

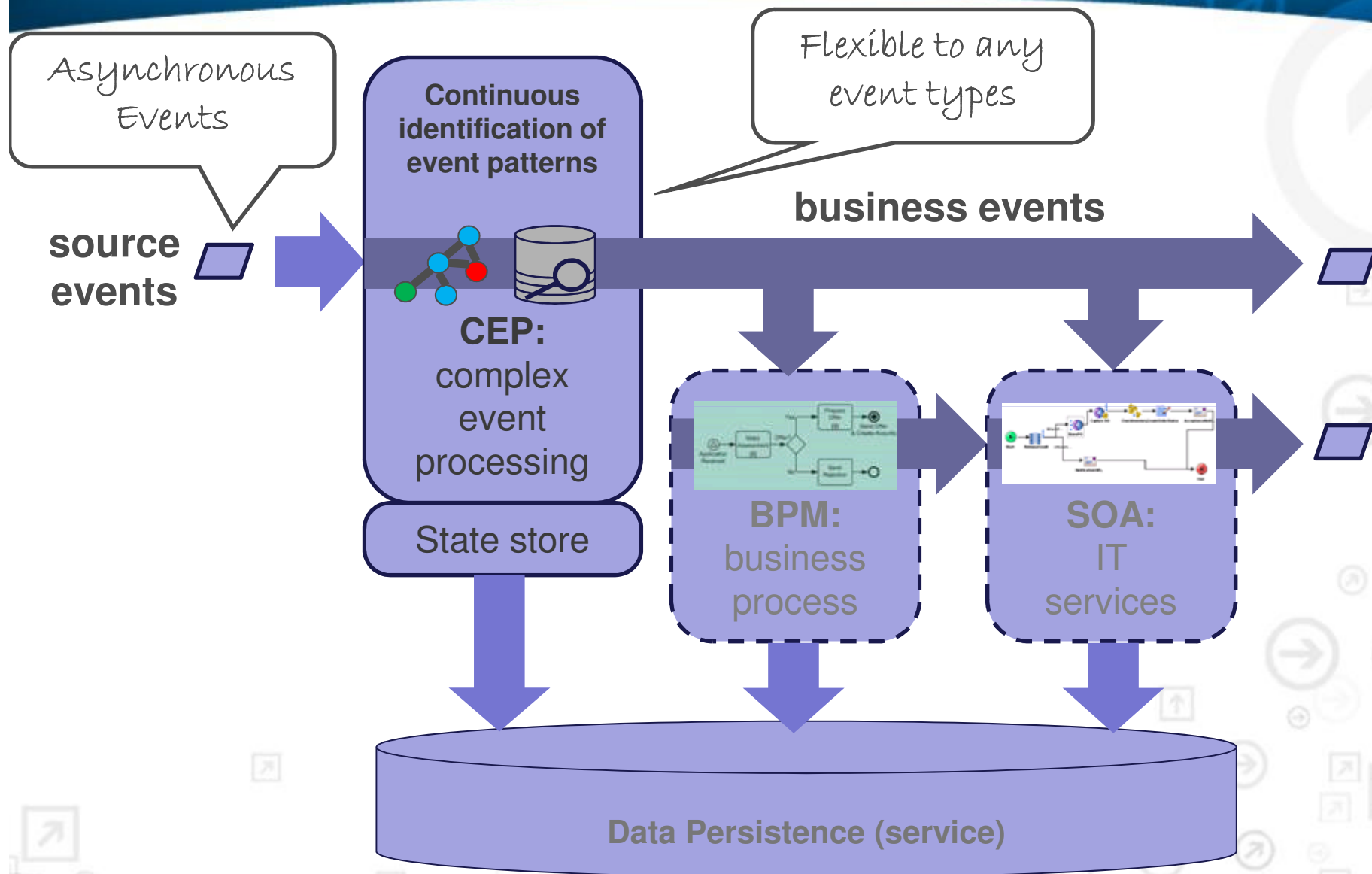
Oktober Rules Fest 2009

What's Different About Rules in CEP?

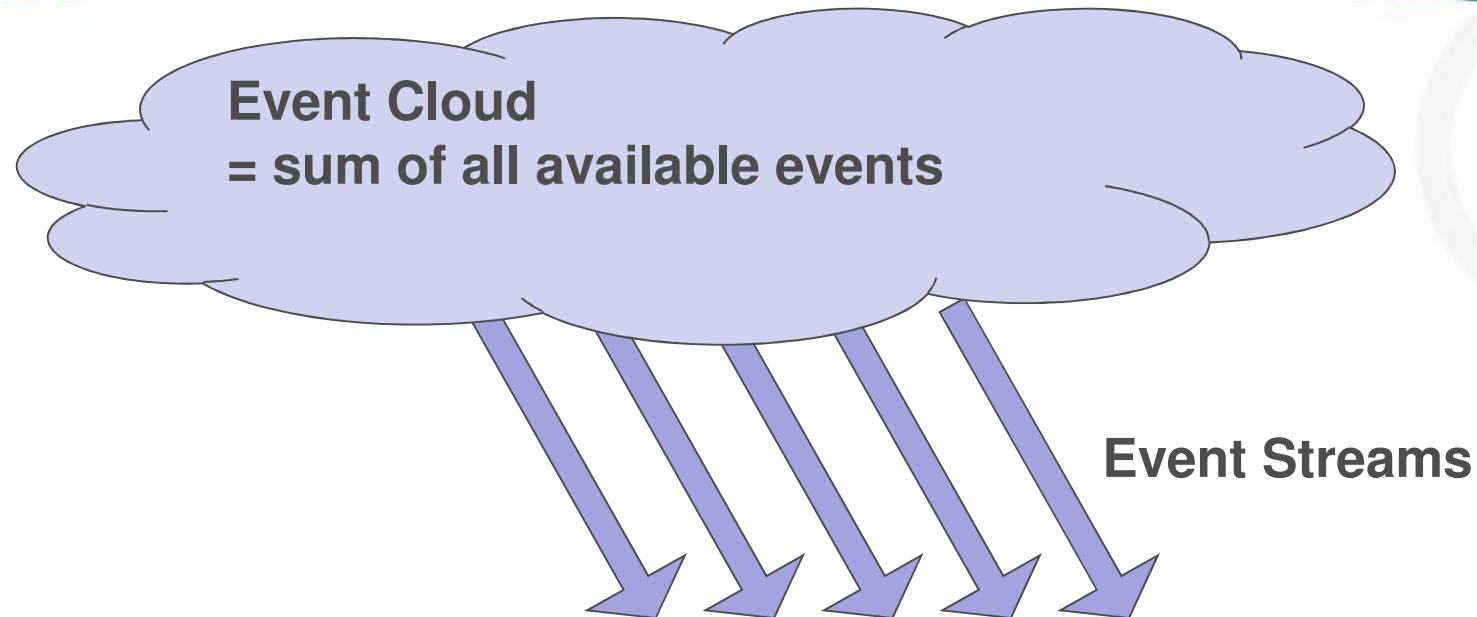
Paul Vincent, TIBCO Software – a CEP Company



