CEP – Complex Event Processing

Roland Stühmer, Nenad Stojanovic
FZI Forschungszentrum Informatik, Karlsruhe

COIN, PlanetData Winter School 2011
Ljubljana
Attribution of Slides

Thanks to:

- Pedro Bizarro, Universidade de Coimbra
- Opher Etzion, IBM
- K. Mani Chandy, Caltech
Why this tutorial?

- Real-time has become one of the crucial characteristics of modern applications and is completely changing the game in the data processing
  - Data is on the move
    - Find results immediately or never
      - *one should be informed as soon as her flight has a delay*

- Information searches for the relevant consumers
  - instead of searching for information, it should find us
    - *one should be automatically informed as soon as her flight has a delay*

- Google search vs. Twitter followers
Since when does Real-time exist

- Real-time is essential for everything we are doing, but we are not aware that it will be possible to:
  - Inform me immediately if my luggage is not onboard and we are about to start (and not after landing)
  - Inform me immediately when two of my friends are sitting in the café close to that I am currently sitting
    - Combining different events in the relevant context
  - Inform me immediately after it becomes very likely that there will be jam on my road (but it is not yet)
    - Even predicting the future events
What time is Real-time

  - Beyoncé’s pregnancy announcement during the MTV VMA show resulted in **8,868 tweets per second**.
  - The previous record was during the final of FIFA Women’s World Cup, between Japan and the United States. That resulted in **7,196 tweets** per second.
  - In terms of past record events, Bin Laden’s death drew a significant peak in Tweets Per Second with **5,106 TPS**. Super Bowl 2011 saw **4,064 TPS**, and the previous all-time high was New Years Eve 2010 in Japan, which hit **6,939 TPS** at its peak.

- Financial market
  - Nanoseconds trading

- eHealth: Remote patient monitoring
  - One semantic signal in 5 sec

- Energy: Smart meters
  - One reading in 15 min

- Real-time is the business real-time or **near** real-time
Goal of this tutorial (I)

- This tutorial is about **real-world problems that can be solved by combining** event processing and semantic technologies

- We will mainly talk about higher-level concepts without going into particular implementation details (provided as references, like EP-SPARQL)
Goal of this tutorial (II)

- This tutorial should **create awareness** about the
  - **event-driven nature** of many real-world problems,
  - the importance of having an **event management system**
    for dealing with real-time data processing applications (like you need DBMS for persistency of data) and
  - the role **semantics** can have in this system
Agenda

1. Introduction to Event Processing

2. Application Potential

3. Event Processing Grand Challenge

4. The Role of Semantic technologies
Agenda

INTRODUCTION TO EVENT PROCESSING
Characteristics of event-driven systems (event-view)

- Real-time awareness/processing
- High throughput of events
- Complex relationships between events
- Combining different information/event sources
What is an event

- Event is an occurrence within a particular system or domain; the word event has double meaning: the real-world occurrence as well as its computerized representation

- Represents changes / occurrences

- Data with a timestamp

- Information container

- Triggers actions
CEP - Complex Event Processing

- Event processing is a form of computing that performs operations on events

- CEP is an **enabling technology** that supports on the fly, (business-) real-time processing of huge event streams

- CEP is about a timely (or in head of time) recognition of the situations of interest and corresponding reaction
  - A **complex event pattern** describes a situation of interests
In recent years – architectures, abstractions, and dedicated commercial products emerge to support functionality that was traditionally carried out within regular programming.

For some applications it is an improvement in TCO; for others is breaking the cost-effectiveness barrier.
Timeline

- Discrete Event Simulation
- Active Data Bases
- Middleware / SOA
- Network Development
- CEP
Pattern detection is one of the notable functions of event processing.
What we actually want to react to are – situations

Sometimes the situation is determined by detecting that some pattern occurred in the flowing events.

Toll violation

Sometimes the events can approximate or indicate with some certainty that the situation has occurred

Frustrated customer

TOLL VIOLATOR

FRUSTRATED CUSTOMER
CEP in Decision Making process

Source: Pedro Bizarro, Uni Coimbra, Portugal
Event Driven Architecture

Event driven architecture: asynchronous, decoupled; each component is autonomic.
What have we learned?

