Event-Driven Application Servers

Alexandre Vasseur
Thomas Bernhardt
Project Lead and Founder, Esper

TS-1911
Event-Driven Application Servers
Middleware for event processing

Learn the concepts of Event Stream and Complex Event Processing (ESP/CEP).
Understand the role and rationale for Event-Driven Application Servers.
Agenda

On Events and Event Processing
Esper
Demo 1
Event-Driven Application Server
Demo 2
Summary and Q&A
Agenda

On Events and Event Processing

Esper
Demo 1
Event-Driven Application Server
Demo 2
Summary and Q&A
Real-Time Businesses

• Gain an information advantage
  • Anticipate customer needs
  • Create opportunities
  • Beat the competition!

• Trends
  • More information available electronically
  • More fine-grained data
  • Frequent data changes
  • Timely responses are more valuable
What Is an Event?

- Events observe a change in state
  - A stock tick
  - The receipt of a credit check request
  - A password change
  - A sensor measuring temperature every 10 msec
  - A service response time for the last request

- Represented in the system
  - XML
  - Plain Old Java™ Object
  - Key-value pairs
Event Example: The RFID Domain

RFID: Radio Frequency Identification

- LocationReport
  - **asset id**—a unique identifier of the tagged asset
  - **x**—the x location value
  - **y**—the y location value
  - **zone**—derived from x and y

- Use cases
  - “When a given group of assets are not moving together from zone to zone, then...”
  - “When a given asset stays too long in the same zone, then...”
Event-Driven Architecture

Definition and principles of an EDA

- Event Driven Architecture (EDA) is a software architecture supporting the production, **detection**, **consumption** of, and **reaction** to events

- Key principles
  - Loosely-coupled
  - Message-based
  - Location-independent/routed
  - Transport abstracted

Source: http://www.wikipedia.org
Event-Driven Architecture

EDA complements SOA

• Service at the core: expose and handle
  • SOA enables events to flow across systems, transports, and actors

• Event at the core: when…then…
  • EDA is even more loosely coupled and targets sense and respond applications, possibly with extreme transaction processing requirements

“Your nerves
Your five senses
Your brain

“When Hot and kitchen
Then remove my hand!”
Event Processing Solutions

Technical requirements

- Events can be related
  - Time, ordering, and causality (happened before) are first class citizens

- Events can be streamed
  - High throughput, high availability

- Events can change frequently
  - Low latency

- When…then…
  - Expressiveness
  - In and out adapters
Terminology

- **ESP**—Event Stream Processing
  - Monitor streams of event data, analyse those events, and act upon opportunities

  Volume weighted average of Google stock over the last (moving) 30min

- **CEP**—Complex Event Processing
  - Detecting patterns among events

  If this Google VWAP increased more than 5% two times followed by Yahoo! VWAP decreased more than 10% then...

- When **ESP/CEP statement** then...

Source: http://www.eventstreamprocessing.com
Agenda

On Events and Event Processing

Esper
Demo 1
Event-Driven Application Server
Demo 2
Summary and Q&A
Esper—Overview

http://esper.codehaus.org

- Esper—a Java technology ESP/CEP container
  - When Esper statement then your Java code as usual
  - Lightweight and embeddable into any Java technology process
  - Open source
  - Convergence of ESP and CEP

- Project background
  - 3 years of development, more than five developers
  - Release 1.0 announced in June 2006
  - NEsper for .NET
  - Commercial liability, support, and services available EsperTech Inc.
Esper—Architecture

Queries are registered, data flows through them

- Engine: Unit of isolation (time, threads, streams)
- Statements: Event Processing Language (EPL)
- Listener: A simple Java technology interface

Source: http://www.espertech.com
Esper—Sample

1) The ESP/CEP Statement

If a given set of assets are not moving together from zone to zone, alert

// A statement can produce implicit events
insert into CountZone
select zone, count(*) as cnt
from LocationReport.std:unique('assetId')
where assetId in (1, 2, 3)
group by zone