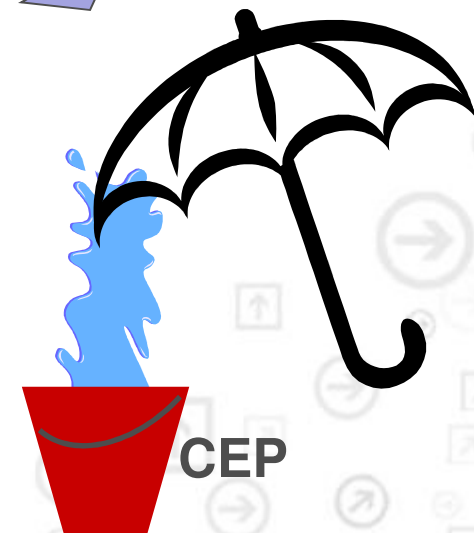
Complex Event Processing



CEP's terminology



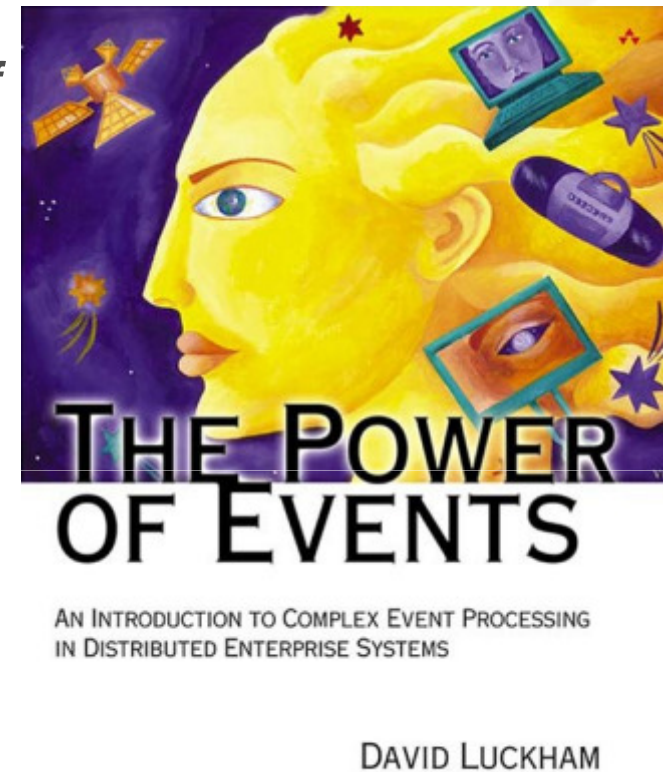
- ❑ CEP uses pattern detection to the event clouds & streams, and their histories
- ❑ Multiple modelling + execution paradigms available for pattern detection
- ❑ Problems solved: situation awareness, sense and respond, track and trace



What does CEP cover?

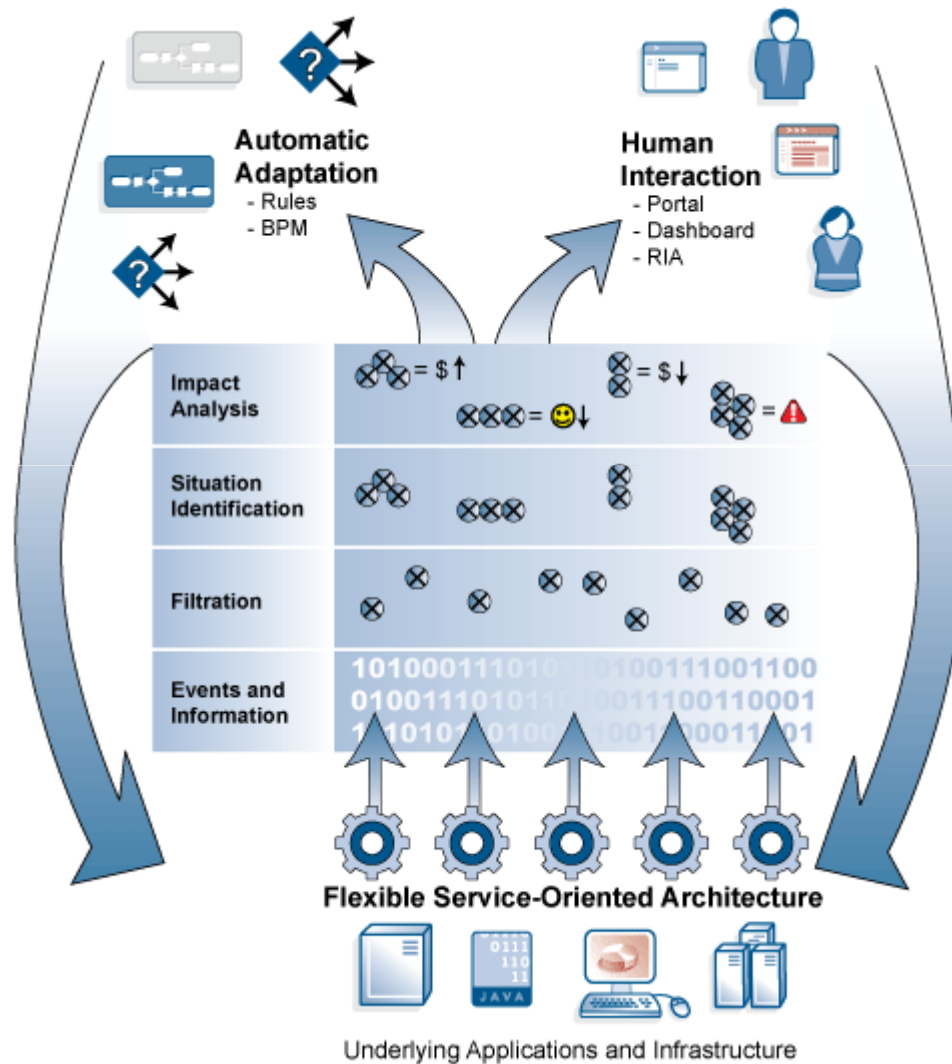
*“CEP applies to a very broad spectrum of challenges in information systems”
e.g.*

- Business process automation
- Service routing and coordination
- SLA, Policy fulfillment and breach checking
- Security and fraud detection
- Activity Monitoring