- Event processing is a computing paradigm for real-time data processing
  - (Near) Real-time data processing, (near) real-time answers, lean streaming architecture
  - Events per se might not be meaningful, situations are more complex
  - Processing data on-the-fly

- Pattern detection is one of the notable functions of event processing
Agenda

APPLICATION POTENTIAL
Emerging technologies in enterprise computing (Gartner Hype Cycle)

Figure 1. Hype Cycle for Application Architecture, 2009

Source: Gartner (July 2009)
CEP - another (Gartner) view

Source: Gartner (October 2007)
CEP – a market view

Figure 1. CEP Market and Forecast

Source: Gartner (August 2009)
CEP – a view on vendors

Key
- Query-based
- ECA Rule-based
- Inference Rule-based
- State-based

Research tool (italics)
- Research project or not supported
- Left the market or embedded-only
- Left the market (permanently)

CEP Market Players to March 2011

Source: Tibco
Applications: Smart Cities

- Urban environment which proactively satisfy needs of its "prosumers"

- It **senses continuously**
  - Continuously monitoring the whole environment in order to anticipate problems and opportunities, incl. users' needs

- It **responses proactively**
  - Changing running processes before the situation escalates

- It **self-evolves**
  - The discovery and inclusion of new event/data sources and new consumers of events that will be pushed. It must be done dynamically and in an automatic way.

- **Benefits:**
  - Improved emergency management
    - More efficient early warning systems
  - More efficient traffic management
    - Better resolving of ad-hoc situations (e.g. jams)
  - Advanced m-Commerce
    - More offering for LBA
Applications: Smart eHealth

- Many diseases require a very **personalized treatment** based on the **real-time user’s data**
- Many health **situations** require the **reaction** in the real-time in order to prevent some problems
- Sensor data that can be used for the **real-time health monitoring** (directly/indirectly) is everywhere

- Preventive eHealth
  - covers prevention – the monitoring systems can advise a patient what to do in order to avoid certain diseases
  - adverse behaviour detection – the system can detect some unusuality in the patient‘s behaviour and react correspondingly before escalation
Applications: Smart Grid – big picture

**PRODUCTION**
- local energy management and optimization of distributed and centralized generation mix
- massive use of microgeneration connected close to the end user
- harmonized regulatory framework to facilitate the crossborder trade of energy

**DISTRIBUTION**
- reliability and quality of supply adaptable to the user needs, and adequate to the digital era requisite
- optimal and flexible network operation, maintenance and development
- active demand management and provision of added value services for the end user

**MARKT**

**USERS**
CEP - potential

1. The role of the **real-time push of information** has become very crucial for many application areas:
   - eHealth (e.g. real-time patient monitoring)
   - Energy (e.g. real-time energy consumption monitoring)
   - Transportation/Logistics (e.g. real-time traffic monitoring)
   to name but a few

2. Push of Information is **very data-intensive**, e.g.
   - Twitter is generating about 1000 tweets/sec

3. This overload will be **ever "worse"**
   e.g. Real-time Web:
   - growing number of resources on the Web move away from traditional request/response communication
   - real-time Web technologies:
     - Facebook Graph API supports real-time updates as JSON
     - Google supports push-notifications through PubSubHubbub
     - HTML5 WebSockets can push data to browsers
What have we learned?

- Great outlook for CEP
- The vendor market is very turbulent
- Great application potential in all kind of real-time applications
  - Mainly to improve real-time awareness
Agenda

EVENT PROCESSING GRAND CHALLENGE
Event Processing Grand Challenge

Grand Challenge

- Identify a single, though broad challenge that impacts society
- (measures progress of a research community)

**Event Processing Fabric** / Real-time Web:

- A decentralized, global, Internet-like infrastructure, built upon widely-accepted open standards (EP-TS 2011)
Event Processing Fabric

Distributed ownership and reach of the World Wide Web

Community-based, self-curated, constantly updated of Wikipedia

Adaptive nature of complex adaptive systems
Event Processing Fabric

Designed to be the highway of global real-time data, and the enabler of applications for a proactive society.
The Applications – a preview

- Wide-range of applications in scope and complexity
- From detecting incoming earthquakes
- To warnings of schedule changes in daily commutes
Challenges of building the Fabric (1)

- Thousands, millions of different sources
- From across the globe
- Filtering, aggregating, transforming, & detecting patterns
- Using real-time and historical data
Challenges of building the Fabric (2)

- Manage subscriptions and locations of millions of users
- In a secure and anonymous way
- Across different geographic and administrative domains
- Sending alerts in a timely fashion
- Utilizing the most appropriate channels of communication.
Limitations

