Tutorial on Advanced Design Patterns in Event Processing

Adrian Paschke
Freie Universität Berlin
Koenigin-Luisen-Str. 24/26, Germany
paschke AT inf.fu-berlin.de

Paul Vincent
Tibco Software Inc., UK
Business Optimization
Business Rules & CEP
pvincent@tibco.com

Alex Alves
Oracle Corp.
alex.alves@oracle.com

Catherine Moxey
IBM, UK
catherine_moxey@uk.ibm.com

ABSTRACT

We introduce a reference architecture for event processing, as defined by the EPTS reference architecture (RA) working group. An event processing reference architecture allows users to quickly create event processing solutions that adhere to known stakeholder requirements and architectural qualities. The focus in this paper is on the EPTS reference architecture description of the functional view which is supported by a mapping of its functions into design patterns as means to derive and prove these architectural descriptions to be usable solutions for recurring best practice implementations in common CEP languages.

Categories and Subject Descriptors

D.2.11 [**Software Architectures**]

General Terms

Design, Standardization, Languages.

Keywords

Complex Event Processing, Design Patterns, Reference Architecture

1. INTRODUCTION

Event Processing (EP) is considered an increasingly "mainstream" view in IT and is realized in different vendor products and solutions. These incorporate various software technologies that provide an event-oriented view of systems, providing continuous and stateful views of incoming events. The use of multiple software techniques in high performance stateful event processing has resulted in associated specialized software and system architectures. Potential adopters (stakeholders) need a reference to understand suppliers' architectures (reference architecture) and best practice solutions to recurring problems (design patterns).

To this end the Event Processing Technical Society (EPTS) set up a Reference Architecture group to provide a common Reference Architecture for event processing (EP). The goal is to define product- and system-independent abstractions and stakeholder views on current EP architectures, architectural practices, and design patterns.

A Reference Architecture predefines customizable abstract frames of reference for specific stakeholder concerns and application domains. It aids reuse of successful EP architectures for frequently occurring EP design problems. The description is base on a Reference Model that defines the terminology and components in Event Processing architectures.

The EPTS Reference Architecture (EPTS-RA) Working Group has developed a general Reference Architecture that supports multiple stakeholders with their specific views, interest and concerns. The functional view describes the functions of event processing (EP) / complex event processing (CEP) operations. [2] [6] This includes design and administration as well as runtime considerations.

The development of the reference architecture is closely aligned with the description of best practice CEP design patterns as means to derive and prove these architectural descriptions to be usable solutions for recurring CEP problems. [7][8] For the purposes of this tutorial we focus on the functions of the runtime aspect of EP/CEP, and drill down into some of them as design patterns [9] for some typical commercial and/or open source EP/CEP tools.

This Tutorial extends the work taught in last year's Event Patterns tutorial (Architectural and Functional Design Patterns for Event Processing at DEBS 2011 [9]). The tutorial is based on the EPTS Reference Architecture (version 1) with additional design patterns for Event Processing from a top-down (Reference Architecture) and bottom-up (sample code from popular event processing tools) perspective, based on the 4 levels of Event Processing Functions are classified loosely as Event Preparation, Event Analysis, Complex Event Detection and Event Reaction types.

This tutorial paper is structured as follows. In section 2 we introduce the underlying methodology and terminology for the reference architecture description. In section 3 we describe the EPTS Reference Architecture with two of its views – the logical view and the functional view. In section 4 we map these functions to design patterns. We conclude this paper in section 5 with a summary of the tutorial content and an outlook on future work of the EPTS-RA working group.

2. TERMINOLOGY AND METHODOLOGY

The EPTS Reference Architecture description follows the ISO/IEC 42010:2007 standard methodology for architectural descriptions of software intensive systems. [1] This methodology defines an architectural description as a collection of documentations of the architecture addressing different views on the system for different stakeholders.

The six important terminological elements are:

Architecture

The fundamental organization of a system embodied in its components, their relationships to each other, and to the environment, and the principles guiding ist design and evolution.

Architectural Description

A collection of products that document the architecture.

System

A collection of components organized to accomplish a specific function or set of functions.

System Stakeholder

A system stakeholder is an individual, team, or organization (or classes thereof) with interests in, or concerns relative to, a system.

View

A representation of the whole system from the perspective of a related set of concerns.

Viewpoint

A specification of the conventions for constructing and using a view - a pattern or template which to develop individual views by establishing the purposes and audience for a view and the techniques for its creation and analysis.

Furthermore, we distinguish between reference architecture and its underlying reference model:

Reference Architecture

A reference architecture **models the abstract architectural elements** in the domain independent of the technologies, protocols, and products that are used to implement the domain.