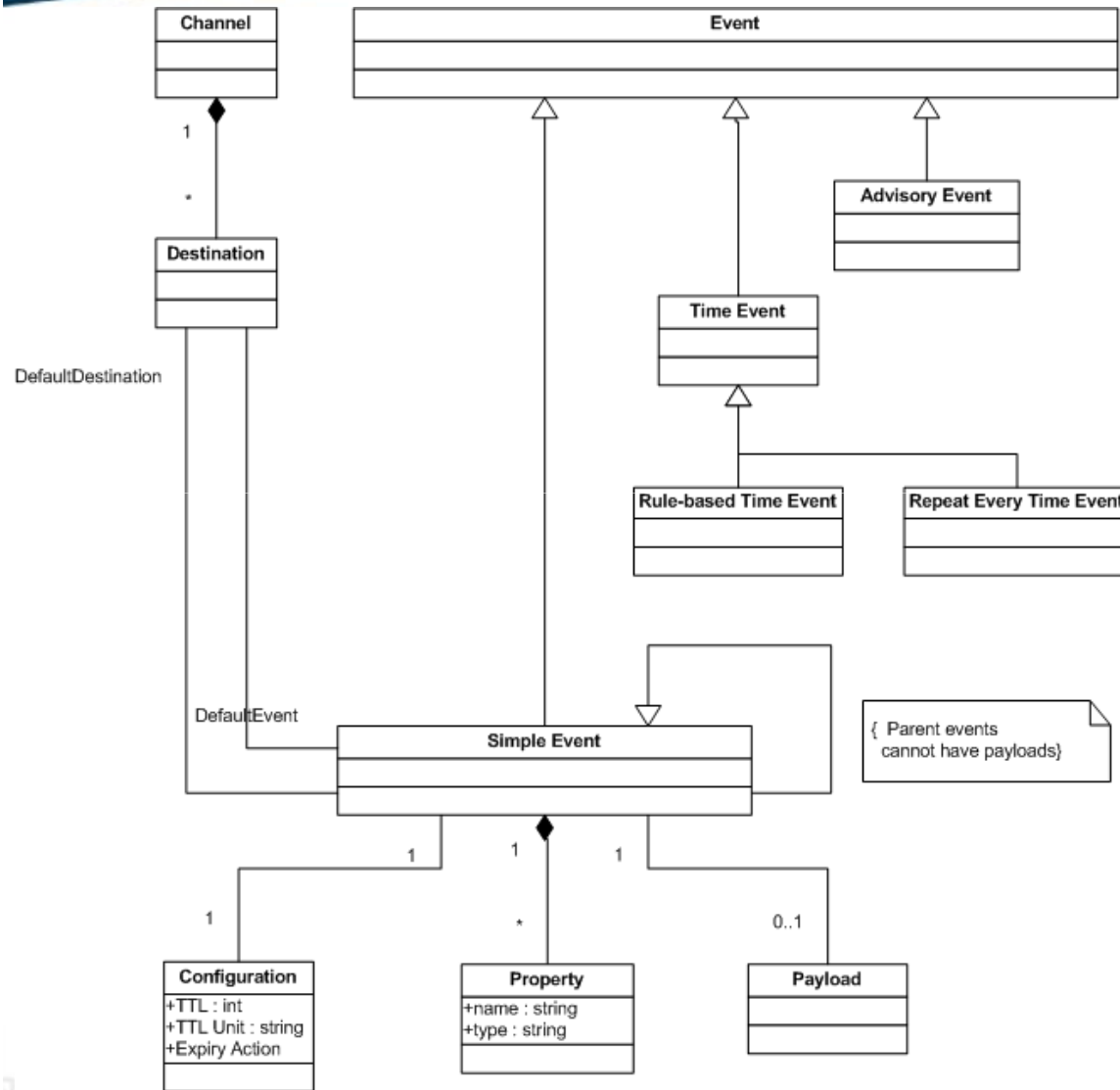


The Power of Events, Addison Wesley, ISBN: 0-201-72789-7, 2002

Complex Event Processing



What does CEP Need? 1. Events



- **Event** is sourced from **Channel** via address **Destination**
- Events can be **modelled hierarchically**
- Event metadata includes **Time To Live**, **Payloads** like XML, etc
- Events are **immutable** (except when...)
- Expected events may **not occur**

Event examples

- ❑ **SOA service requests**
 - time, destination, payload
- ❑ **Scans (parcel, baggage, RFID, production line...)**
 - location, time, payload
- ❑ **Web requests**
 - source IP, destination, payload, frequency
- ❑ **Messages / packets (telco, smartgrid)**
 - source, destination, time, location
- ❑ **Data streams (data feeds)**
 - payload, time, source

... 2. Pattern Matching for events

- ❑ Filter events
- ❑ Join events
- ❑ Events can be across time
- ❑ Events can be aggregated
- ❑ Events can be ordered

