

Orientation in Objects

Complex Event Processing (CEP) mit Esper

Esper ist ein leichtgewichtiges Open Source Tool, das die Verarbeitung von Ereignisströmen erlaubt. In dieser Session wird anhand des Projekts Esper der Themenkomplex CEP - Complex Event Processing - erläutert. Eine Abgrenzung zu verwandten Technologien, wie ESB und JMS, und ein kleines Beispiel sollen dem Teilnehmer die Technologie und das Tool Esper näher bringen und Einsatzmöglichkeiten verdeutlichen.

Version: 1.1

EDA - Event Driven Architecture

Orientation in Objects

- Grundprinzipien
  - sehr lose Kopplung
  - basiert vollständig auf Nachrichten
  - abstrahierter Nachrichtentransport
  - Unabhängigkeit der Eventbearbeitung von Ort und Weg zwischen den beteiligten Komponenten

Realtime Business

Orientation in Objects

- Wirtschaftliche Potentiale entdecken
  - Marktbedürfnisse münden in Nachfragen
  - Früherkennung schafft Wettbewerbsvorteile
- unterstützende Trends
  - zunehmende Verfügbarkeit elektronischer Daten
  - elektronische Bereitstellung von Detailinformationen
    - **feinkörnige Daten**
    - **häufige Updates**
  - zeitnaher Zugriff Reaktionsschnittstellen möglich

Events

Orientation in Objects

- Beobachtung einer Zustandsänderung
  - Quittierung einer Autorisierungsprüfung
  - Antwortzeit der letzten Anfrage
  - Messwert eines Sensors im Haus
  - Aktienpreisänderung
- benötigen eine technische Repräsentation
  - Schlüssel/Wert- Paare
  - XML
  - POJO

Event Beispiele im Engineering

Orientation in Objects

- KFZ-sensoren
  - Ermittlung von Betriebsgrößen durch verschiedene häufig über ein Bussystem verbundender Sensoren
    - **Drehgeschwindigkeit des Rades**
    - **Beschleunigung in allen drei Raumachsen**
    - **Luftdruck der Reifen**
    - **Aussenlufttemperatur**
    - **Luftfeuchtigkeit**
    - **Sitzbelastung**
    - **Lambda-Wert**
- Anwendungsbeispiele
  - ABS
  - Einspritzsysteme
  - Airbagabschaltung
  - Verhaltensforschung?

EDA Requirements

Orientation in Objects

- Events benötigen Ströme
  - Durchsatz
  - Verfügbarkeit
- Events treten häufig auf
  - geringe Latenz
- Eventbeziehungen
- Unterstützung einer Eventmodellierung
  - ausdrucksstarke Semantik
    - **Grund-Folge Modellierung**
  - Kopplung mit Stromkonzept zur In/Out-Adaptierung

EDA Paradigma

Orientation in Objects

- EDA konzentriert sich auf Anwendungen nach folgendem Muster
  - Events müssen zeitgerecht in Ihrem Eingang erkannt
  - konsumiert
  - und anschließend folgerichtig darauf reagiert werden
- Event getriebene Anwendungen lassen sich in natürlicher Sprache regelartig komprimieren
  - - „wenn .. dann...“
    - **When-clause**
    - **Handler code**

ESP

Orientation in Objects

- ESP(Event Stream Processing)-Statement
  - Überwachung von Eventstromdaten
  - Analyse der Events
  - Reaktion bei bestimmten Gegebenheiten
  - z.B. Kfz-ABS:
    - **Wenn Drehgeschwindigkeit des Rades beim Auto null während Fahrzeugbeschleunigung negativ dann öffne Bremsdruckventil**
- Grund-Folge -Modellierung nutzt als Grund ein „When “-clause
  - wird auch als ESP/CEP-Statement bezeichnet

CEP-Statements

Orientation in Objects

- CEP(Complex Event Processing)-Statement
- Mustererkennung zwischen Events
  - Filtern
  - Aggregieren
  - Korrrelieren
  - z.B. automatische Notrufzentrale
    - **wenn Fahrersitzbelastungsänderung auf null innerhalb 200ms mit negativer Beschleunigung>a und Reifendruckabfall dann automatische Anwahl Notrufzentrale**

