The Synergy of SOA, Event-Driven Architecture (EDA), and Complex Event Processing (CEP)

Gerhard Bayer
Senior Consultant
International Systems Group, Inc.
gbayer@isg-inc.com
http://www.isg-inc.com
Table of Contents

- What drives event processing
- CEP defined
  - Typical business cases
  - Overview of event processing technologies
- The integration architecture – pre-requisite for CEP solutions
  - SOA vs. Event-Driven Architecture (EDA)
- Conceptual event processing architecture
- Requirements for a CEP solution
- Conceptual CEP Architecture
- Example CEP application
- What does CEP “programming” look like?
  - Key characteristics of CEP rules
  - CEP patterns
Defining Events

- Business example of an event:
  - Adding a new customer in an insurance company is a business event that a number of applications are interested in

- Technical definition of an event:
  - An event is an action that is usually initiated outside the scope of a program and that is handled by a piece of code inside the program
  - Typical sources of events include the user (who inputs data through a user interface), hardware devices (e.g. a timer), or other programs (event publishers)
  - Physically, the event contains data
  - It may include
    - Type of event, time stamp, source identification, etc.
The Growing Event Stream

- Companies are faced with a growing number of business events
  - Radio Frequency Identification (RFID)
  - B2B partners
  - Batch processes have been replaced by near real-time processes
  - Customer interactions have become personalized and require instantaneous and more complex processing
Business Velocity Is Increasing

Source: Gartner, Inc. June 2008
What Drives Event Processing

- Companies have been constantly reducing their response time to events that affect their business
  - Customers expect immediate action when they have a service inquiry
  - The turn around time for orders is being minimized
  - Fluctuations in supply and demand often mandate instantaneous changes to business processes
- Companies move towards (near) real-time business operations
  - *Real Time Enterprise (RTE)*
    - Some times also referred to as the Zero Latency Enterprise (ZLE)
The Pursuit of the Real-Time Enterprise (RTE)

- The goal of the RTE is to significantly reduce the elapsed time between the moment a business event occurs and the completion of the activities that the business performs in order to respond to the event.
  - This requires to reduce the dependency on manual processes and replace them as much as feasible with automated processing.
  - Many businesses did not have this requirement a few years ago, but today’s competitive pressure is driving them more and more towards the RTE.
From Simple To Complex Event Processing

- Event processing is becoming increasingly complex:
  - **Real-time event notification** –
    - A business event occurs and those individuals or systems who are interested in that event are notified, and potentially act on the event.
  - **Event stream processing** –
    - Many instances of an event occur, such as a stock trade, and a process filters the event stream and notifies individuals or systems only about the occurrences of interest, such as a stock price reaching a certain level.
  - **Complex event processing** –
    - Different types of events, from unrelated transactions, correlated together to identify opportunities, trends, anomalies or threats.
A CEP Definition

- Complex event processing aggregates information from distributed systems in *real time* and applies rules to discern patterns and trends that would otherwise go unnoticed.
  - This gives companies the ability to identify and anticipate opportunities represented by seemingly unrelated events.
  - CEP gives businesses insight into which events will have the greatest operational impact so they can focus their resources to seize opportunities and mitigate risks.
Real-time Event Correlation

- Real-time event correlation is becoming more important in many industries
- Examples include:
  - Trading systems
  - Compliance tracking
  - Fraud detection
  - Airline scheduling systems
  - Supply chain management
CEP Business Cases

- Program traders need the flexibility to execute various strategies to capitalize on opportunities
  - For example, if the stock price of X moves above 2% of the volume weighted average price (VWAP) and stock price of Y moves below 2% of VWAP, then sell 1000 shares of X and buy 5000 shares of Y.
  - The ability to capture real-time events and present those events in context gives traders a better understanding of market changes and the ability to respond quickly to those changes.
CEP Business Cases

- The delayed departure of a flight triggers recalculation of the estimated arrival time.
- This is a new event that is input to other event processing rules:
  - Schedule baggage handling
  - Gate reservation
  - Crew scheduling
- This is fairly simple for one flight, but it gets complex for:
  - An entire airline
  - An entire airport
Typical Applications of CEP

Business Problems and Strategies for Event Processing

- Other: 8%
- Predictive Behavior: 26%
- Security: 20%
- Remote device communication: 12%
- Management by exception: 34%
- Real-time business activity monitoring: 62%
- Risk management: 29%
- Operational efficiencies: 60%
- Logistics Management: 60%
- Supply Chain Management: 32%
- Compliance: 23%
- Algorithmic trading: 9%

Source: eBizQ
Event Processing Technologies

- Technologies that can turn a stream of events into valuable – and actionable – information for business users:
  - **The integration foundation**
    - Service Oriented Architecture (SOA)
    - Event Driven Architecture (EDA)
  - **Business Process Management (BPM)**
    - Automate the actions that the event processing produces as a result
  - **Business Activity Monitoring (BAM)**
    - Provide insight into the performance of the business
    - Complements CEP and BPM
  - **Complex Event Processing (CEP)**
    - Allows applying business rules to streams of real time events, including temporal correlations
One pre-requisite before implementing an event processing architecture is (near) real-time integration of business applications.