// Second statement detects a pattern among implicit events
select Part.zone from pattern [every Part=CountZone(cnt in (1, 2)) ->
timer:interval(1 min) and not
CountZone(zone=Part.zone, cnt in (0, 3))]]
Esper—Sample
2) Listener and Engine

```java
import net.esper.client.*;

// Get engine instance and register statement
EPServiceProvider engine =
    EPServiceProviderManager.getDefaultProvider();
EPStatement statement =
    engine.getEPAdministrator().createEQL("...");

// Attach a listener
statement.addListener(new UpdateListener() {
    public void update(EventBean[] newEvents,
                          EventBean[] oldEvents) {
        // Handle complex event

        ...
    }
});
```
Esper—Sample
3) Sending events

```java
import net.esper.client.*;

// Get the same engine instance
EPServiceProvider engine =
    EPServiceProviderManager.getDefaultProvider();
EPRuntime runtimeEngine = engine.getRuntime();

... LocationReport event = new LocationReport(assetId, x, y, zone);
runtimeEngine.sendEvent(event);
...```
Expressiveness of Esper EQL/EPL

LR = LocationReport

• Event filtering
  
  // Filter for location report by location rectangle
  select * from LR(x in [4:10], y in [6:12])

• Sliding windows and aggregation
  
  // Count all assets reporting zone 10 in last 30 sec
  select count(*) from LR(zone=10).win:time(30 sec)

• Grouped windows and output rate limiting
  
  // Get a count per zone of the last 10 minutes per zone
  // every 1 minute
  select zone, count(*) as cnt
  from LR.std:groupby('zone').win:time(10 min)
  output every 1 min
Continuous joins

- Joins and outer joins

  ```sql
  // Fire when any asset enters zone 2 before zone 1
  select Zone2.assetId
  from LR(zone=2).win:time(1 day) Zone2
  left outer join
  LR(zone=1).win:time(1 day) Zone1
  on Zone1.assetId = Zone2.assetId
  where Zone1.assetId is null
  ```
Expressiveness of Esper EQL/EPL

Static data is in this good old database

- Historical or reference data