EDA vs. SOA

Orientation in Objects

- EDA konzentiert sich auf Anwendungen mit Wahrnehmungsreaktionsmustern
  - oft verbunden mit komplexer Transaktionalität
  - sehr viel „loser“ als typische SOA
- SOA Paradigma - „Dienst anbieten und Nachfragen bearbeiten“
  - SOA ermöglicht Events zwischen Systemen, Protokollen und Rollen auszutauschen

EDA Infrastruktur

Orientation in Objects

- EXtreme Transaction Processing (XTP)
  - Begriff wurde 2003 durch Gartner erstmals genannt
  - > 100 000 Events/s
  - Korrelationsrate < 2%
  - Kombination von Events mit langlebigen Historiendaten
- Event Repositories und Ontologien

Event Driven Applicationservers

Orientation in Objects

- Middleware für Deployment,Laufzeit und Management von EDA-Anwendungen
  - Grundlegende Konzepte
    - **Event Stream Processing (ESP)**
    - **Complex Event Processing (CEP)**
  - offen
  - standardisiert
  - Java/.Net support

EDAS Services

Orientation in Objects

- Abstraktion von Transport und Transformation
- Ortstransparenz
  - Eventrouting zwischen EDAS-instanzen
  - Listener können an beliebigen Instanzen registriert werden
- Event Protokollierung/Speicherung
- Replay-mechanismen
- Finite Zustandsautomaten
  - Modellierung
  - Dialogzustand
- Zeitgeber und Kalender

EDAS Zukunft ?

Orientation in Objects

- „By 2011, a new generation of application platforms stemming from the convergence of diverse XTP-enabling technologies will supersede Java EE and .NET as the platforms of choice for large-scale, business-critical applications (0.8 probability).“
- „By 2010, support for advanced, event-centric programming models will be a standard feature in all application platforms aimed at XTP applications (0.8 probability).“
  - Quelle: Gartner RAS Core Research Note G00146107 Q1/2007

Esper

Orientation in Objects

- Esper ist ein Softwarebaustein für CEP und ESP Anwendungen, der in Java als Esper und .NET als Nesper bereitgestellt wird.
- Esper erlaubt die zeitnahe Entwicklung von Anwendungen mit hohem Aufkommen eingehender Ereignisse bzw. Nachrichten. Dabei können Ereignisse auf verschiedene Arten in Echtzeit analysiert, gefiltert und konsumiert werden.
  - Quelle: Esper Homepage
    - **http://esper.codehaus.org**

Projekt Überblick

Orientation in Objects

- Lizenz:
  - Open Source License: GPL 2.0
  - EsperTech
    - **Kommerzielle Vermarktung**
    - **Q2/2007 BEA Partner für BEA Event Server**
- Parallele .NET Implementierung (Nesper)

OHLOH Projektinformationen

- Summary:
  - Mostly written in Java
  - Small development team
  - Few source code comments

Java	61%
XML	39%
Other	
HTML	<1%
CSS	<1%
DOS batch script	<1%
Perl	<1%
shell script	<1%
SQL	<1%

Filter on: Filter [4 total]

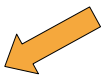
Name	Activity (5 years)	Commits	Person Years	Lines Modified	Comment Ratio
bernhardttom	5 year commit count	306	1.4	395,839	10.6%
srdan	5 year commit count	56	0.2	3,438	12.7%
avasseur	5 year commit count	28	0.4	2,224	12.7%
bwalding	5 year commit count	4	0.0	0	n/a

Projekt Dokumentation

Orientation in Objects

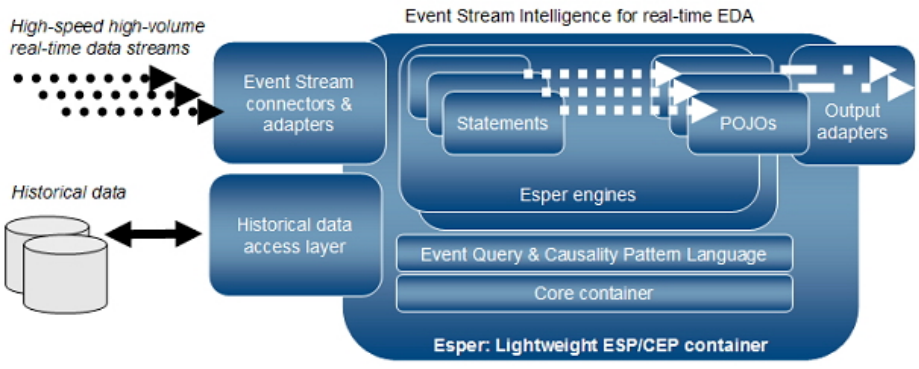
- Umfangreiche Referenzdokumentation
  - PDF Dokument mit 130+ Seiten
- Viele Beispiele
  - Benchmark Beispiel
  - Viele Tests/ Beispielanwendungen mit Dokumentation