c.f. Rete

Event Store, State

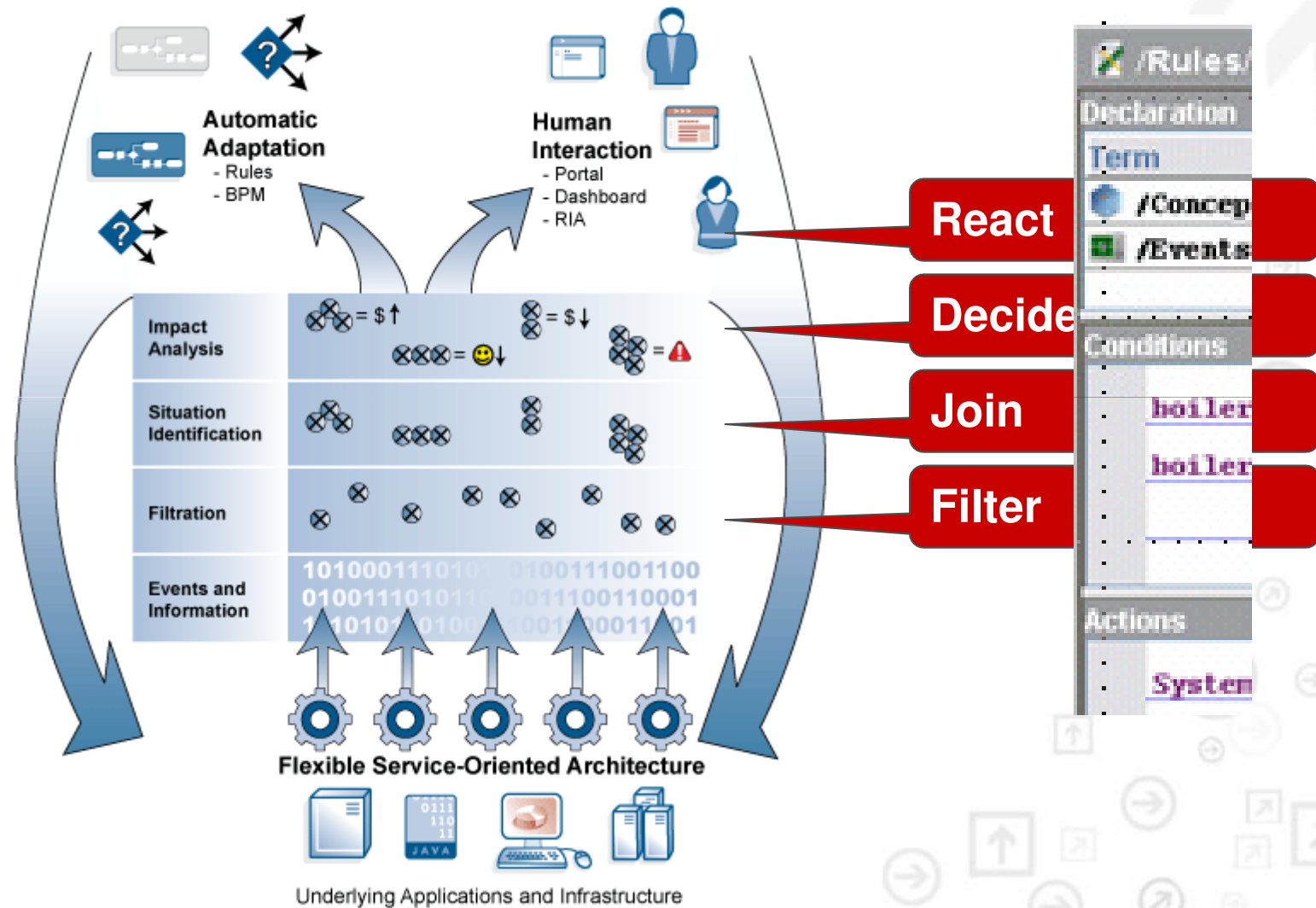
State

Collections

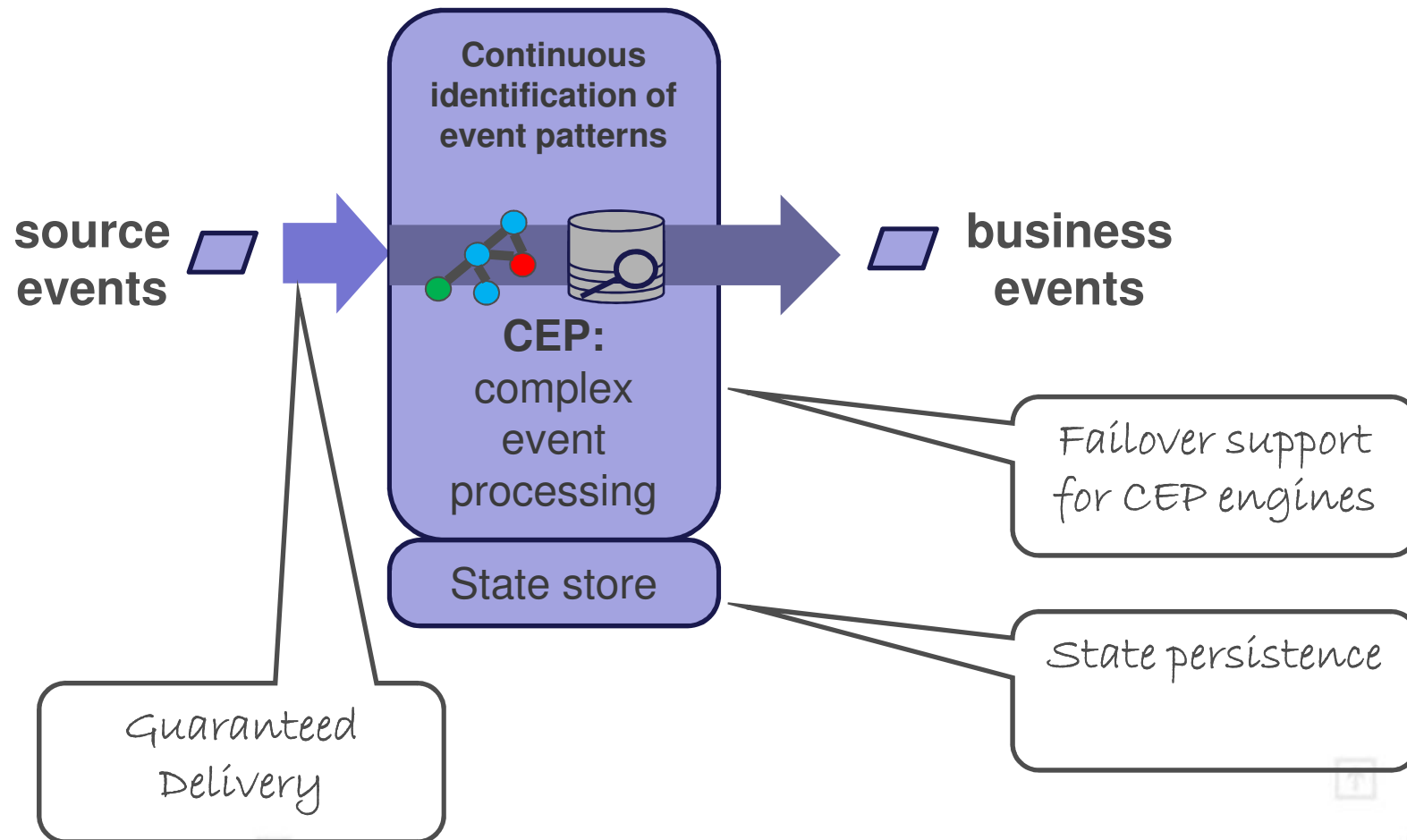
OO paradigms / facts

Queuing etc policies

... hence Rules for CEP



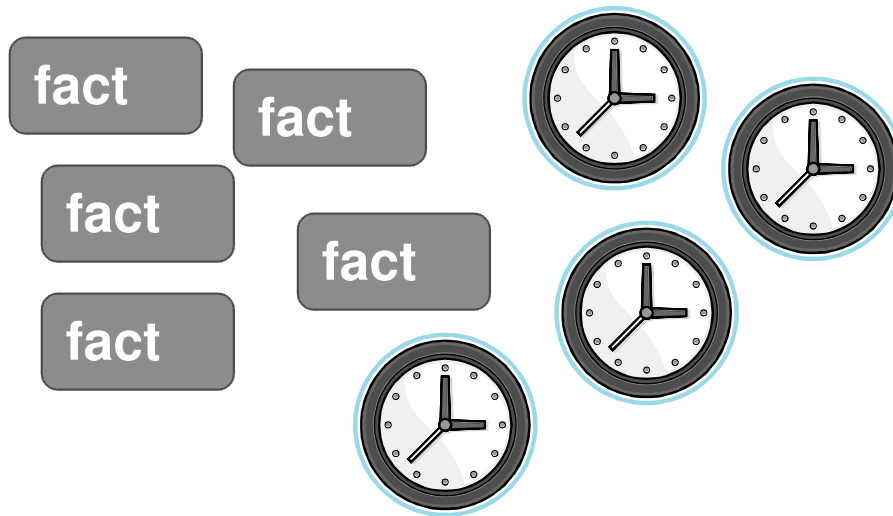
... 3. Stateful Support



Rules Require: A. Temporal Awareness

□ General “patterns”:

- Event occurs in time T
- Event doesn't occur in time T

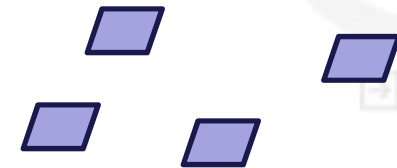


□ State “patterns”:

- State time-out

□ Event state “patterns”:

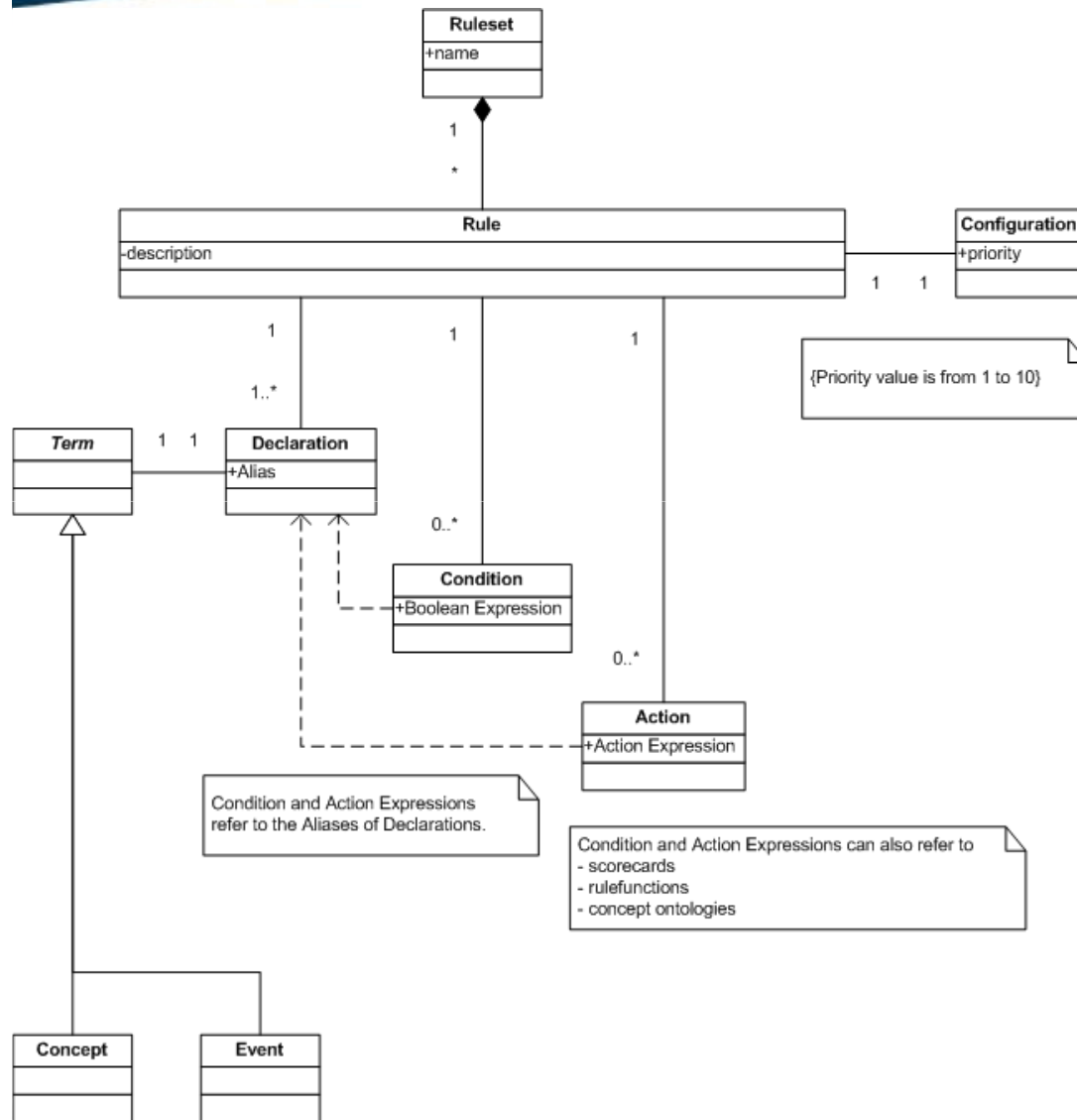
- Event time-out (via TTL)



