Event-driven Reactivity
A Survey and Requirements Analysis

SYSTEMATIC THOUGHT LEADERSHIP FOR INNOVATIVE BUSINESS

Kay-Uwe Schmidt, SAP

02.06.2008

Co-Authors:
Dakro Anicic, FZI
Roland Stühmer, SAP

Agenda

1. **Introduction**
2. Survey of Event-triggered Reactivity
3. Requirements
4. Future Work

Introduction

Event-driven processing becomes ever important in various application domains

- Ranging from traditional business applications, like supply-chain management
- To the entertainment industry, like on-line gaming applications

The market value should increase tenfold by 2010 and should reach something like $4bn in total (IBM).

Key role of even-driven processing for making business more agile

Main benefit of "eventizing" business systems

- Event processing introduces a kind of reactive dynamics
- Enabling active responding on signals sensed/derived from the current context

Event-triggered Reactivity (EtR)

- Opens great opportunities for system/process improvements.

Agenda

1. Introduction
2. **Survey of Event-triggered Reactivity**
3. Requirements
4. Future Work

Event Condition Action Rules

First mention of the ECA paradigm by U. Dayal, A. P. Buchmann, and D. R. McCarthy in 1988

- “Rules are objects too: A knowledge model for an active, object-oriented databasesystem”. In Lecture notes in computer science on Advances in object-oriented database systems, pages 129-143, New York, NY, USA, 1988. Springer-Verlag New York, Inc.
- This paper describes work in progress on the knowledge model (an extended data model that includes constructs for representing rules) of HiPAC, an active, object-oriented DBMS.

“Central to our knowledge model is the concept of *event-condition-action (ECA) rules,* which generalizes the many different mechanisms introduced previously in the literature to support active DBMS functions.

- The *event* part of an ECA rule specifies database operations, temporal events, or signals from arbitrary processes;
- the *condition* part specifies database queries;
- and the *action* part specifies a program.
- When the event occurs (is *signaled,* the condition is evaluated; if the condition is *satisfied,* the action isexecuted.”
- **ON** event **IF** condition **DO** action

Event Processing

The two kinds of event processing:

- Complex Event Processing (CEP)
- Event Stream Processing (ESP)

Dealing with different problems in event processing using different approaches

ESP – extraction of simple events from a stream

- Events are totally ordered by time
- Emphasis of ESP on efficiency for high throughput and low latency
- Algorithmic stock trading

CEP – extraction of complex event patterns from a cloud

- Only a partial temporal order of events
- Other partial order of interest for CEP is for instance causality
- More time and memory needed
- Business Process Monitoring

CEP is a superset of ESP

CEP and ESP nowadays adopt each others approaches

Event Detection Languages

Development for active databases in the late 80’s and early 90’s

- use complex event specifications to facilitate database triggers
- not only listen to simple events but observe complex combinations of events until the trigger procedures are executed

Simple events carry a type, their occurrence time and possibly other parameters

Creation of complex nested expressions, using operators like And, Or, Sequence, and others

Complex events are detected from occurrences of one or more of simple or complex events

Structure of event patterns: event operators

A nested event operator might have several event types as arguments

An event detector for the given pattern functions as a stream pattern matcher and listens for events that satisfy the type constraints and together satisfy the semantics of the given operator, e.g. occurred in sequence

Complex Event Detection Algorithms

Finite State Automata

- Ode 1992
- Transformation of complex event expressions into deterministic finite automata
- Convenient model to define the semantics of complex event operators
- Downside no acceptance of overlapping occurrences of the same complex event

Colored Petri Nets

- SAMOS 1994
- Convenient model to define the semantics of complex event operators
- Also the detection of overlapping occurrences is possible

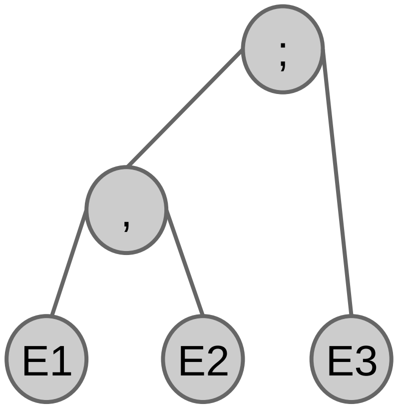
Graph-based Approaches

Graph-based Approaches

Sentinel based on Snoop, SnoopIB in the mid 90's

Construction of the graph from the event expressions

Example: (E1,E2);E3



The graph is a directed acyclic graph and generally does not form a tree for two reasons: nodes may have several parents, when their represented expression is part of more than one complex events, and secondly there is no single root node, when there is no overarching, single most complex event.