History



Version 1.11.0 Update
Released September 15, 2007
Version 1.10.0 Update
Released July 18, 2007
Version 1.9.0 Update
Released June 6, 2007
Version 1.8.0 Update
Released April 22, 2007
Version 1.7.0 Update
Released March 18, 2007
Version 1.6.0 Update
Released February 17, 2007
…
Version 0.7.0 (Alpha Release)
Released January 16, 2006

Big picture



© EsperTech

API Overview

Orientation in Objects

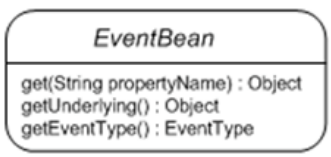
- **EPServiceProvider**
  - Engine Prozessierungseinheit
  - Threads,Zeit, Streams werden hier registriert
- **EPStatement**
  - Statements / Queries
  - in EQL (Event Query Language)
- **UpdateListener**
  - Listener
  - POJI

Events

Orientation in Objects

- Sehr flexibler Ansatz
- Esper verschickt EventBeans
- EventBeans können
  - POJOs
  - Java.util.Map
  - org.w3c.dom.Node

  enthalten



Esper Events - POJOS

Orientation in Objects

```
public class EmployeeEvent {
   public String getFirstName();
   public Address getAddress(String type);
   public Employee getSubordinate(int index);
   public Employee[] getAllSubordinates();
}

every EmployeeEvent(firstName='myName')
every EmployeeEvent(address('home').streetName='Park Avenue')
every EmployeeEvent(subordinate[0].name='anotherName')
every EmployeeEvent(allSubordinates[1].name='thatName')
every
       EmployeeEvent(subordinate[0].address('home')
           .streetName='Water Street')
```


Esper Events - XML

Orientation in Objects

```
<?xml version="1.0" encoding="UTF-8"?>
<Sensor>
   <ID>urn:epc:1:4.16.36<ID>
       <Observation Command="READ_PALLET_TAGS_ONLY">
             <ID>00000001<ID>
             <Tag> <ID>urn:epc:1:2.24.400<ID> </Tag>
             <Tag> <ID>urn:epc:1:2.24.401<ID> </Tag>
       </Observation>
</Sensor>

select ID, Observation.ID, Observation.Command,
   Observation.Tag[0]
   from SensorEvent.win:time(30 sec)
```

Esper Events – java.util.Map

Orientation in Objects

```
Map event = new HashMap();
event.put("txn", txn);
event.put("account", account);
epRuntime.sendEvent(event, "TxnEvent");

select account.id, account.rate * txn.amount
  from TxnEvent.win:time(60 sec)
  group by account.id
```


Esper 1x1

Orientation in Objects

```
String stmtText =
       "insert into ThroughputPerFeed " +
       " select feed, count(*) as cnt " +
       "from " + FeedEvent.class.getName() + ".win:time_batch(1 sec) " +
       "group by feed";

EPServiceProvider engine = EPServiceProviderManager.getDefaultProvider();
EPStatement stmt = engine.getEPAdministrator().createEQL(stmtText);

stmt.addListener(new MyListener());

        while(true)
        {
            FeedEvent event;
            event = new FeedEvent(FeedEnum.FEED_A, "IBM", 70);
            engine.getEPRuntime().sendEvent(event);
            event = new FeedEvent(FeedEnum.FEED_B, "IBM", 70);
            engine.getEPRuntime().sendEvent(event);
        }
```

Processing Model

Orientation in Objects

- Kontinuierliche Bearbeitung
- Listener werden benachrichtigt, wenn sich die Ereignismenge verändert
- Datenbank „inside-out“
  - Nicht Anfragen gegen die Daten ausführen sondern Daten durch die Anfragen schicken.