□ Object history “patterns”:

- Prior values

Simple Rule Model

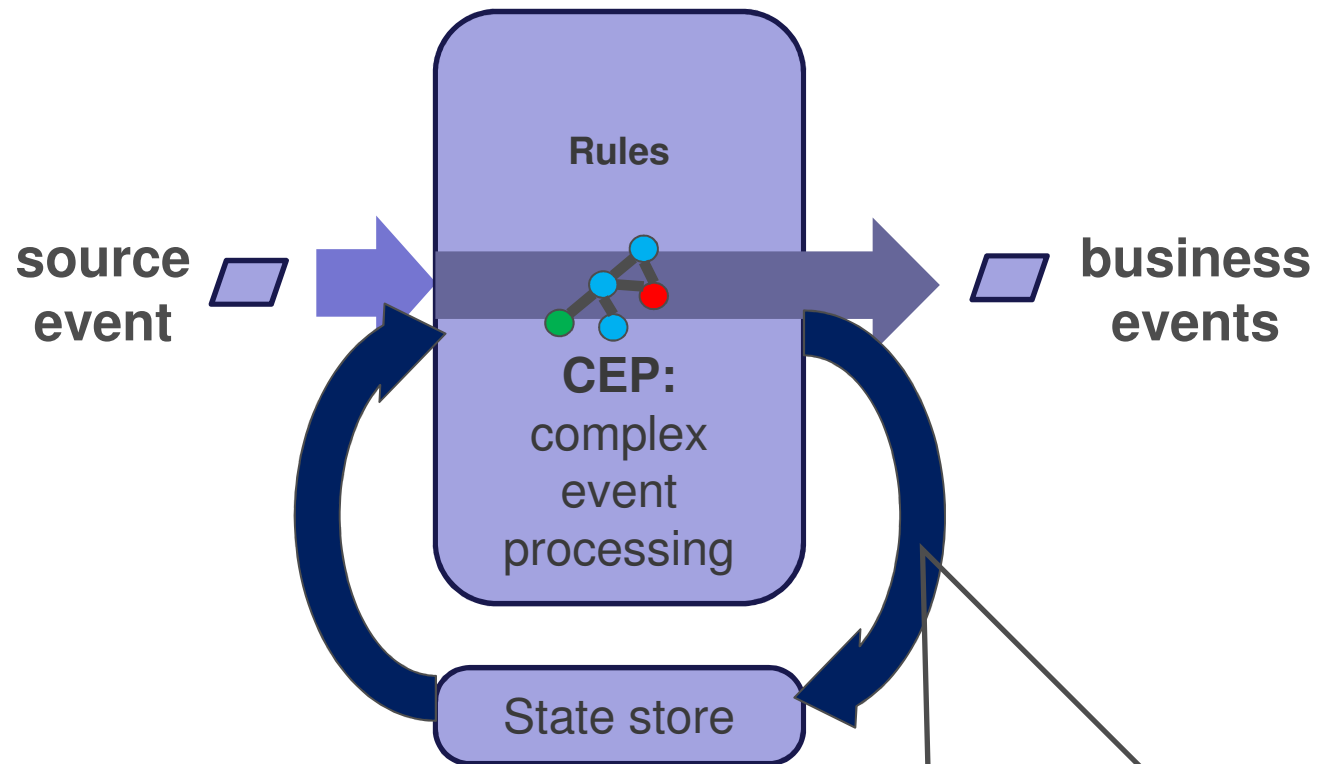


- **Rulesets** contain, organize **Rules**
- Rules are made up of **rule variables, conditions, actions**
- Rules execution context is a **Run To Completion** cycle
- New events can expire after a **single RTC**, or **on demand**, or **after some time**

Example Rule Types

- ❑ **Basic:**
Condition-Action
- ❑ **Triggers:**
Event-Condition-Action
- ❑ **Timers/schedulers:**
TimeUp-Action,
TimeInterval-Action
- ❑ **Event lifecycle:**
TimeToDie-Action

...B. Low latency, scalability



Event performance is dependent on minimizing RTC times

Example: fraud event processing

/Channels/RV

DebitTransaction

DebitTransaction (Destination)

Configuration

TIBCO Rendezvous

Name: DebitTransaction

Description:

Default Event: /Events/Debit.event

Serializer/Deserializer: com.tibco.cep.driver.tibrv.serializer.T

Subject: BE.DEBIT.TXN

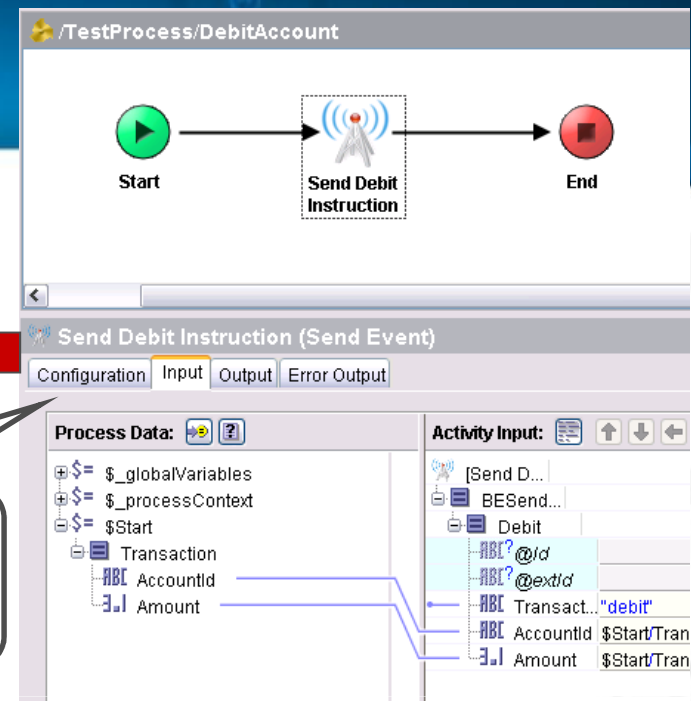
RVCN Pre Registration:

LimitPolicy: DISC

MaxEvents:

DiscardAmount:

Source Event definition / transform



Event definition:
Fire n' forget

Debit (SimpleEvent)

Configuration Properties Payload ExtendedProperties

Name: Debit

Description:

Time To Live: 0 Seconds

Inherits From:

Default Destination: /Channels/RV.channel/DebitTransaction

Expiry Action:

Example: fraud event processing rules

/Rules/ProcessDebits/ApplyDebit

Term	Alias
/Events/Debit	debit
/Concepts/Account	account

Conditions

```
//Checks whether the extId of an Account in  
//matches the incoming event's account ID  
account@extId == debit.AccountId;
```

Actions

```
//If Account Status is not Suspended, debit  
if (account.Status != "Suspended") {  
    account.Debits=debit.Amount;  
    System.debugOut("##### Deb  
    account.Balan  
    ##### Acc
```