Reference Model

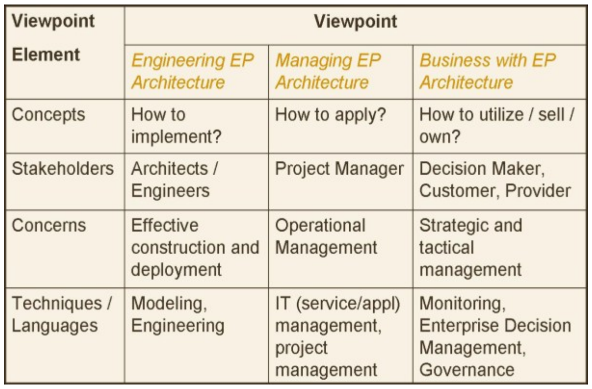
A reference model **describes the important concepts and relationships** in the domain focusing on what distinguishes the elements of the domain.

For further information on the underlying conceptual reference model and reference methodology we refer to [2]. For the terminology of event processing concepts and components used in the EPTS RA description we also refer to the EPTS Glossary. [3]

3. REFERENCE ARCHITECTURE DESCRIPTION

The EPTS RA group have combined "architectural diagrams" (representing product and system specific architecture information) from multiple sources - including vendors with customers using their technologies in everyday use - to derive some common architectural definitions. [6] Some of these input architectures are based on implemented system architectures while others are a generic representation of possible specific architectures.

For the EPTS Reference Architecture description we have currently defined two views – a *functional view* and a *logical view* – which address typical stakeholders such as the business analyst, the IT operations manager, the software architect / designer.



Viewpoint Element	Viewpoint		
	Engineering EP Architecture	Managing EP Architecture	Business with EP Architecture
Concepts	How to implement?	How to apply?	How to utilize / sell / own?
Stakeholders	Architects / Engineers	Project Manager	Decision Maker, Customer, Provider
Concerns	Effective construction and deployment	Operational Management	Strategic and tactical management
Techniques / Languages	Modeling, Engineering	IT (service/appl) management, project management	Monitoring, Enterprise Decision Management, Governance

Figure 1: Reference Architecture Viewpoints

In the following two main views on the reference architecture will be introduced: logical and functional architectures.

3.1 Logical View

The Logical View (of the Event Processing Reference Architecture) (see figure 2) describes the logical layout of Event Processing Agents (EPAs) [3] in an Event Processing Network (EPN) [3].

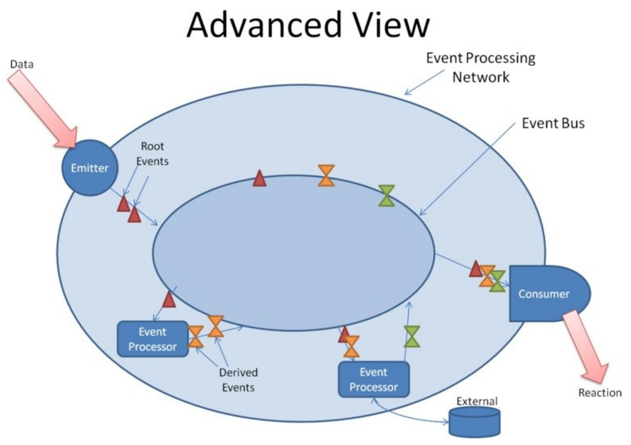


Figure 2: Logical View on EPTS Reference Architecture

It is important to note that the Logical View is an abstraction and not defined to some strict definition of EPA in terms of a concrete software technology which is used: the EPN represented in a logical view could itself form an "agent" in some higher level EPN. Conversely, an EPA in a Logical View could be defined as a subsidiary EPN. The Logical View is therefore an abstract view, and has the goal of describing the logical layout of EPAs relative to each other as well as to any specific event-handling network or distribution mechanism, from some perspective of interest (system or subsystem).

As a "view" of an EPN, the Logical View can itself be used in various contexts. Two typical uses or contexts are:

Business Logical View When a business analyst is viewing the overall event processing application, it is useful for them to understand the inputs, outputs and roles of the various EPAs.

Therefore an "overview" of the application as a "business logical view" may be useful.

Example: a business logical view may show the event flow from some sources through some EPAs and on through to management dashboards. This may be of interest because certain EPAs may be under the jurisdiction of different departments in the business.

Comparison: the business logical view may be compared to a high level process diagram or value chain, as a sequence of activities (in this case EPAs).

Operations Logical View When an IT Operations Manager is viewing the overall EPN, it is useful for them to understand the numbers of operating EPAs and their associated performance characteristics. Therefore an "operational overview" of the system as an "operations logical view" may be useful.

Example: an operations logical view may show active and inactive EPAs, and their associated operational Key Performance Indicators (KPIs), allowing operations staff to monitor the overall load on EPAs and thence deduce any operator actions (such as increase available resources for some EPA).

Comparison: the operations logical view is equivalent to systems monitoring dashboards and executive systems that are in widespread use in IT

The logical view of an EPN and associated EPAs can also be considered as a model, or diagram, of an overall Event Driven Architecture (or EDA).

3.2 Functional View

The Functional View of the EP Reference Architecture focuses on the functions that can be required within an event processing system – the functions that can be spread across the EPAs in the EPN. The following explanation references the Functional View shown in figure 3.