Overview of the State of the Art in Event and Action Processing

ECA Rules

- **Event Processing**
  - **Complex Event Processing**
    - Graph-based Approaches
    - Finite State Automata
    - Colored Petri Nets
  - **Event Stream Processing**
    - Pattern Matching in Streams then DB Tables, Sliding Windows
    - StreamSQL
- **Action Processing**
  - **Production Rules**
    - Rete (Forgy in 1982)
    - TREAT (Miranker 1990)
    - LEAPS (Batory 1994)
  - **Logic**
    - IBM Tivoli Enterprise Console (Prolog)
    - Bry, Paschke 2007

Agenda

1. Introduction
2. Survey of Event-triggered Reactivity
3. **Requirements**
4. Future Work

Requirements for Handling EtR

Efficient combination of complex event processing with action processing

Comprehensive framework capable to deal with

- Semantics of events and actions
- Termination of rule processing
- Rule ordering

Context and Situation

- Event + Condition + Context = Situation

Reasoning over all reactivity rule constituents

Vision Event-triggered Reactivity

Holistic approach: Event-driven Reactivity

Unique handling of the different constituents of an event-driven architecture

- events, actions, conditions, contexts and situations

Realization of next generation efficient and manageable event-driven (reactive) applications

Decrease the complexity of setting-up/evolving event-driven applications, that nowadays requires lots of manual work, especially in defining what an event is

Increase the benefits (added value) of such applications, which are currently constrained on the complex monitoring of events

Open new possibilities to apply them in highly dynamic and distributed environments

Vision Requirements

Efficient modeling of the sense-and-respond (reactive) nature of a system, especially its contextualization

Comprehensive management of the reactivity life cycle of a system, including automatic discovery of relevant situations, efficient detection of events reasoning about actions

Efficient implementation of the reactivity life cycle management

Context Detection

In fact, we argue that the ECA (event-condition-action, such as it is) model is too simple presentation of the (intelligent) event processing nature

Efficient context detection process is inevitable for the efficient event processing and is totally neglected in the literature

Unified mechanism for formal representation of all phases in the reaction cycle is needed for efficient and complex event processing

Usage of a richer conceptual model for describing reactions on events

Context as a first class citizen

Context as a first class citizen in the event processing is currently completely missing in the literature for event processing

Context-based event processing is the next "big thing“

- Shaping of the future of the computing
- Context-Driven Architecture (CoDA) is the most promising paradigm that will extend SOA

New possibilities for event triggered reactivity by reasoning about situations and context

Reasoning Services about situations and context

Opens new possibilities for event triggered reactivity

Situations as formal logic models; some very interesting reasoning services can support the whole event processing

The system can check formally the consistency of the system and backtrack if a conflict (meaning inconsistency in the system) is to appear.

Another service would be the synchronization of situations if we consider that two or more reactions will run in parallel, which is a quite natural assumption in the rich-event systems.

Agenda

1. Introduction
2. Survey of Event-triggered Reactivity
3. Requirements
4. **Future Work**

Future Work

Development of a new conceptual model and architecture of event-triggered reactivity (EtR)

- Introducing novel concepts (situation and context)
- Its formal, logic-based representation

Development of a model for managing the whole life cycle of EtR, including

- Language for modeling EtR concepts (e.g. situations, context)
- User-friendly editor based on pattern modeling metaphor
- Methods for ensuring the consistency of such a rule base and its interoperability with other reactive systems
- New methods and tools for the automatic discovery of complex event and situation patterns from stream data by taking into account their evolution as well
- New algorithms for scalable ECA reasoning, based on the selected logic and its implementation in a new reasoning engine that will serve as the event-, condition- and action-handler in a reactive system.
- Realize, test and refine an integrated software framework for the management of EtRs life cycle, containing elements of the distributed event processing, that can be easily deployed in the selected legacy landscape
- Development of use cases, their implementation, testing and evaluation in real-world pilot studies in order to validate proposed model and framework.

Thank you!

Definition and halftone values of colors

RGB 4/53/123	RGB 240/171/0	RGB 204/204/204	RGB 153/153/153	RGB 102/102/102	**Primary colors 100%**

RGB 68/105/125	RGB 21/101/112	RGB 85/118/48	RGB 119/74/57	RGB 100/68/89	**Secondary colors 100%**
RGB 96/127/143	RGB 98/146/147	RGB 110/138/79	RGB 140/101/87	RGB 123/96/114	**85%**
RGB 125/150/164	RGB 127/166/167	RGB 136/160/111	RGB 161/129/118	RGB 147/125/139	**70%**
RGB 152/173/183	RGB 154/185/185	RGB 162/180/141	RGB 181/156/147	RGB 170/152/164	**55%**
RGB 180/195/203	RGB 181/204/204	RGB 187/200/172	RGB 201/183/176	RGB 193/180/189	**40%**

100%	RGB 73/108/96	RGB 129/110/44	RGB 132/76/84		
85%	RGB 101/129/120	RGB 148/132/75	RGB 150/103/110		
70%	RGB 129/152/144	RGB 167/154/108	RGB 169/130/136		
55%	RGB 156/174/168	RGB 186/176/139	RGB 188/157/162		
40%	RGB 183/196/191	RGB 205/197/171	RGB 206/183/187	RGB 158/48/57	**Tertiary color**

Grid

SAP RESEARCH