EQL Kurz & Gut

Orientation in Objects

- Analogie zu SQL
  - Ereignisströme sind Tabellen
  - Ein Ereignis entspring ein Datensatz
  - Eigenschaften im Ereignis entsprechen Datenfelder
- EQL Queries lassen sich grob in
  - ESP Queries und
  - CEP Queries

  einteilen.

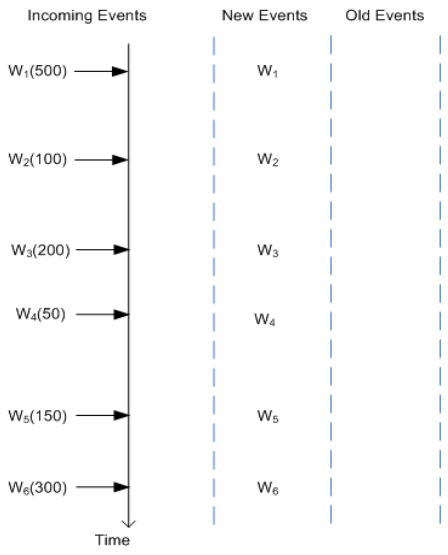
ESP Queries

Orientation in Objects

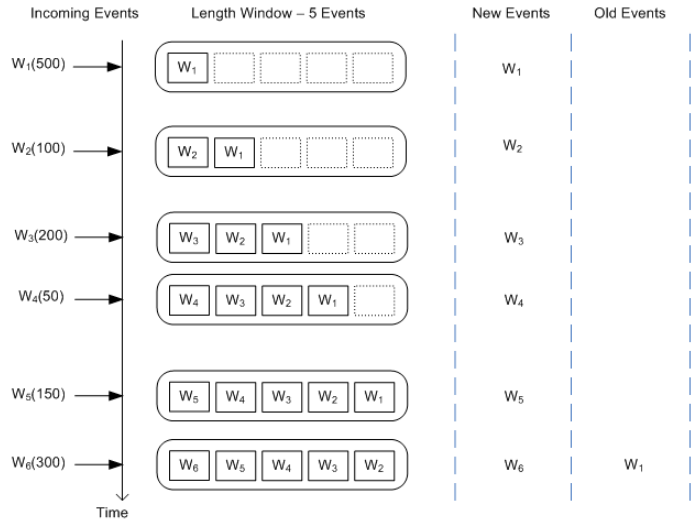
- Abfragen über
  - Einzelne Ereignisse
  - Ereignisse in einem Zeitfenster
  - Ereignisse in einem Mengen-Fenster

select * from Withdrawal

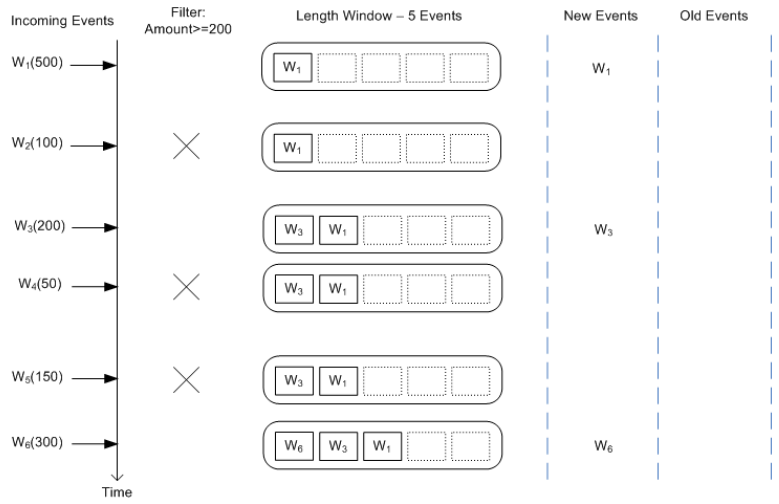
Orientation in Objects



select * from Withdrawal.win:length(5)



select * from Withdrawal(amount>=200).win:length(5)



Insert into…

Orientation in Objects

```
insert into
  WithdrawalFiltered
      select * from Withdrawal
      where Math.ceil(amount) >= 200

select * from WithdrawalFiltered
```