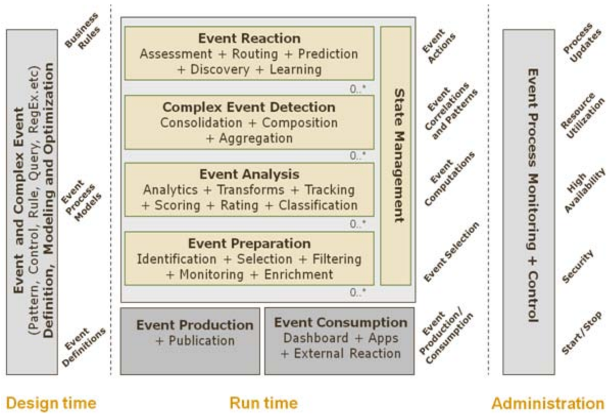

Event and Complex Event
(Pattern, Control, Rule, Query, RegEx.etc)
Definition, Modeling and Optimization
Business Rules
Event Process Models
Event Definitions
Event Reaction
Assessment + Routing + Prediction + Discovery + Learning
0..*
Complex Event Detection
Consolidation + Composition + Aggregation
0..*
Event Analysis
Analytics + Transforms + Tracking + Scoring + Rating + Classification
0..*
Event Preparation
Identification + Selection + Filtering + Monitoring + Enrichment
0..*
State Management
Event Production + Publication
Event Consumption
Dashboard + Apps + External Reaction
Event Actions
Event Correlations and Patterns
Event Computations
Event Selection
Event Production/ Consumption
Event Process Monitoring + Control
Process Updates
Resource Utilization
High Availability
Security
Start/Stop
Design time
Run time
Administration

Figure 3: Functional View on EPTS Reference Architecture

In addition to run time functions (when events are produced, processed and consumed) there are associated design time functions (including such things as definitions of events and patterns of events to be detected), and also administration functions (including security administration and ensuring qualities of service for the system). In the figure, the design time and administration functions are shown to either side of the run time functions. This functional view is primarily focussed on automated event processing operations, but it would also be possible for some or all of the functions to be manual operations. In the following we will describe the functions. For a compact overview see the appendix.

3.2.1 Design Time

At design time, the events and event-based rules relevant to the particular event processing system are identified. Raw events, simple events, and derived events can be defined, along with the patterns of events that are significant to the system and rules which define how the derived (or complex) events arise from the emitted root events. Modeling is a design-time activity in which the expected or possible behaviours of events in the system, and how the events are to be handled, are modelled. The actions to be taken (event reactions) when events or patterns of events are observed are also defined. Design time functions also provide a means to query on the event definitions, patterns, regular expressions etc., and offer scope for improvements to the way the system responds to events.

3.2.2 Run Time

At the most basic level -as was shown in the Logical View- at run time, raw events are produced by an event producer or emitter, some event processing is carried out resulting in derived events which are ultimately consumed by an event consumer or sink. Events can be emitted into the system by event producers which might be devices such as sensors, monitors or probes, or elements of a system such as business processes, services, or applications. Events can be made available for event processing by such means as publication of events into a communication system, or retrieval of events from an event detection system. After processing, events are can be consumed by actuators, by dashboards which display events as they occur to some user community (ranging from IT system operators to business analysts), by business processes, applications or services which are driven as a result of the events, or other external reaction to the events produced by the event processing system. For the purposes of the reference architecture, event consumers are regarded as downstream consumers of events generated in event processing. but event processing can itself be regarded as an event consumer, and event consumers can in turn act as producers of events.

The layout of the run time portion of the figure shows that between event production and event consumption, a number of event processing functions might be carried out, depicted as Event Preparation, Event Analysis, Complex Event Detection and Event Reaction. However, none or all of these might be involved, some might be carried out more than once (represented by the "0.." cardinality on each), and the ordering of the functions is not mandated (although there is some degree of logical ordering). Within each of the functions there are multiple subfunctions, of which representative samples are shown.

- *Identification* – An important step to perform before the analysis of events is to detect situations where events reference the same named entities (e.g., products, locations, persons, etc.) in different ways. In order for the subsequent steps of event processing (i.e., filtering, enrichment, correlation, etc.) to be accurate, we need a mechanism that will tag all these references with the same identifier. The above functionality can be achieved through the use of the

Entity Name System (ENS) [4] [5], which is a scalable infrastructure for assigning and managing unique, global identifiers for named entities.

- *Preparation* - Typically, the first processing is likely to be some form of selection from the events that have been received. The events may be Filtered on information in the event payload or in metadata, such that some subset is selected for further processing. Event adaption can convert events from an external format into some other format suitable for further processing, or for consumption. Events could also be Enriched by adding additional information to them from other data sources or from other events.

- *Analysis* - some form of computation might be carried out on events. Examples include: Identification of events, and potential removal of duplicates; Transformation of events, which normally acts on the event payload, and might convert the event to a different format, or normalize the event; Analytic techniques, which can include predictive capabilities such as trend computations; Tracking of events passing through the system, in terms of their location and time attributes; Scoring and Rating of events by computing values of events and their associated data, and Classification by identifying event types and associations.