- May be inappropriate for highly secure applications
  - such as military or homeland security.
- May be unsuitable for high-performance applications
  - such as real-time stock trading.
- As with the Internet, extremely useful, but not the only way to connect components and systems
Event Processing Fabric Usage

- Earthquake, tsunamis, tornadoes:
  - Data: noisy from cheap sensors and cameras
  - Action: Take shelter; put infrastructure in safe mode
- Healthcare for poor and aging
  - Data: noisy, inexpensive
  - Action: Should I go to a clinic now or stay at home?
- Safety and terrorism
  - Data: Obtained by security personnel, e.g., police cars
  - Action: Take shelter; contain danger
Motivation: Web-scale CEP

Motivation

Real-time Web:
- growing number of resources on the Web move away from traditional request/response communication

Real-time Web technologies:
- Facebook Graph API and Twitter API support real-time updates as JSON
- Google supports push-notifications through PubSubHubbub
- W3C has new Web Notification Working Group
- HTML5 WebSockets / Server Sent Events can push data to browsers

Goals

- Process these events on-the-fly
- High throughput of events
  - Distributed CEP
- Elasticity
  - scalability with dynamic autonomous adjustment
- Federation
  - multi-tenancy
  - permissions
Requirements

For the Web:

- Event format
  - Standard, extensible
- Event pattern language
  - Matching the format
- Event marketplace
  - metadata
Requirements for Event Model (1/2)

- An event is something that has happened, or is contemplated as having happened (Etzion & Niblett 2010)
- Events are first-class objects
  - Fundamental information unit which can be stored, queried and merged with other events (Gupta & Jain 2011)
  - Not inferred from changes in value or in class membership, etc
- Time properties
- Type hierarchy
- Inter-event relationships
An event is an occurrence within a particular system or domain (Etzion & Niblett 2010)

In many real-life systems the domain is quite large and cannot be modelled at design time (Gupta & Jain 2011)
  - Requires model which explicitly models known relationships, cf. open world assumption
  - RDF (W3C Recommendation)

Open/extensible system, everybody should be able to produce or consume events
Requirements for RDF Format

- **Ontology reuse:**
- **Time**
  - DOLCE defines Endurant and Occurrent (Gangemi et al. 2002)
  - startTime, endTime
- **Location**
  - NeoGeo based on W3C Geo predicates (Salas et al. 2011)
  - Point, LineString, Polygon, …
RDF Events (1/2)

- **Goal**: no closed system, everybody should be able to use events
- RDF Resource Description Framework
- Linked Data:
  - “a recommended best practice for exposing, sharing, and connecting pieces of data, information, and knowledge on the Semantic Web using URIs and RDF“
  - include links to other URIs improve discovery of related information on the Web.
RDF Events (2/2)

- Linked Data in CEP:
  - Context for events:
    - Additional information e.g., on everything in the Wikipedia
    - Information is structured e.g., for geo data
  - Publishing of events:
    - HTTP URIs for historic events by ID
    - HTTP URIs for event streams by stream ID

- We are collaborating with the LD community to write a position paper on Streaming with LD:
    http://km.aifb.kit.edu/sites/lodstream/
Event Type Hierarchy

- First set of event types was modelled for Use Cases
Resource Description Framework (RDF)

General method for conceptual description or modeling of information

- Making statements about resources (in particular Web resources)
- In the form of subject-predicate-object expressions (triples)
- Extended with provenance (quadruples)
Event Example

e:e1 {
  <http://events.event-processing.org/ids/e1#event>
    a :AvgTempEvent ;
    :startTime "2011-08-24T14:40:59.837"^^xsd:dateTime ;
    :endTime "2011-08-24T14:42:01.011"^^xsd:dateTime ;
    :members <http://events.event-processing.org/ids/e2#event> <http://events.event-
               processing.org/ids/e3#event> ;
    :source <http://sources.event-processing.org/ids/NiceWeatherAggregator#source> ;
    :stream <http://streams.event-processing.org/ids/NiceWeatherStream#stream> ;
  dbpedia:Nice :avgTemp [ rdf:value "25" ; :event <http://events.event-
                        processing.org/ids/e1#event> ] .
}
Event Example, some notes:

- Each event uses **quadruples** (shown in TriG syntax). The graph name (a.k.a context) before the curly braces is used in the storage backend to enable efficient indexing of contiguous triples.

- There is an **event ontology** (taxonomy) from which type :AvgTempEvent is inherited.