  ```
  // Alert when we hit the minimum inventory
  // for a given zone
  select zone, count(*)
  from LR.std:unique(assetId) as lr,
  sql:db[select mini from Minimum where zone=${lr.zone}]
  having mini < count(*)
  ```

- Esper offers LRU and expiry-time eviction caches
Esper—Performance

Event highways

• Some background information
  • Commercial ESB: 2 600 msg/s on a 4x2.8 GHz in a simple straight through processing pipeline
  • 1 Gb network ~ 65 536 msg/s (2Ko/msg)

• Esper RFID asset tracking
  • 1 000 groups, 3 000 assets, 20 zones
  • 2 000 statements
  • DualCore 2.16GHz
  • ~ 110 000 LR/s
Other Event Processing Approaches
Some possible and some dead wrong solutions

• Database
  • Requires polling
  • Massive data space wasted
  • Not tailored for temporal logic and causality

• Distributed cache or JINI™ network technology space
  • Listener API but no event-processing language

• Rule engines
  • Not optimized towards temporal streams of data
  • Not continuously evaluated
DEMO 1

Asset Tracking RFID
Agenda

On Events and Event Processing
Esper
Demo 1
**Event-Driven Application Server**
Demo 2
Summary and Q&A
Introducing the EDAS
Event-Driven Application Server

• Extreme Transaction Processing (XTP)
  • 100 000 events/second and more
  • Correlation/matching ratio: 2% or less
  • Combining events with long lived historical data

• A middleware server to deploy, run, and manage event-driven applications
  • Open and standardized
  • Java platform or .Net

• Event repositories and ontologies
The EDAS Platform
Open yet specialized, with tooling for event-driven applications

EDAS Studio and Console

EDB Container

Event-Driven Bean

- Built-in transports and formats
- Transport and formats SDK

Event-Driven Application Server

- OA&M
- Security
- HA
- Real-time

Build-in transports and formats

Transport and formats SDK

Connectors to historical and reference data
EDAS Core Services

...that are hard to do without

- Transport and transformation abstraction
- Location independence
  - Intelligent event routing between EDAS instances
  - Honors subscriptions across EDAS
- Event Journalling/Persistence
- Event Replay
- Finite state machines
  - Modelling complex behaviour and processes
  - Conversational scope
- Timer and scheduling services
EDAS Development Tools

Simplifying development and test

• EDAS Studio
  • Debug
  • Test
  • Visualization
  • Tracing
  • Event ontology and model

• Deployment unit: Event-Driven Bean
Event-Driven Bean: a Deployment Unit
Annotation defined, strongly typed when possible

// Fires when an active tag stayed over 1 minute in zone 1

@EventEngine("RFIDOverstayZone1")
public class RFIDQuietTag {
    @When("every LR(zone=1) ->
            (timer:interval(1 min) and not LR(zone!=1))")
    public void onEvent(LR assetEvent) {
        ...
        // Handle complex event
        ...
    }
}
### The Role of Java Platform, Enterprise Edition (Java EE), Java Business Integration, ESB, OSGi?

**Doing new with old? Some assembly required?**

<table>
<thead>
<tr>
<th>Feature</th>
<th>Esper</th>
<th>xxx+Esper</th>
<th>EDAS</th>
</tr>
</thead>
<tbody>
<tr>
<td>Specialized</td>
<td>+</td>
<td>–</td>
<td>++</td>
</tr>
<tr>
<td>Open platform</td>
<td>+</td>
<td>++</td>
<td>?</td>
</tr>
<tr>
<td>Performance</td>
<td>+</td>
<td>+/-</td>
<td>+(+?)</td>
</tr>
<tr>
<td>Embeddable</td>
<td>++</td>
<td>+</td>
<td>–</td>
</tr>
<tr>
<td>Tools</td>
<td>+/-</td>
<td>+</td>
<td>?</td>
</tr>
<tr>
<td>Monitoring</td>
<td>+/-</td>
<td>+</td>
<td>?</td>
</tr>
<tr>
<td>Off-the-shelf</td>
<td>–</td>
<td>–</td>
<td>++</td>
</tr>
</tbody>
</table>
**Runtime: Esper + Java EE platform**

Can Esper inside an MDB serve as an EDAS?

- **Bright side**
  - Deployment
  - Monitoring
  - Tooling
  - Transport
  - Choice
- **Dark side**
  - Performance
  - Other transports
  - Easy to develop?
  - Easy to test?
Deploy bundled Esper into an OSGi = EDAS?

- **Bright side**
  - Deployment
  - Monitoring
  - Lightweight
  - Performance
  - Choice

- **Dark side**
  - Transport?
  - Tooling/APIs
  - Easy to develop?
  - Easy to test?
DEMO 2
Esper as an Event-Driven Application Server
Future Direction

• Service Level Agreement
  • Resource allocation and management
  • Latency requirements across grids and HA clusters

• Event visualization

• Standardization
  • CEP/ESP languages
  • EDAS deployment-unit
  • Event-processing reference architecture working group (EPRAWG)
  • The Object Management Group: EDA RFI
Summary

• CEP/ESP
  • Data streamed against registered queries and listeners
  • Time and causality
• Assemble EDA using Esper today
  • EPL: SQL-like and easy-to-learn
  • Embeddable
• EDAS
  • Help solve difficult EDA development challenges
  • Emerging technology category
For More Information

• Esper at Codehaus http://esper.codehaus.org
  NEsper for .NET

• EsperTech Inc. http://www.espertech.com
  • Contact for support, services, and training

• Book—David Luckham: The Power of Events

• CEP portal—http://www.complexevents.com

• Forum—CEP-Interest@yahoogroups.com
Q&A
Alexandre Vasseur
Thomas Bernhardt
Event-Driven Application Servers

Alexandre Vasseur
Thomas Bernhardt

Project Lead and Founder, Esper

TS-1911