- *Detection* - this processing covers aspects of deriving different ("complex" or derived) events from individual events, where multiple events might be Aggregated or Consolidated into a smaller number of events. Functions include Consolidation, in which additional event data can be added into complex events, Composition in which new events are created based on preceding events, and Aggregation, where information across multiple events is combined to provide summarized data. A key aspect to event detection is Pattern Matching, which enables new events to be derived by detecting a particular pattern of events, often across time, and event Correlation which allows related events to be correlated.

- *Reaction* - Event Reaction is identifying actions to be taken as a result of events, usually events arising from previous processing. This can include Assessment of a change in situation that should be acted upon and Routing of events to appropriate destination(s). Event reaction can also involve advanced capabilities such as Prediction of future events or behaviours, using machine learning or predictive analytics, application of business rules to make decisions based on the events, and even the Discovery of new event types, event patterns and analyses.

Note that the box within which all these event processing functions appear does **not** represent any deployment configuration. For example, filtering of events down to a subset, or transformation of raw events, could occur at the event producer rather than being carried out within an event bus. It would also be possible for the processing, or aspects of it, to be federated across a number of event processing systems. Coming back round the figure to event consumption, it is possible for there to be further event selection on the complex events that have been detected. Furthermore, any of the functions could be implemented as internal components or as external services.

Administration

Administrative functions involve monitoring the correct behaviour of the event processing run time system, ensuring adequate performance and availability, controlling resource utilization for performance and other means, making updates to the system, and managing the security of the system. The security considerations can range from securing access to design-time definitions and models, controlling deployment of aspects of the run time function, and securing the production and consumption of events.

4. MAPPING FUNCTION TO EVENT PROCESSING PATTERNS

In this section we will map the functions from the Reference Architecture into event processing (design) pattern. We use the following template for the description of the Event Processing Patterns

- **Name & Alternative Names**
- **Classification versus other references**
  - EPTS RA and other references e.g. "EP in Action" book [10]
- **Description**
  - Role, and Business Rule type specification / requirement
- **Structure**
  - EPTS Glossary terms
- **Implementations**

4.1 Identification Pattern

Identification: incoming data, events identified relative to prior events and event types

- *e.g. recognizing an event from data (event entity recognition, event extraction) and identifying the event to be of a particular event entity type*
- *e.g. associating a SMS event with particular mobile phone account*

Alternative Names

Event Lookup, Event Entity Recognition, Event Extraction

Classification

- EPTS Reference Architecture: *"Incoming events will need to be identified relative to prior events and event types, such as associating events with particular sources or sensors or recognizing / extracting events from data relative to event type information (event type systems, event ontologies)"*

Description

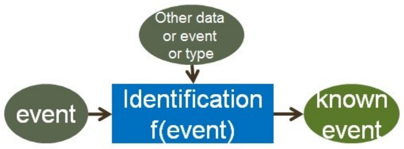
- Role / Explanation

Associating an event with some existing entity (data or past event, event type)

- Associated Business Rule Specification

A selection (to be enforced during the processing of events):

All <data, event entities> whose <attribute(s)> matches the <attribute(s)> of <some other event or data or type> is related by <relationship> to that <some other event or data or type>.



Structure

EPTS Glossary comparison

- Event type (event class, event definition, or event schema)
  - A class of event objects.
  - Event types should be defined within some type definition system ... will usually specify certain predefined data (attributes), examples of which might be: A unique event identifier used to reference the event

Implementations

see tutorial slides [11]

4.2 Selection Pattern

Selection: particular events selected for further analysis or pattern matching

- *e.g. Selecting the n'th event as a part of a sampling function*
- *e.g. Selecting events related to some existing event to allow enrichment of that existing event*

Alternative Names

Event Query (see also Event Identification, Filtering, Monitoring)

Classification

- EPTS Reference Architecture: *"During event preparation, particular events may be selected for further analysis. Different parts of event processing may require different selections of events. See also event filtering and event selectors in monitoring. "*

EPIA: *Filter*

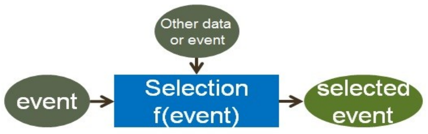
Description

- Role / Explanation

Selecting or locating an event due to some criteria or rules

- Associated Business Rule Specification

A selection (to be enforced during the processing of events): *All <event entities> whose <attribute1> matches the <attribute2> of <some other event or data> is selected for <processing>.*



Structure

EPTS Glossary comparison

- Event type (event class, event definition, or event schema)
  - A class of event objects.
  - Event types should be defined within some type definition system ... will usually specify certain predefined data (attributes), examples of which might be a unique event identifier used to reference the event

Implementation

see tutorial slides [11]

4.3 Filter Pattern

Filter: filter out all events that have some property in their payload / data

- *e.g. customer purchases: filter out those with values < $100*

Classification

- EPTS Reference Architecture**:** *"During event preparation, a stream or list of events may be filtered on some payload or metadata information such that some subset is selected for further processing."*
- EPIA: *Filter (EPA or Event Processing Agent) performs filtering only and has no matching or derivation steps, so it does not transform the input event.*

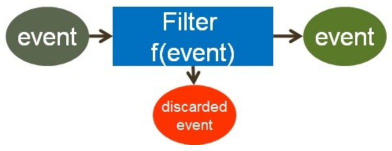
Description

- Role / Explanation

Comparing some property of the event (an attribute or metadata) with some other value (from some other event, or data)

- Associated Business Rule Specification

As a constraint: a selection (to be enforced during the processing of events):
All <event entities> that have <filter expression> must have <some status>.