/Rules/ProcessDebits/CheckNegativeBalance

Term	Alias
/Concepts/Account	account

Conditions

```
//Checks that the balance is less than zero  
account.Balance < 0;  
//Checks that Account status is not set to Suspended  
account.Status != "Suspended";
```

Actions

```
account.Status="Suspended";  
System.debugOut("##### Account ID <"+
```

/Rules/ProcessDebits/FraudDetection

Term	Alias
/Concepts/Account	account

Conditions

```
//1. Checks the number of debits in the  
Temporal.History.howMany(account.Debits,  
    DateTime.getTimeInMillis(DateTime  
    DateTime.getTimeInMillis(DateTime  
    true)  
    > FraudCriteria.num txns;  
  
//2. Checks the percentage of the average  
Temporal.Numeric.addAllHistoryDouble(acce  
    DateTime.getTimeInMillis(DateTime  
    > FraudCriteria.debits_percent*accou  
  
//Check whether Account status is not set  
account.Status != "Suspended";
```

Actions

```
account.Status="Suspended";  
System.debugOut("##### Account
```

Basic event processing

Event history processing / real time analytics

Name: ApplyDebit

Description:

Priority: 1 (Highest)

FraudDetection (Rule)

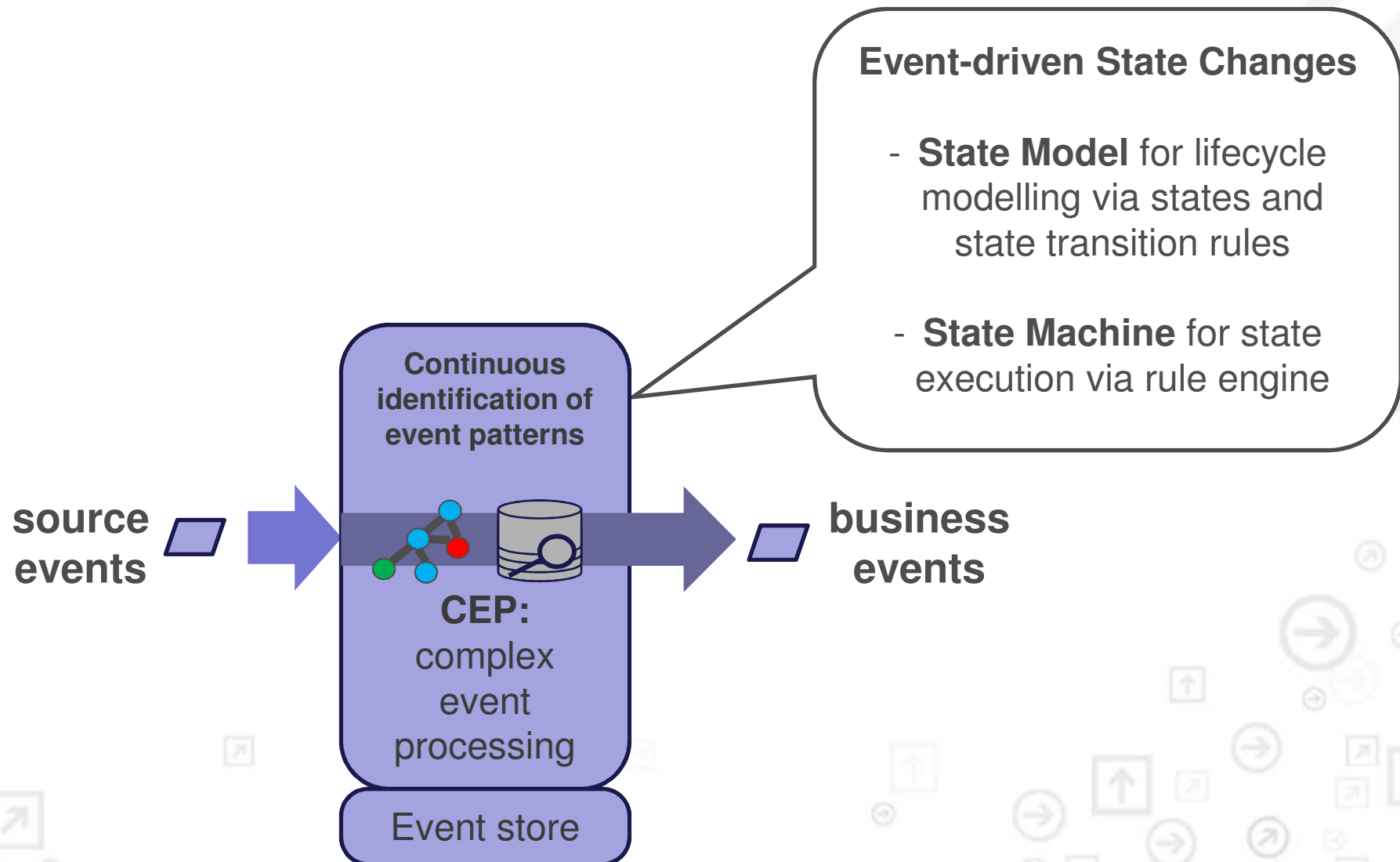
Configuration

16

© 2009 TIBCO Software Inc. All Rights Reserved. Confidential and Proprietary.

BCO[®]
of Now[®]