- The event format supports **interval-based events** as well as point-based events by using either just :endTime for a point or both, :startTime and :endTime for an interval.

- The event **links to two other events** e2:event, e3:event in different streams which where used as input to create the :AvgTempEvent. These events are depicted below. They could have further input events themselves.

- The event links to a **stream** where current events can be obtained as they happen.

- The namespace **event-processing.org** was chosen as a generic home for this schema.
Requirements for Event Pattern Language

- Fit the RDF data model
- Query real-time AND historic events (Dindar 2011)
- Support typical temporal operators (Allen 1981)
EP-SPARQL

- SPARQL Protocol And RDF Query Language
- SPARQL is extended to query events with temporal operators such as:
  
  SEQ, EQUALS, time windows, …
- EP-SPARQL was first discussed in:
- Extended to support
  - more expressive events
  - rich set of functions from Xpath
  - easy distinction between real-time and historic part
EP-SPARQL Example

CONSTRUCT ?x dc:mbox ?mbox

WHERE

EVENT ?e1 { ?x dc:mbox ?mbox}
SEQ
EVENT ?e2 { ?x dc:mbox ?mbox}
GRAPH ?e3 { ?x dc:mbox ?mbox}
GRAPH ?e4 { ?x dc:mbox ?mbox}

Real-time events with temporal operators

Historic data with standard SPARQL
Agenda

OUTLOOK
Requirements for Distributed CEP

- **Throughput**
- **Multi-tenancy**
  - federated environment, distributed services
  - better utilizing resources
- **Cloud**
  - paradigm for large-scale computing
  - approach of “everything-as-a-service”, XaaS
  - focus on data-intensive (throughput-oriented) computing
  - (Boniface et al. 2010) focus on time guarantees
    - not yet available in production
Requirements for Distributed CEP

- **Cloud Stack**
  - Levels of the stack
    - low level offerings, called infrastructure-as-a-service (IaaS),
    - greatest flexibility for the programmer
    - platform-as-a-service (PaaS)
    - software-as-a-service (SaaS)
  - ETALIS requiring a fast Prolog interpreter: IaaS

- **Elasticity**
  - variations in event flow
  - and query complexity
  - cost model of cloud infrastructure
The Event Marketplace

- Event-driven World: services exchange events asynchronously
- Events are input for services
- Just as Services should be tradable, so should events
- To receive events, the sources of events must be known
- A marketplace like a search engine provides visibility to distributed sources of events
Agenda

THE ROLE OF SEMANTIC TECHNOLOGIES
Beyond CEP, iCEP view

- CEP is a **process**, consisting of:
  - **Discovery and modeling** the situation of interests
  - Efficient (near real-time) **processing of large numbers of events**
  - **Detection and prediction** of the situations of interest
  - **Contextualization** of detected situations in order to find the best possible reaction

- CEP in a nutshell: **responsive** (Sense & Response) system
Existing CEP approaches: missing points

- CEP as a **process**
  - CEP is usually focused only on the complex event detection

- Complex event patterns as **knowledge artifacts**
  - Complex event patterns are treated as yet another data

- Complex Event detection as a **reasoning task**
  - Detection is performed as a “simple” pattern matching

- Events are geographically **distributed and heterogenous**
  - Events are syntactical structures

- **Advanced analytics** are needed to better understand event flow
  - Events are analysed in isolation
Existing CEP approaches: missing points

- **CEP as a process**
  - CEP is usually focused only on the complex event detection process.

- **Complex events as knowledge artifacts**
  - Complex events are treated as yet another data source.

- **Complex event detection as a reasoning task**
  - Detection is performed as a simple pattern matching.

- **Events are geographically distributed**
  - Events are syntactical structures that are heterogeneous.

- **Advanced analytics** are needed to better understand event flow.
  - Events are analysed in isolation.

**Process view**

**Knowledge Modeling and Management**

**Logic-based processing**

**Semantic Modeling**
iCEP approach: Process view on the responsiveness

Complex processing of EVENTS in order to discover selected situations

Responsive System

Visual presentation in order to better understand discovered situation

ORIENT

OBSERVE

PLAN
Efficient modeling of the situations of interest

React on the situation (automatic / manual)

new situation

reaction
iCEP: Modeling phase

- **Quality of modeled situations** of interests (complex event patterns) is of the crucial importance for the quality of a CEP system
  - An analogy to the knowledge-based systems: reasoning mechanism is useless without comprehensive domain knowledge
- Complex event patterns (CEPATs) are **knowledge artefacts**. Domain experts are required to model them
  - Another analogy: Modeling complex event patterns is a knowledge acquisition bottleneck
- Complex event patterns are **changed continually**
  - A knowledge evolution process is required

=> Knowledge and Semantic Modelling for CEP
Knowledge Evolution process for complex event patterns (CEPATs)

**Generation** of CEPAT can be done manually by an expert, automatically by using historical data and data mining and semiautomatically by using on-line statistics about defined patterns.