Structure

EPTS Glossary comparison

A filter pattern can relate to several terms in the EPTS Glossary

- can be specified in an event pattern (A template containing event templates, relational operators and variables...)
- .. by an event pattern constraint (A Boolean condition that must be satisfied by the events observed in a system...)
- .. as part of an event processing rule (A prescribed method for processing events.)
- .. or as part of event stream processing (Computing on inputs that are event streams.)

Implementations

see tutorial slides [9]

4.4 Enrichment Pattern

Enrichment: add some information based on prior events

- *e.g. customer purchase: add the customer history to the purchase event*
- *e.g. enrich the event with semantic background knowledge*

Alternative Names

Event Annotation, Semantic Event

Classification

- EPTS Reference Architecture: *"During event preparation, events may be "enriched" through knowledge gained through previous events or data (including semantic background knowledge)."*
- EPIA: *Enrich (EPA): a translate EPA that takes a single input event, uses it to query data from a global state element, and creates a derived event which includes the attributes from the original event, possibly with modified values...*

Description

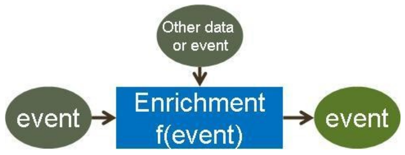
- Role / Explanation

Updating some attribute or metadata of the event with some other values, possibly computed, from some other event or data

- Associated Business Rule Specification

A selection (to be enforced during the processing of events):

All <event entities> that are also <enriched data expression> must have <some status>.



Structure

EPTS Glossary comparison

- Enrichment can be specified in an event pattern (pattern definition containing additional meta data, semantic knowledge)
- .. as part of an event processing rule (A method for enriching events during processing, e.g., by analysing the meta data and querying external data sources or inferencing semantic knowledge bases)
- .. or as part of event stream processing (additional information to event stream data, e.g. by querying external data sources)

Implementations

see tutorial slides [9]

4.5 Monitoring Pattern

Monitoring: particular event channels are monitored to identify events of interest

– *e.g. Selecting a particular channel and event selector in some middleware subscription*

Alternative Names

Event Subscribe; (see also Event Selection)

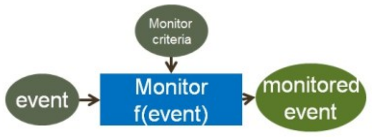
Classification

- EPTS Reference Architecture: *"During event preparation, particular types of events may be monitored for selection for further processing. This may utilise specific mechanisms external to the event processing such as exploiting event production features."*
- EPIA: *Filter*

Description

- Role / Explanation
  – Directing an external agent to select events
  – Applying some monitor function over some event channel
- Associated Business Rule Specification

A monitor (of an event source):
All <event entities> whose <attribute1> matches the <attribute2> of <some monitoring criteria> is selected for <processing>.



Structure

EPTS Glossary comparison

- Publish-and-subscribe (pub-sub)
  – A method of communication in which messages are delivered according to subscriptions .
  – Note 1. Subscriptions define which messages should flow to which consumers.
  – Note 2. Event processing applications may use publish-and - subscribe communication for delivering events. However, publish - and - subscribe is not definitional to event processing – other communication styles may be used.
- Subscriber
  – An agent that submits a subscription for publish- and – subscribe communication.

Implementations

see tutorial slides [11]

4.6 Consolidation Pattern

Consolidation: combine disparate events together into a main or primary event

– *e.g. multiple events occur to indicate a complex event; the complex event is given a new consolidate identity as derived primary event or main event*

– *e.g. abstraction of events into a situation which is initiated or terminated by the (derived complex) event as effect of the detection and consolidation of the complex event, e.g. the complex event "plane take-off" has the effect initiating the situation "flying"*

Alternative Names

Event Reinforcement from subevents, *Event / Situation Reasoning*

Classification

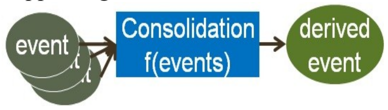
- EPTS Reference Architecture: *"During complex event detection, combining events together into a "main" or "primary" event. Similar to event aggregation", but with a new explicit derived event from the aggregated/composed events (the consolidated derived event might not contain all events from which it was derived); A special interpretation* *abstraction of the **effect** of a (derived) event is a **situation** which is initiated or terminated by events.*

- EPIA: *Aggregate (EPA): "a transformation EPA that takes as input a collection of events and creates a single derived event by applying a function over the input events."*

Description

- Role / Explanation
  - Consolidation is used to describe several events supporting the creation of a single composite event, generally where component events together provide evidence.
  - For example: a "conflict" event (with situation effect "in conflict) is considered to take place when both a "conflict declaration" occurs and an "act of violence" occurs.
- Associated Business Rule Specification

A consolidation (to be enforced during event processing):
The <collection of events> supporting <event> that are <related by some relationship constraint>.