Model for Rule-driven CEP: event lifecycles via states



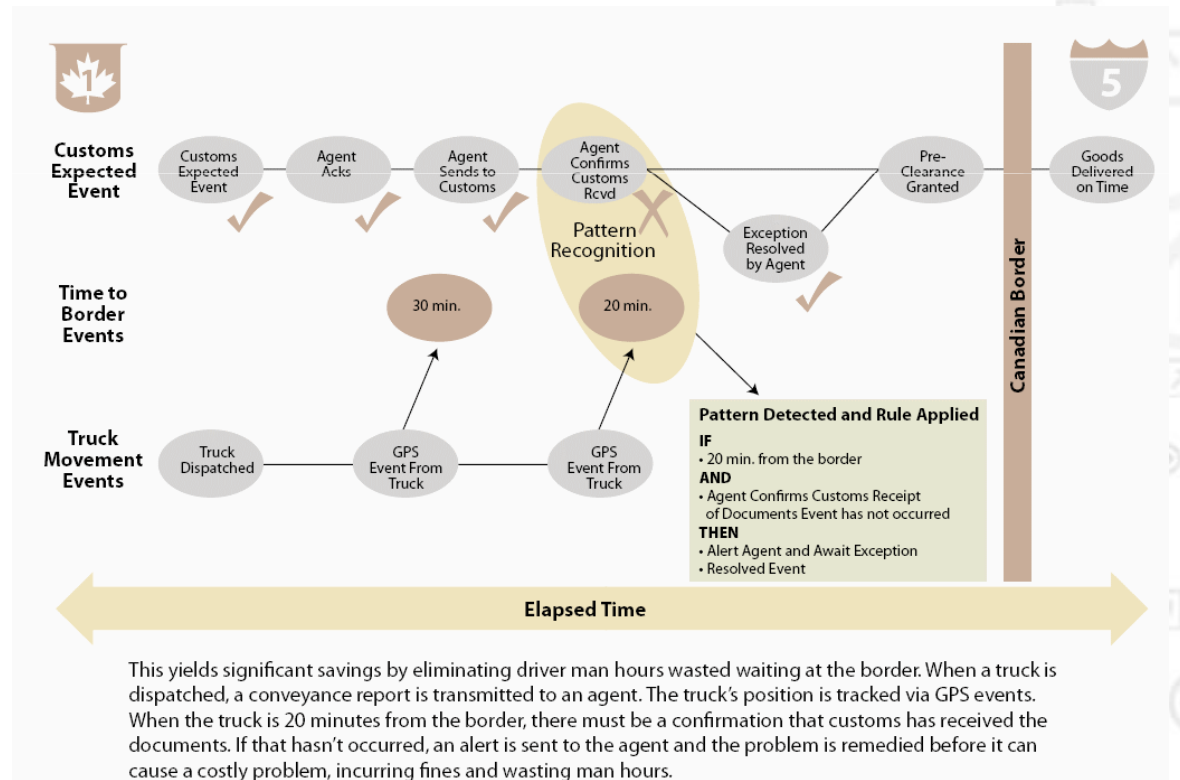
State Models in CEP

1. Visual modeling metaphor

- State diagram is simple to follow
- UML standard

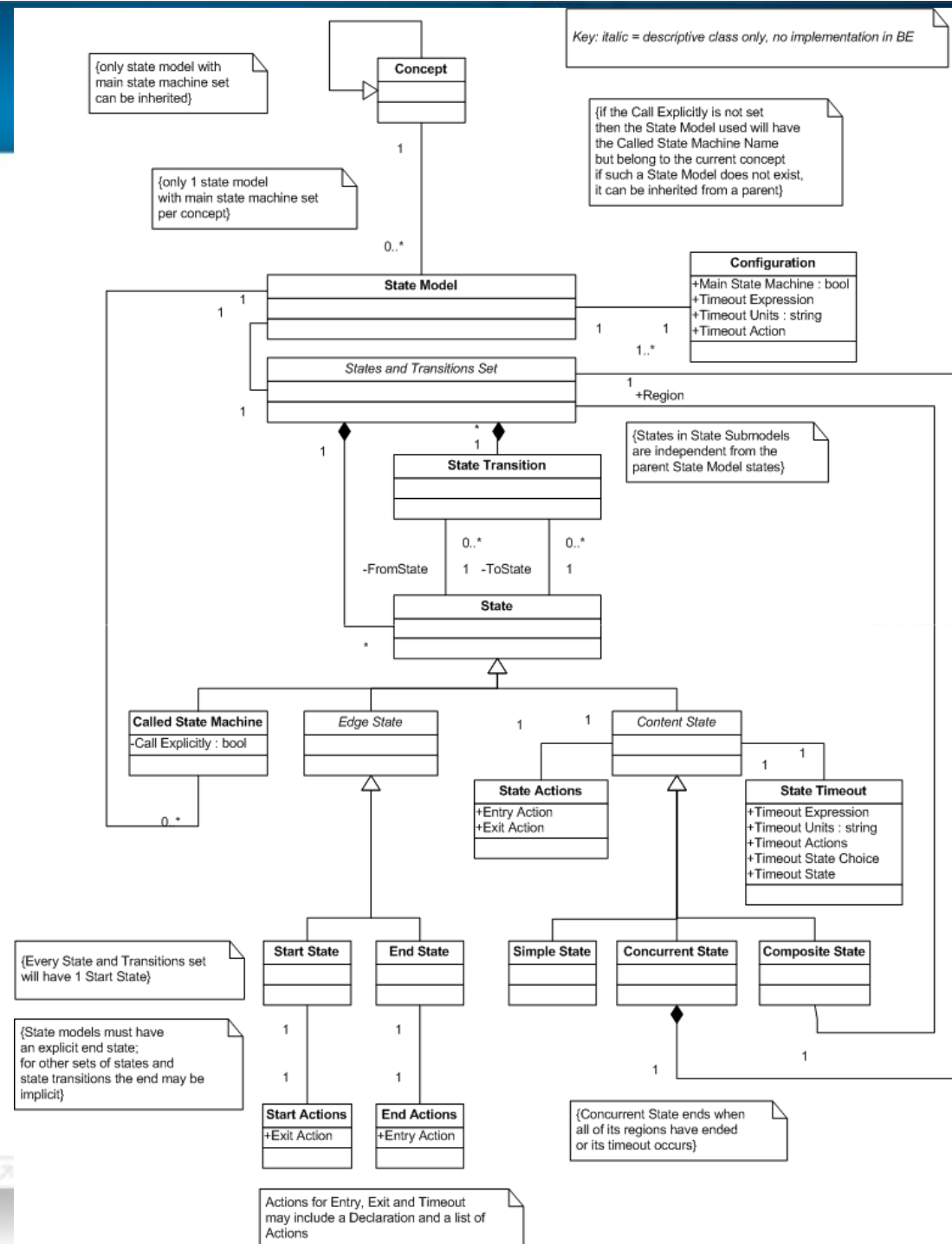
2. State / flow transitions are event or time-related

- Lifecycle is a set of states
- Missing events modelled through time-outs



State Models

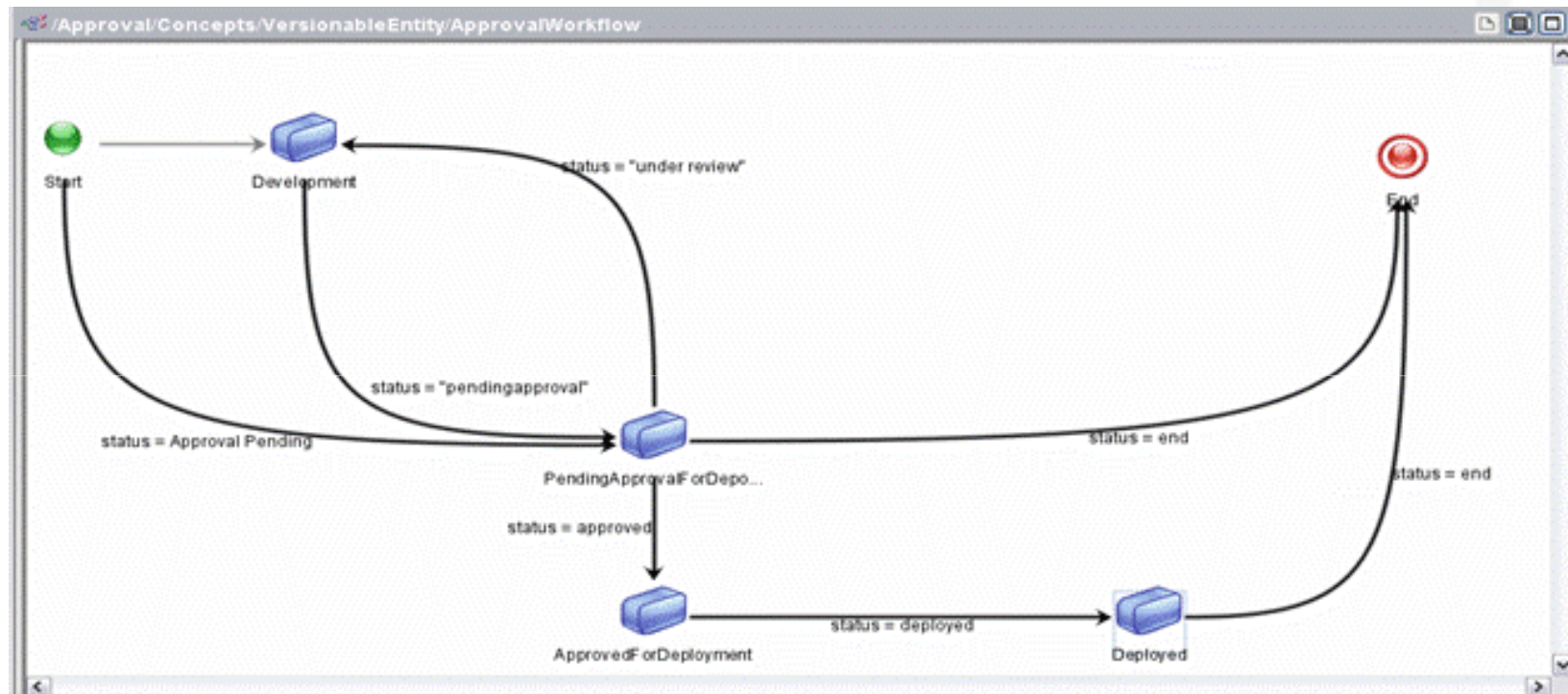
- **States** defined per instance, by type, can be **inherited**
- State transitions are defined by **events**
- Arrival / departure state events drive **actions**
- Multiple **concurrent states** can exist



