jsp?id=502259&tab=overview for more details.