Structure

EPTS Glossary comparison

- The term consolidated event is sometimes used for some forms of composite or derived event.
- Composite event: a derived, complex event
  - is created by combining base events using a specific set of event constructors such as disjunction, conjunction, sequence, etc.
  - always includes the base (member) events from which it is derived.
- Derived event (/synthesized event): an event that is generated as a result of applying a method or process to one or more other events.

Implementation

See tutorial slides [11]

4.7 Composition

Composition: composing new complex events from existing, possibly source, events

- *e.g. deduce a complex event from a history of past events*

Alternative Names

Event Creation from subevents

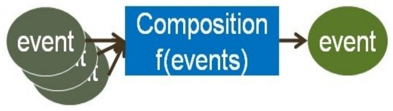
Classification

- EPTS Reference Architecture: *"Composite event …is created by combining base events using a specific set of event constructors (disjunction, conjunction, sequence, etc) + … always includes the base (member) events from which it is derived."*
- EPIA: *Aggregate (EPA): a transformation EPA that takes as input a collection of events and creates a single derived event by applying a function over the input events.*

Description

- Role / Explanation
  - Composition is used to describe combining multiple events into a composite event, generally where the component events justify the existence of the complex event.
  - For example: a telephone call event is considered to take place when a call end event occurs after a call start event.
- Associated Business Rule Specification

A composition (to be enforced during event processing):
The <collection> of <event entities> that are <related by some relationship constraint>.



Structure

EPTS Glossary comparison

- The term consolidated event is sometimes used for some forms of composite or derived event.
- Composite event: a derived, complex event
  - is created by combining base events using a specific set of event constructors such as disjunction, conjunction, sequence, etc.
  - always includes the base (member) events from which it is derived.
- Derived event (/synthesized event): an event that is generated as a result of applying a method or process to one or more other events.

Implementations

see tutorial slides [11]

4.8 Aggregation Pattern

Aggregation: add some information based on prior events

- *e.g. customer purchase history: track the sum total of order values for a customer*

Alternative Names

Event Summarization, *Composite Event*, *Complex Event*

Classification

- EPTS Reference Architecture: *"During complex event detection, combining events to provide new or useful information, such as trend information and event statistics. Similar to event consolidation "*
- EPIA: *Aggregate (EPA): a transformation EPA that takes as input a collection of events and creates a single derived event by applying a function over the input events.*

Description

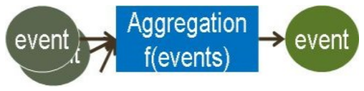
- Role / Explanation

Aggregation is used to describe several events as a single composite event, generally summarizing a metric of a set of component events.

For example: generate a (composite) event that contains the medium price of a stock over a 10 minute stream of events.

- Associated Business Rule Specification

A summarisation (to be enforced during event processing):
The <aggregation fn> of <event entities> that are <selection constraint> must have <some constraint>.



Structure

EPTS Glossary comparison

- The term aggregate event is sometimes used for some forms of composite or derived event.
- Composite event: a derived, complex event
  - is created by combining base events using a specific set of event constructors such as disjunction, conjunction, sequence, etc.
  - always includes the base (member) events from which it is derived.
- Derived event (/synthesized event): an event that is generated as a result of applying a method or process to one or more other events.

Implementations

see tutorial slides [9]

4.9 Assessment Pattern

Assessment: evaluate the event for inclusion in some process, collection, classification or complex event

– e.g. assess a cash withdrawal event for signs of it being a fraud event

Alternative Names

Event Classification

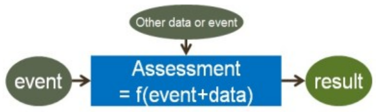
Classification

- EPTS Reference Architecture: *Event assessment is the process by which an event is assessed for inclusion in some process, incorporation in some other event, etc.*
- EPIA: no direct analogy, possibly maps to translate *Translate (EPA): a stateless Transformation EPA that takes a single event as its input, and generates a single derived event which is a function of the input event, using a derivation formula*

Description

- Role / Explanation
  - Assessment is used to describe the process of evaluating an event for the purposes of inclusion in some process, collection or classification.
  - For example: assess a parcel scan event to check whether it is classed as “in order” or “out of order” relative to some presumed parcel process
- Associated Business Rule Specification

A constraint (to be enforced during event processing):
The <event> that satisfies <assessment constraint>achieves a <boolean result>