**Recommendation** support the fine tuning of the CEPAT created in the first phase. The new CEPAT is compared to existing ones in order to find either inconsistencies or improvements.

**Execution** provides various collections of statistics about the CEPAT usage in the CEP engine.

**Evolution** enables changing CEPATs by taking into account the execution statistics, the changes in the environment and the relations between patterns.
Main advantages of having semantic models:
- **Common understanding** regarding a situation of interests
- Enables the **recommendation** of semantically similar patterns
- Supports better **refinement / evolution** of patterns
iCEP: Detection phase

- Event processing with on-the-fly knowledge evaluation and stream reasoning:
  - Classification and filtering
  - Context
  - Intelligent recommendation
  - Predictive analysis

= ETALIS
Existing Styles for EP languages (samples) *

* - if we add simple and mediated event processing the picture is even more diversified
ETALIS Language for Events

- **ETALIS** is an event processing system based on, so called, event-driven backward chaining rules;
- Formal **semantics** to ground well defined behaviour in Event Processing;
- Detection of complex **events, states, and situations** of interest;
- Standard **event operators** supported: sequence, conjunction, negation, disjunction, parallel composition, window operators etc.
- **Sliding windows** and all common **aggregations** supported;
- **Reasoning** over time, space, and contextual knowledge;
- **Actions/reactions/workflows**, triggered by detected events, are all realised in the same formalism;
- **ETALIS** is an **open source** engine that is implemented in Prolog.
Evaluation Test

- Intel Core Quad CPU Q9400 2.66GHz, 8GB of RAM, Vista x64;
- ETALIS on SWI Prolog 5.6.64 and YAP Prolog 5.1.3 vs. Esper 3.3.0

- Test patterns:

\[ d(Id, X, Y, Z): \neg a(Id, X) \text{ SEQ } b(Id, Y) \text{ SEQ } c(Id, Z). \]

\[ d(Id, X, Y): \neg \text{NOT}(c(Id, Z)).[a(Id, X), b(Id, Y)]. \]

![Throughput vs. Stream Size (Sequence)](chart1)

![Throughput (Negation)](chart2)

![Selectivity of negated events](chart3)
Role of Logic in Event Processing

- **Formal declarative semantics** to ground well defined behaviour of event-based systems;
- **Domain knowledge** evaluation during the complex event detection – Knowledge-based CEP;
- **Justifications**: why did an event occur? Why didn’t it occur?
- **Reasoning** about events (over time, space, context, their relations and constraints):
  - Event retraction (revision) and out-of-order events
  - Contradicting complex events/situations;
  - Detection of not yet fulfilled complex patterns (e.g., 80% fulfilled event).
- **Detection** of complex events, states, situations of interest, and further controlling reactive behaviour (actions/reactions) by means of logic;
- **On-the-fly adaptivity**: everything is data (patterns can be as easy added, changed or removed in the same vein as data);
- Pattern **rule management**: consistency checking, minimal set of pattern rules, correctness of pattern rules etc.
iCEP: Visualization phase

- Trade-off in CEP:
  defining more interesting situations to be followed will cover the problem space better, but it will introduce complex event overload

- Visualization of complex events:
  - Their temporal context (complex event flow in time)
  - Their synthesis (how a complex event has emerged based on atomic events)
  - Their context wrt. other complex events

- The goal: better support for the decision making process

- 3D visualization model
  - Different views on the complex event flow, e.g. using rotation
  - Better contextualization (more characteristics can be presented)
  Enables active visualization (reacting from the visualization interface)
iCEP: Visualization framework

- Pattern-level filter
- The diameter corresponds to the value of the complex event
- Time window slider
- Rotation control
- Change viewpoint
- Synthesis out of atomic events

The diameter corresponds to the value of the complex event.
Thank you for your attention!

Questions?

Follow us:

Twitter: @play_fp7
RSS: http://news.play-project.eu/
Web: http://www.play-project.eu/