- Many companies have already integrated a number of their key applications.
- However, integration solutions are still evolving:
  - Replacing point-to-point integration with a systematic approach.
  - Often applications have been integrated using a batch or file transfer based mechanism.
  - Adherence to standards and a service oriented approach to integration are becoming best practice:
    - XML and Web Services standards.
Pre-requisites For Event Processing Solutions

- CEP products often provide connectivity to custom and packaged applications through adapters
  - It is important to evaluate a CEP solution in the context of the integration architecture guidelines that are (hopefully) in place within IT
    - Avoid creating islands of integration
  - Integration middleware may have already been deployed
    - The Enterprise Service Bus (ESB) has become a technology of choice for many IT organizations
The CEP layer receives business events from applications that have been integrated, for example, through an ESB.

- The integration layer creates “event visibility” for the CEP layer.

**Diagram:**
- CEP Layer: Step 1, 4, 5, 6, 11
- Application Integration Layer:
  - Steps 1-4
  - Step 5
  - Steps 6-9
  - Steps 10-11
Pre-requisites For Event Processing Solutions

- A mix of Service Oriented Architecture (SOA) and Event-Driven Architecture (EDA) should be put in place first
  - They can form the foundation of an Event Processing architecture
    - They make business events available to BPM, BAM, and CEP tools in a standardized way
  - New services are designed specifically for SOA or EDA
  - Legacy systems and packaged applications are integrated using service oriented wrappers
SOA Characteristics

- **Loose coupling** is a key concept in SOA
- It pertains to two levels
  - **Services level**
    - Services and the clients that utilize them should be as independent from each other as possible
  - **Architecture level**
    - Services do not communicate directly, but through an intermediary
    - Allows to implement non business logic oriented functionality outside of the services
      - E.g. security, routing of service requests to different services, version control, logging, data transformation, etc.
- The desired result is a high degree of **flexibility**
  - Enabling rapid composition of new business processes or changes to exiting ones
Architectural Decoupling

Clients invoke services directly

Service Consumer \rightarrow\text{Service Registry}

Service Provider

Legacy Systems, Packages, New Services

Wrapper

WSDL

Architectural decoupling

Service Consumer \rightarrow\text{Service Registry}

Transformation
Routing
Security
Logging etc.

Service Provider

Legacy Systems, Packages, New Services

Wrapper

WSDL
SOA is usually associated with a *client/server model* as opposed to an *event-driven model*.

- In the typical client/server model, a service consumer (client) invokes a service synchronously.
  - It is then typically blocked, waiting for the result to come back from the service.

- Often business scenarios require business events to be propagated across a number of applications (or services).
  - *Event Driven Architecture (EDA)*
  - Services typically use asynchronous communication in form of the publish/subscribe model.
Client/Server SOA vs. EDA

**Client/Server SOA**
- Service Consumer
- Request
- Reply
- Block, synchronous
- Wrapper
- Legacy Systems, Packages, New Services
- Service Provider

**EDA**
- Service Consumer
- Receive asynchronously
- Variable latency
- Publish, no wait, no reply
- Wrapper
- Legacy Systems, Packages, New Services
- Service Provider
The Integration Foundation For Events

- A comprehensive SOA/EDA will enable a layer of high-value services that have a visible impact on the bottom line of the business
  - Business Process Management (BPM)
    - Automated process flow that composes a new, easy to change business process based on the invocation of a number of business services
    - Typically requires support for both request/reply and asynchronous invocation patterns
  - Business Activity Monitoring (BAM)
    - Standardized service interfaces and service access protocols facilitate collection of information about business operations that can be utilized for real-time business intelligence applications
    - Needs to include many diverse business applications
Conceptual Event Processing Architecture

BAM

CEP

Service Oriented Architecture

Event Driven Architecture

Service

Service

Service

Legacy System

Packaged Application
Conceptual Event Processing Architecture

- A typical Event Processing architecture consists of the following elements:
  - SOA and EDA form the foundation
  - BPM is conceptually part of as well as outside of the SOA/EDA foundation
    - Some of the BPM logic is comprised of orchestration of services, which is within the realm of SOA
      - Utilizing standards like BPEL
    - Some of the BPM logic could be implemented using a proprietary tool
    - The inclusion of people performing human process steps has elements both inside and outside of SOA/EDA
      - BPEL4People and WS-HumanTask are inside
      - Worklist clients are applications external to SOA
A typical Event Processing architecture consists of the following elements (cont.):