Structure

EPTS Glossary comparison

- Constraint (also event pattern constraint): A Boolean condition that must be satisfied by the events observed in a system.
- Event sink (event consumer) is an entity that receives events.
  - Examples:
    - Software module
    - Database
    - Dashboard
    - Person

Implementations

see tutorial slides [11]

4.10 Routing Pattern

Routing: based on the type pass the event on to the appropriate service

– *e.g. customer purchase event: pass on to a provisioning service based on the type of product / product classification*

Alternative Names

Event Summarization, Composite event, Complex event

Classification

- EPTS Reference Architecture: *"During event reaction, event routing is the process by which an event is redirected to some process, computation element, or other event sink. "*
- EPIA**:** no direct analogy, possibly maps to split / compose / project - *Project (EPA): a translate EPA that takes an input event and creates a single derived event containing a subset of the attributes of the input event.*

Description

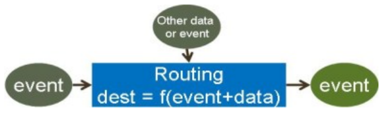
- Role / Explanation

Routing is used to describe the process of adding one or more new destinations to an event.

For example: route an input event to the appropriate specialist agent for that event type.

- Associated Business Rule Specification

A constraint (to be enforced during event processing):
The <event> that satisfies <selection constraint> must be assigned to <destination>.



Structure

EPTS Glossary comparison

- Routing (a process on events) is not defined, but is related to associating an event to an event sink.

- Event sink (event consumer) is an entity that receives events.

Implementations

see tutorial slides [9]

Additional patterns such as patterns for event reactions, e.g., event prediction, event analytics, learning events, are addressed in the tutorial [11] .

5. CONCLUSION AND FUTURE WORK

In this paper we have described the methodology and approach for the EPTS Reference Architecture. We have introduced two views – the logical and the functional view - and have focused on the functional Reference Architecture description. We have described several Event Processing Design Patterns for the functions of the Reference Architecture in order to support developers and tools. In future work we will evolve the event patterns to other types of pattern descriptions, as defined in the categorization model for CEP pattern. [7]

6. APPENDIX

This appendix gives a compact description of the functions defined in the functional view on the EPTS Reference Architecture.

6.1 Design Time Functions

Covers the definition, modeling, improvement /maintenance of the artifacts used in event processing:

- event definitions, including event metadata and payloads,
- event and event object organisations and structures,
- event processing transformations / queries / rules / procedures / flows / states / decisions / expressions (although these can sometimes be considered as administrative updates in some situations)

6.2 Administrative concepts of monitoring and control

This may involve

- starting and stopping the application and event processing elements, including application monitors
- providing and updating security levels to event inputs and outputs (also can design-time)
- management of high availability and reliability resources, such as hot standby processes
- resource utilisation monitoring of the event processing components
- process updates, such as how-swapping of event processing definitions to newer versions.

6.3 Runtime Functions

Event Production: the source of events for event processing.

- Event Publication: As a part of event production, events may be published onto a communication mechanism (eg event bus) for use by event consumers (including participants in event processing). This is analogous to a "push" system for obtaining events.
- Event Retrieval: As a part of event production, events may be explicitly retrieved from some detection system. This is analogous to a "pull" system for obtaining events.

Event Consumption: the process of using events from event publication and processing. Event processing itself can be an event consumer, although for the purposes of the reference architecture, event consumers are meant to indicate downstream consumers of events generated in event processing.

- Dashboard: a type of event consumer that displays events as they occur to some user community.
- Applications: a type of event consumer if it consumes events for its own processes.
- External Reaction: caused through some event consumption, as the result of some hardware or software process.

Event Preparation: the process of preparing the event and associated payload and metadata for further stages of event processing.

- **Entity Identification**: Incoming events will need to be identified relative to prior events and event types, such as associating events with particular sources or sensors or recognizing / extracting events from data relative to event type information (event type systems, event ontologies).
- **Event Selection**: particular events may be selected for further analysis. Different parts of event processing may require different selections of events. See also event filtering and event monitoring.
- **Event Filtering**: a stream or list of events may be filtered on some payload or metadata information such that some subset is selected for further processing.
- **Event Monitoring**: particular types of events may be monitored for selection for further processing. This may utilise specific mechanisms external to the event processing such as exploiting event production features.
- **Event Enrichment**: events may be "enriched" through knowledge gained through previous events or data (including semantic background knowledge).

Event Analysis: the process of analysing suitably prepared events and their payloads and metadata for useful information.

- **Event Analytics:** the use of statistical methods to derive additional information about an event or set of events.

- **Event Transforms:** processes carried out on event payloads or data, either related to event preparation, analysis or processing.

- **Event Tracking:** where events related to some entity are used to identify state changes in that entity.

- **Event Scoring:** the process by which events are ranked using a score, usually as a part of a statistical analysis of a set of events. See also *Event Analytics*

- **Event Rating:** where events are compared to others to associate some importance or other, possibly relative, measurement to the event.