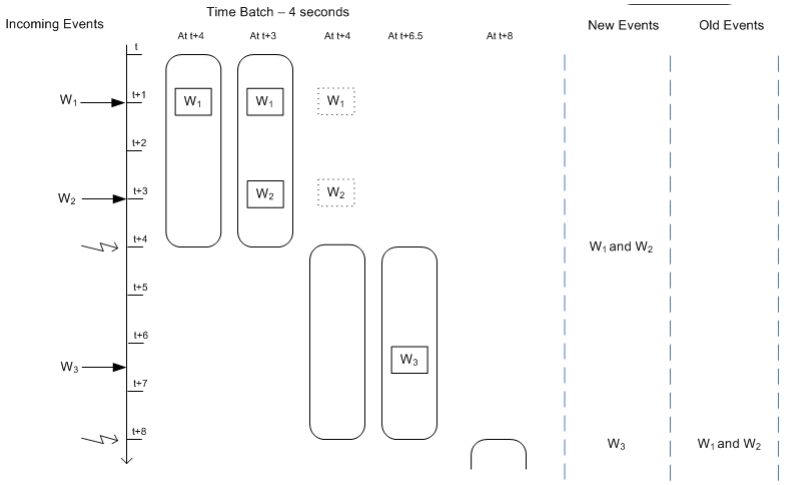

Batch Modus

Orientation in Objects

- Im Batch-Modus werden erst Listener Benachrichtig, wenn eine definierte Menge an Events eingetroffen sind
  - Zeitlich
    - **select * from Withdrawal.win:time_batch(4 sec)**
  - Numerisch
    - **select * from Withdrawal.win:length_batch(5)**

select * from Withdrawal.win:time_batch(4 sec)

Orientation in Objects



CEP Queries

Orientation in Objects

- Abfragen über
  - Muster zwischen Ereignisse/ Ergenissstöme
  - Verwendeter Begriff: „Pattern“

Event Pattern Überblick

Orientation in Objects

Patterns werden über den Konstrukt „`pattern [...]`“ definiert

- Wiederholungsangabe mit „`every`“
- Logische Operatoren
  - **and, or, not**
- Zeitliche Operatoren
  - „`->`“ **(followed-by)**
- Guards are where-conditions that control the lifecycle of subexpressions.
  - **`timer:within`**
- Observers observe time events as well as other events.
  - **`timer:interval`**
  - **`timer:at`**

Pattern Beispiele

Orientation in Objects

- Ereignisse:
  - A1 B1 C1 B2 A2 D1 A3 B3 E1 A4 F1 B4
- `pattern [ every A -> B ]`
  - {A1,B1}, {A2,B3}, {A3,B3}, {A4,B4}
- `pattern [ every ( A -> B ) ]`
  - {A1,B1}, {A2,B3}, {A4,B4}

„Subexpression“

Pattern Beispiele

Orientation in Objects

- Ereignisse:
  - A1 B1 C1 B2 A2 D1 A3 B3 E1 A4 F1 B4
- `pattern [ A -> every B ]`
  - {A1,B1}, {A1,B2}, {A1,B3}, {A1,B4},
- `pattern [ every A -> every B ]`
  - {A1, B1}, {A1, B2},
  - {A1, B3}, {A2, B3}, {A3, B3},
  - {A1, B4}, {A2, B4}, {A3, B4} and {A4, B4}

Guards und Observer

Orientation in Objects

- (A or B) where timer:within (5 sec)
  - Ein A oder B in den nächsten 5 Sekunden
- (every A) where timer:within(10 sec)
  - Alle A Ereignisse in den nächsten 5 Sekunden
- A -> timer:interval(10 seconds)
  - Nach A 10 Sekunden warten
- every timer:at(5, *, *, *, *)
  - Alle 5 Minuten

Temperatur…

Orientation in Objects

- Ein Temperatursensor erzeugt Temperaturereignisse
  - Sample
    - **sensor**
    - **temp**
- Es soll gewarnt werden, wenn innerhalb 90 Sekunden
  - 3 aufeinanderfolgende Ereignisse
  - Temperatur > 50
  - Auf dem selbem Sensor

Complex Event Processing…

Orientation in Objects

```
every sample= Sample(temp > 50)
  -> (
       (
         Sample(sensor=sample.sensor, temp > 50)
          and not
         Sample(sensor=sample.sensor, temp <= 50)
       )
        ->
       (
         Sample(sensor=sample.sensor, temp > 50)
          and not
         Sample(sensor=sample.sensor, temp <= 50)
       )
      ) where timer:within(90 seconds)
```