- **BAM is an application external to SOA/EDA**
  - Events typically emerge from the SOA/EDA and feed into the BAM application
    - The BAM application can query services (i.e. using the pull mode of SOA) since BAM is not a very low latency application
    - But: this could cause processing overhead
  - BAM suites can also include their own, proprietary event source adapters, which are outside of SOA/EDA
  - The BAM engine processes the events and *may* generate new events
    - The results are typically visualized in a dashboard for a human
    - Generated events can be fed into a BPM application
A typical Event Processing architecture consists of the following elements (cont.):

- CEP is an application external to SOA/EDA
  - Events emerge from the EDA and feed into the CEP application
    - The CEP application would not query services (i.e. using the pull mode of SOA) since CEP is a very low latency application
  - CEP suites can also include their own, proprietary event source adapters, which are outside of SOA/EDA
  - The CEP engine processes the events and generates new events
    - They can be visualized in a dashboard or analysis tool for a human
    - They can be fed into a BPM application
    - They can be fed into a BAM application
How To Manage Growing Event Rates?

- The integration architecture creates visibility and access to business events, but …
- … many companies struggle to capitalize on the growing amounts of internal and external streaming data
  - External market data feeds and internal transaction feeds can generate tens of thousands of messages per second
- Processing these events gets increasingly more sophisticated
  - Increasing number and complexity of regulatory and market-specific business rules
Potential Solutions

- Database systems
  - Cannot keep up with such high levels of data rates and complex processing
- Messaging systems
  - Provide virtually no support for sophisticated application development
- Custom development
  - Too costly, takes too long
- Industry specific black box systems
  - Insufficient customization capability

\(\Rightarrow\) CEP tools can be a solution
Requirements For a CEP Solution

- **Real time**
  - The results of processing need to be produced with virtually zero latency
  - Requires in-memory processing, prohibitive to write data to disk

- **Stream processing**
  - Sophisticated time-windowed processing operations are needed
    - For example, windowed join operations
      - E.g. correlating a portfolio stock average with a sector average in the same time window
Requirements For a CEP Solution

- Combining real time and historical data
  - Real time data arrives in streams, while historical data is retrieved from data bases
  - May not be feasible if performance penalty cannot be tolerated

- Handling stream imperfections
  - Frequently, inherent variations, losses, or reordering of the data streams cause data to arrive in the wrong order, or with variable delays
  - In addition, data may sometimes be lost due to glitches in the data feeds or network congestion
Requirements For a CEP Solution

- **Scalability**
  - In many industries the event rates have been dramatically increasing
    - E.g. the Options Price Reporting Authority (OPRA), which aggregates all the quotes and trades from the options exchanges, has peak rates of about 1M messages/sec

- **High availability**
  - CEP solutions often require high availability due to a number of factors:
    - Shortening the response times to critical business events
    - Increased level of automation for handling events
Typical CEP Solution

- Business events enter the CEP engine in form of data streams
- The engine performs a set of transformations on the data
  - Filtering, cleansing, ordering, aggregation
- Pattern matching and business rules are applied to the data
  - The rules are expressed in an event-processing language
Conceptual CEP Architecture

1. Capture simple events (for example, using adapters)
2. Transport events (often using MOM)
3. Apply EPL rules: Filter, correlate, apply constraints, aggregate and update event logs
4. Notify people, trigger BPM processes or invoke applications and SOA services, optionally enriching events with context from event logs and other databases

Source: Gartner
Typical sequence of activities in a CEP architecture:

1. An integration architecture or standalone adapters capture events
2. The events are transported to the CEP engine
   E.g. through messaging or Web based transports
3. The CEP rules could be implemented in a central engine or distributed in Event Processing Agents (EPAs)
4. The rules create aggregated or complex events that are fed into
   ▲ Dashboard
   ▲ BPM or BAM tool
   ▲ Custom application
CEP Vendors

- IBM
- BEA
- TIBCO
- Oracle
- Microsoft
- Progress Software Apama
- Celequest (Cognos)
- Streambase
- Agent Logic
- Aleri merged with Coral8
Example: Apama Dashboard Tool
## 4 Steps To Implement CEP