- **Event Classification:** where events are associated with some known or to be learned classification scheme for use in downstream processing. See also *Event Identification*

Complex Event Detection: the process by which event analysis results in the creation of new event information, or the update of existing complex events.

- **Event Consolidation:** combining disparate events together into a "main" or "primary" event (which may not contain its base (member) events from which it was derived). See also *event aggregation.*

- **Event Composition:** composing new, complex events from existing, possibly source, events.

- **Event Aggregation:** combining events to provide new or useful information, such as trend information and event statistics. Similar to *event consolidation*, but always includes its base (member) events.

Event Reaction: the process subsequent to event analysis and complex event detection to handle the results of analysis and detection.

- **Event Assessment:** the process by which an event is assessed for inclusion in some process, incorporation in some other event, etc.

- **Event Routing:** the process by which an event is redirected to some process, computation element, or other event sink.

- **Event Prediction:** where the reaction to some event processing is that some new event is predicted to occur.

- **Event Discovery:** where the reaction to some event processing is the disclosure of a new, typically complex, event type.

    - Note that event prediction is predicting some future event, usually of a known type, whereas event discovery is the uncovering of a new event type. See also event-based learning.

- **Event-based Learning:** the reaction to some event processing that uses new event information to add to some, typically statistical-based, understanding of events.
  Note that event-based learning is a specialisation of general machine learning and predictive analytics.

7. ABOUT THE EPTS REFERENCE ARCHITECTURE WORKING GROUP

The Event Processing Technical Society Reference Architecture Working Group started March, 2009. It currently has 18 members and is co-chaired by Adrian Paschke (RuleML) and Paul Vincent (TIBCO). Since July 09 it is merged with the EPTS Metamodel Working Group under a joint charter. Its scope is:

- Define **architecture patterns** that are compatible with EPTS members' Event Processing solutions and products.
- Define **terminology and components** regarding Event Processing in accordance with EPTS
- Identify and utilize **best practices and methods** for technical architecture descriptions and interchange
- **Liaise with relevant standards bodies** for EP metamodels and reference architectures

8. ACKNOWLEDGMENTS

This work has been conducted within the scope of the Event Processing Technical Society Reference Architecture working group.

9. REFERENCES

[1] ISO/IEC 42010:2007 Standard http://www.iso.org/iso/catalogue_detail.htm?csnumber=45991, accessed June 2009.

[2] Paschke, A. and Vincent, P.: A reference architecture for Event Processing. In *Proceedings of the Third ACM International Conference on Distributed Event-Based Systems* (DEBS '09). ACM, Nashville, USA, 2009

[3] David Luckham, Roy Schulte: EPTS Glossary Version 2, Event Processing Technical Society, 2010

[4] Heiko Stoermer, Themis Palpanas, George Giannakopoulos. The Entity Name System: Enabling the Web of Entities. International Workshop on Data Engineering meets the Semantic Web (DESWeb), in conjunction with the IEEE International Conference on Data Engineering (ICDE), Long Beach, CA, USA, March 2010

[5] Zoltan Miklos, Nicolas Bonvin, Paolo Bouquet, Michele Catasta, Daniele Cordioli, Peter Fankhauser, Julien Gaugaz, Ekaterini Ioannou, Hristo Koshutanski, Antonio Mana, Claudia Niederee, Themis Palpanas, Heiko Stoermer. From Web Data to Entities and Back. International Conference on Advanced Information Systems Engineering (CAiSE), Hammamet, Tunisia, June 2010

[6] Paschke, A., Vincent, P., Moxey, C., Alves, A., Palpanas, T.: Event Processing Architectures, *Fourth ACM International Conference on Distributed Event-Based Systems* (DEBS '10). ACM, Cambridge, UK, 2010.

http://www.slideshare.net/isvana/debs2010-tutorial-on-epts-reference-architecture-v11c, accessed Feb 2011

[7] Paschke, A.: Design Patterns for Complex Event Processing, 2nd International Conference on Distributed Event-Based Systems (DEBS'08), Rome, Italy, 2008.

[8] Opher E.: Tutorial Event Processing Architecture and Pattern, 2nd International Conference on Distributed Event-Based Systems (DEBS'08), Rome, Italy, 2008. http://www.slideshare.net/opher.etzion/tutorial-in-debs-2008-presentation, accessed June 2009

[9] Paul Vincent, Alexandre Alves, Catherine Moxey, Adrian Paschke: Architectural and functional design patterns for event processing, *Fith ACM International Conference on Distributed Event-Based Systems* (DEBS '11). ACM, New York, USA, 2011. http://www.slideshare.net/isvana/epts-debs2011-event-processing-reference-architecture-and-patterns-tutorial-v1-2, accessed April 2012

[10] Opher Etzion, Peter Niblett: Event Processing in Action. Manning Publications Company 2010: I-XXIV, 1-360

[11] Adrian Paschke, Paul Vincent, Alexandre Alves, Catherine Moxey: Tutorial on Advanced Design Patterns in Event Processing, *6th ACM International Conference on Distributed Event-Based Systems* (DEBS '12). ACM, Berlin, Germany, 2012