Beispiel - EPL-Syntax

Orientation in Objects

- Falls drei RFID-Tags nicht gemeinsam von einer Zone in die nächste wandern dann Alarm auslösen
  - EPL-Statements können implizite Events erzeugen

```
insert into CountZone
select zone,count(*) as cnt
from LocationReport.std:unique('assetID')
where assetId in (1,2,3)
group by zone
```


Beispiel - EPL-Syntax

Orientation in Objects

- Falls drei RFID-Tags nicht gemeinsam von einer Zone in die nächste wandern dann Alarm auslösen
  - EPL Statements können Patterns zwischen impliziten Events auflösen

```
select Part.zone from pattern [
every Part=CountZone(cnt in (1,2)) ->
(timer:interval(1min) and not
CountZone(zone=Part.zone, cnt in (0,3)))]
```

EQL Semantik

Orientation in Objects

- Events filtern
  - ausfiltern der Locationreports innerhalb eines Rechtecks

```
select * from LR(x in [4:10], y in []6:12])
```

- Aggregation über Zeitfenster
  - aggregiere alle Ids in zone 10 innerhalb der letzten 30 Sekunden

```
select count(*)from LR(zone=10).win:time(30sec)
```


EPL Semantik II

Orientation in Objects

- Zeitfenster Aggregation mit limitierter Ausgabe
  - eine Zählung pro Minute von jeder Zone der letzten 10 Minuten innerhalb dieser Zone

```
select zone,count(*) as cnt
from LR.std:groupby('zone').win:time(10 min) output
  every 1 min
```

- Verbindung mit SQL-möglich
- kontinuierliche joins

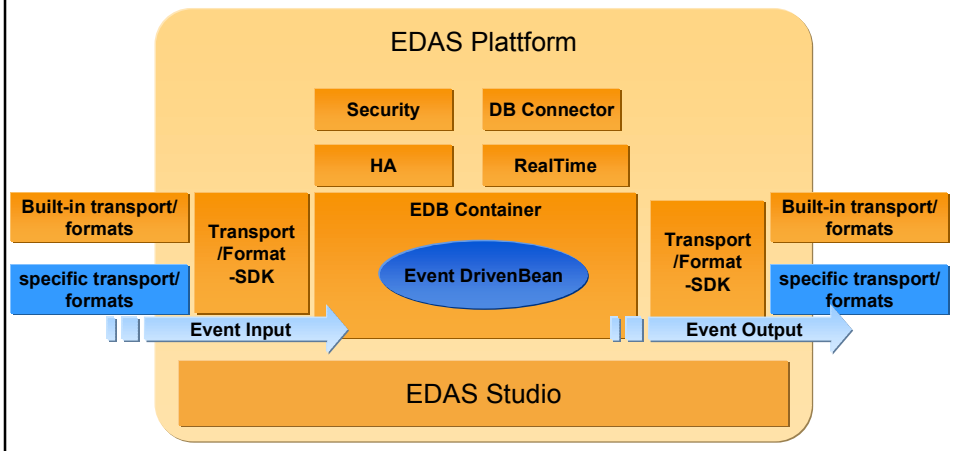
Alternativen zu EDA

Orientation in Objects

- Datenbanken
  - polling
  - Resourcenverschwendung
  - SQL unterstützt keine zeitlichen und kausalen Zusammenhänge
- Rule engines + JMS
  - keine Optimierung für zeitliche Eventflüsse
  - keine kontinuierliche Evaluierung von EPL-Statements
- Distributed Caches bzw. JINI spaces
  - bieten Listener aber kein EPL

Event Driven AppServer

Orientation in Objects



Performance Aspekte

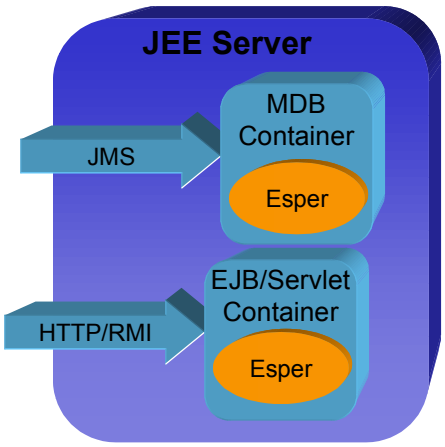
Orientation in Objects

- Esper Benchmark
  - RFID tracking sample
  - 1000 gruppen zu drei Elementen , 20 Zonen
  - 2000 registrierte Statements
  - ~ 110 000 Events/s