<table>
<thead>
<tr>
<th>Task</th>
<th>Personnel type</th>
<th>Tool</th>
</tr>
</thead>
<tbody>
<tr>
<td>Connect to business events</td>
<td>IT developer</td>
<td>SOA, EDA, ESB</td>
</tr>
<tr>
<td>Define, correlate, apply business rules to events</td>
<td>Business analyst</td>
<td>Graphical rules definition tool</td>
</tr>
<tr>
<td>Execute the system</td>
<td>Operations</td>
<td>Runtime platform</td>
</tr>
<tr>
<td>Visualize the results</td>
<td>Business user</td>
<td>Dashboard tool</td>
</tr>
</tbody>
</table>
Example CEP Application

- For a given stock portfolio, compare the Volume Weighted Average Price (VWAP) of all trades executed in the market against the VWAP of trades that we make
  - When the VWAP of our trades drop between a pre-defined threshold generate an alert
  - This calculation is done based on a one hour *sliding time window*
Example CEP Application

My_Buys and Data_Feed represent the two data streams that enter the application. This assumes that some kind of data source adapter or EDA infrastructure has been put in place.

A filter is applied to the Data_Feed input to select only those stocks that are part of our portfolio.

The Aggregate operators apply aggregating functions to sliding windows of incoming messages for each stream. Here we are using averaging functions across a one hour window.

The resulting data streams are passed into a join operator that combines aggregated data for the same stock from each input stream.

A final filter is placed on the result of the join. It tests for instances where our trades are over the market VWAP and creates alerts that go into the output stream.
Several ways to define the rules that govern event processing:

- Graphical rule and process flow definition
- Proprietary high level programming languages
  - StreamBase: “Stream SQL”
  - Aleri SQL, Aleri ML (XML based)
  - Progress Software: “Event Programming Language”
    - Also available in Java
Example: Apama Modeling Tool
Example: IBM Rules Development Tool

Correlate Web and Call Center Activity

In response to Pulls Up Service Guide from Web Site When

Where Multiple Product Inquiries in Last 24 Hours

Always Add Lead on SFA System

Multiple Product Inquiries in Last 24 Hours Checks if

Occurrences Of Product Inquiry Within 1 day Is Greater Than 3
Key Characteristics Of CEP Rules

- **Forward inferencing**:  
  - Rules are evaluated if the state or relationship of an object is created or changed  
    - This is important for scalability in a large-scale environment  
    - The rule only evaluates new input, not the entire rule network, every time something changes  
  - This allows the rules to be designed independently of each other  
    - Rules are standalone code - they don’t know about each other
Key Characteristics Of CEP Rules

- Forward inferencing example:

  **Rule 1:**
  If cust.airmiles>1000000 then cust.status="platinum"

  **Rule 2:**
  If cust.status="platinum"
  then createevent(send_upgrade_voucher)

- Rule 2 fires automatically if Rule 1 changes the status of a customer to platinum
  - Since one of the states referenced by Rule 2 has changed
- Rule 2 will not be fired if the customer status has already been “platinum” before
Key Characteristics Of CEP Rules

- **Temporal reasoning:**
  - Rules are evaluated on the presence of events and the object states as a function of time
  - Rule evaluation over a sliding window is a typical scenario in which temporal reasoning is required

- **Spatial reasoning:**
  - Rules are evaluated on the presence of events and the object states as a function of location.
  - Rule evaluation over geometry to infer proximity (near-by, north-of, south-of) is a typical scenario in which spatial reasoning is required
The rule subscribes to a input stream, evaluates a specified logical condition based on event attributes, and, if the condition is true, publishes the event to a destination stream.

For example, an application monitoring a stream of purchase orders may filter out all orders where the condition is

- Priority != ‘High’ and Amount < 100000
CEP Patterns - Time-Based Correlation

- Time-based correlation are event patterns that involve a number of relationships among events spread over time:
  - A followed by B
    - Event B occurs after event A
  - A and B
    - Both events A and B occur, in either order
  - A or B
    - Either A or B (or both) occur
  - Not A
    - Event A does not occur
CEP Patterns - Joining Multiple Event Streams

- Events arriving in input stream 1 are stored in memory
- Events arriving in input stream 2 are joined with events stored from stream 1, and the matching pairs are published as output events
CEP Patterns - Computing Statistical Metrics

- In this pattern the system keeps events in memory and uses the stored values to compute various statistics
  - A typical example would involve computing a running average over a sliding time window
  - This pattern could be applied across multiple event streams
Conclusions

- Companies are moving towards the Real-Time Enterprise
  - Business velocity is increasing
  - Event rates are overwhelming traditional event management approaches
- CEP is gaining acceptance
  - One of the fastest growing software segments
- A comprehensive integration architecture is an ideal foundation for introducing CEP
  - Should be based on SOA and EDA
  - Initial projects can utilize connectivity solutions provided by CEP tools vendors, but be careful not to create CEP silos