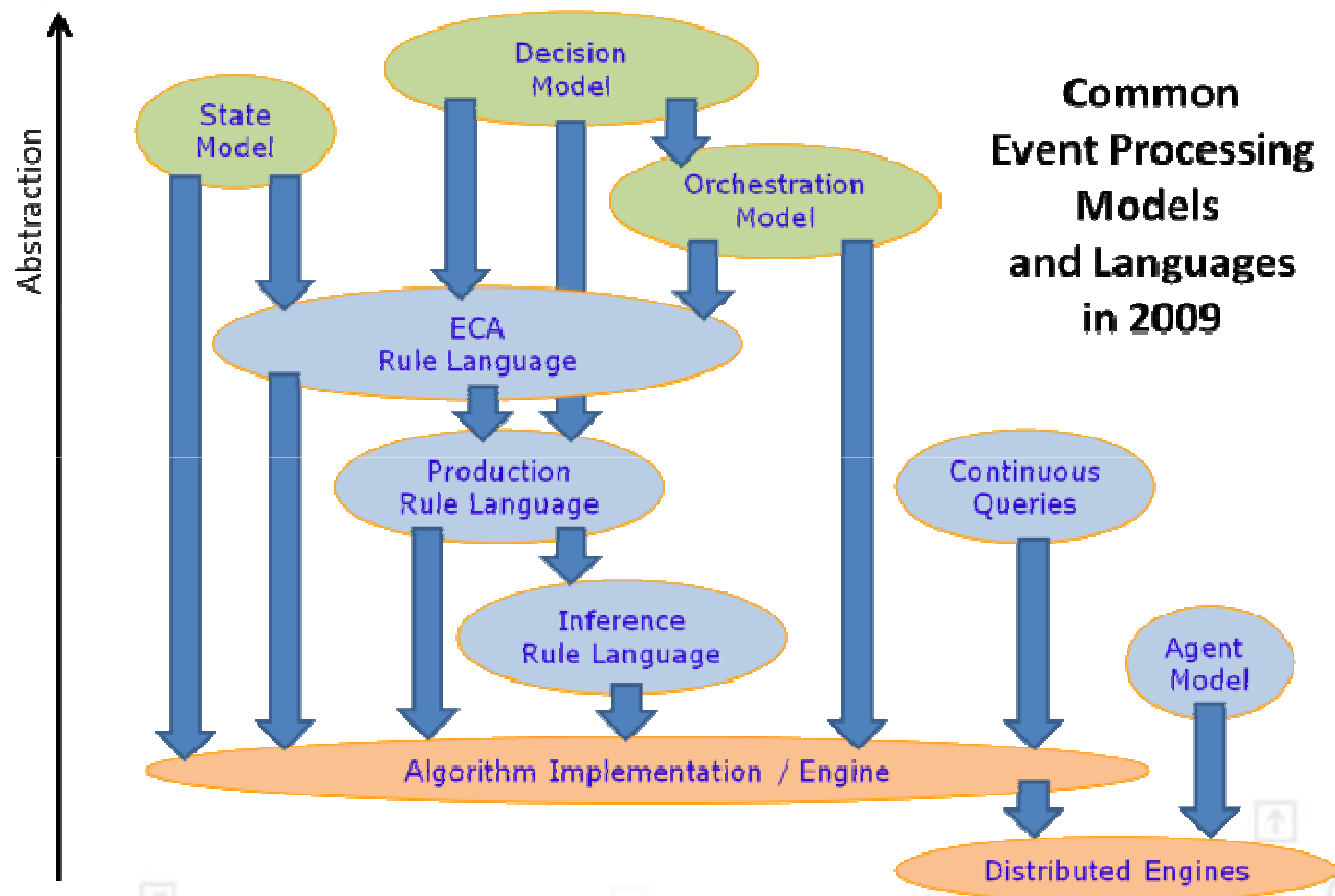
Example Rule Types (continued)

- ❑ **Basic: Condition-Action**
- ❑ **Triggers: Event-Condition-Action**
- ❑ **Timers/schedulers: TimeUp-Action, TimeInterval-Action**
- ❑ **Event lifecycle: TimeToDie-Action**
- ❑ **State transition:
Event-StateChange,
Timeout-StateChange,
StateEntry-Action,
StateExit-Action**

Example: state of rule management...



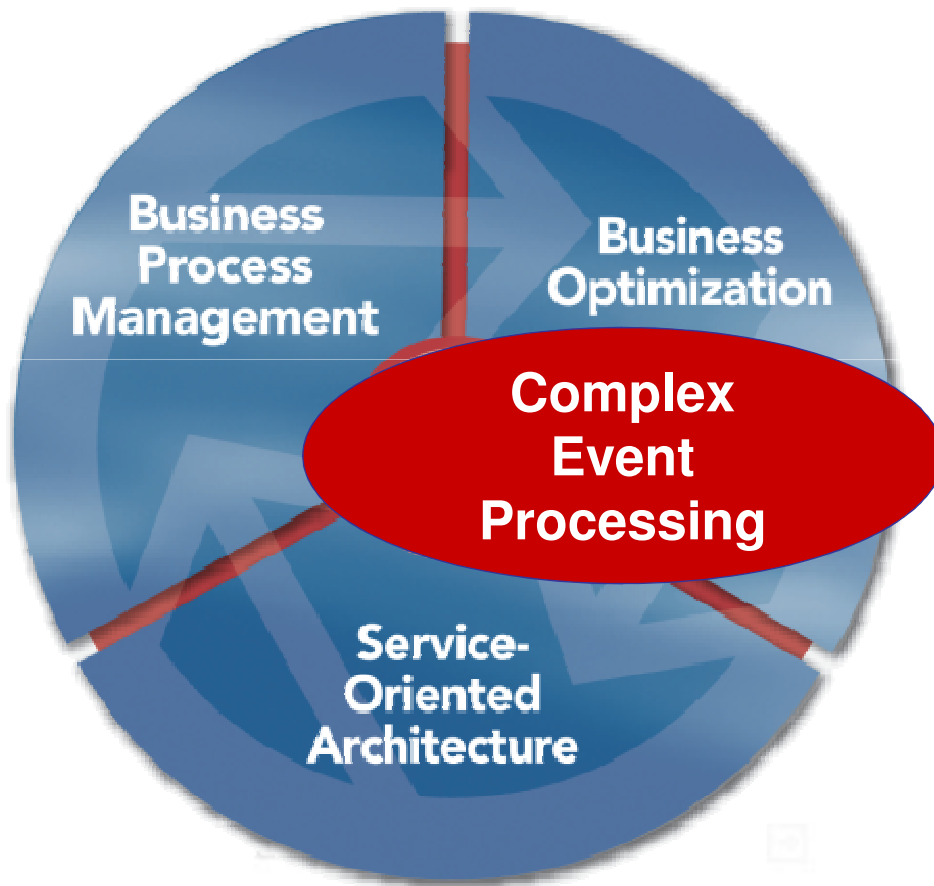
Basic Rule Types



©TIBCO Software 2009

Key: = EP graphical model; = EP language; = Code level; ➡ = Transform
Note: imperative code / scripts subsumed into Production Rule or Engine concepts

Summary



- Production Rules are excellent for many types of event pattern matching
- Stateful, event-driven temporal rule processing can provide a common view of events, data, events, state
- Resources:
www.ep-ts.com
www.tibcoblogs.com/cep