Esper in JEE

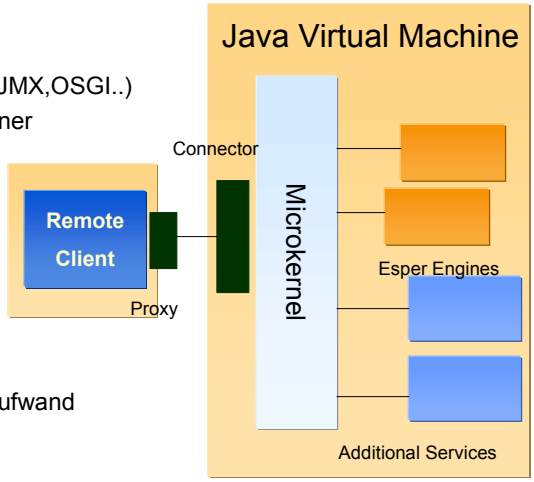
Orientation in Objects

- Vorteile
  - Deployment
  - Monitoring
  - Tooling
  - Transportabstraktion
  - Anbieterunabhängigkeit
- Nachteile
  - Performance
  - Alternativen zu JMS/HTTP
  - Entwicklungs- und Testaufwand



Esper + Microkernel

- Vorteile
  - Anbieterunabhängigkeit(JMX,OSGI..)
  - Leichtgewichtiger Container
  - Deployment
  - Monitoring
  - Performance
- Nachteile
  - Transportabstraktion
  - Tooling
  - Entwicklungs- und Testaufwand
  - Operating



Esper als EDAS Basis

	Esper	Esper-integration	Nischen-Produkt
Open	+	++	?
Komm. Support	–	–	++
Performance	+	+/–	+(+?)
Embeddable	++	+	–
Tools	+/–	+	?
Monitoring	+/–	+	?

Wishlist - EDAS Tooling

Orientation in Objects

- Typical Software Lifecycle
  - Event ontology
  - Event Model
  - Standardisierte Deployment Unit
    - **Event Driven Bean**
  - Debug
  - Tracing
  - Test
  - Visualization
  - Monitoring

Esper Project Roadmap

Orientation in Objects

- Integration mit ESB (Mule, JBossESB, Synapse?)
- HA Clustering
- Grid Lösung
- SCA Integration
- Standardisierung (OMG)
  - CEP/ESP Sprache (EPL)
  - Deployment Unit

Zusammenfassung

Orientation in Objects

- EDA Architekturparadigma für „Real Time Business“- Anwendung
  - Anwendungen mit Wahrnehmungsmustern
  - komplementär zu SOA
- CEP/ESP sind
  - zur Mustererkennung werden Eventdatenströme gefiltert, aggregiert und korreliert
  - besitzen Zeit und Kausalitätsanforderungen an eine Programmiersystem

Zusammenfassung

Orientation in Objects

- Esper
  - ESP/CEP Engine
  - Leichtgewichtig/integrierbar/open Source
  - Eigene EQL
    - **Ausdruckstark**
    - **Zeitfenster und Zeitabhängigkeiten modellierbar**
- EDAS
  - erleichtert EDA Entwicklung ähnlich JEE Containern
    - **schnell wachsender Toolmarkt ?**

Informationsquellen

- Esper at Codehaus http://esper.codehaus.org
- NEsper for .NET
- EsperTech Inc. http://www.espertech.com
  - support, services, and training
- Literatur —David Luckham: The Power of Events
- CEP portal—http://www.complexevents.com
- Forum—CEP-Interest@yahoogroups.com



Vielen Dank für Ihre Aufmerksamkeit !

Version: 1.1

Orientation in Objects

Fragen ?

Version: 1.1

Event Driven Bean

Orientation in Objects

- Annotationsbasiert und Typisiert

```
@EventEngine
public class RFIDQuietTag {
   @When("every LR(zone=1) ->
   (timer:interval(1min) and not LR(zone!=1))")

   public void onEvent(LR assetEvent) {
     //action handler

   }
}
```