Key words:
CEP, business process, business rules, adaptive rules, HFD, market data

Eva ZÁMEČNÍKOVÁ, Jitka KRESLÍKOVÁ[[1]](#footnote-1)\*

DEsign of adaptive business rules model for high frequency data processing

In this paper we would like to discuss high frequency data processing and the use of complex event platform in combination with business rules approach. For such a high volume of data it is suitable to use complex event platform (CEP), because CEP allows for big data processing in real time. We would like to focus on improvement of decision making process under the condition of dynamical adaptation of the process on the fly. We will use pattern recognition for detecting and predicting the trends in data by mining this information from historical data. After the distinguishing patterns we will build the set of business rules according to which the process runs and we will control the process flow by defining the restrictions.

We would like to use this model for building trading systems. Algorithmic trading applies complex event processing by calculating complex algorithms that indicate when to sell or buy based on real-time processing. Market data can be viewed as events. This data needs to be analyzed in real time in order to identify the trends in data and to react to these trends automatically. Traditional approach for detecting anomalies on stock market has been statistical analysis, but a CEP-based approach is able to react faster than the traditional approach.

1. INTRODUCTION

Nowadays there exist many models (or rather, whole platforms) for big data processing. By the notion of big data we generally understand high volume of source data from multiple sources like live streams, databases etc. For processing of such a high volume of data we found as the most suitable the Complex Event Platform (CEP). Lately CEP (also abbreviated as complex event processing) is in the spotlight mainly because of its capability to process data in real time. In the next chapters there will be given the definition and basic knowledge about this platform and how it processes the data. More detailed insight will be devoted to decision making process and authors will suggest the use of adaptive business rules to control the flow of process events. First chapter shortly introduces complex event processing to reader of this paper. In the following chapter, a more detailed information about CEP and its solution is provided and the rules for real-time processing are explained. Third chapter discusses high frequency data domain and its applications in financial markets. The fourth chapter covers the theme of business rules and the use of these rules for controlling of data processing and the decision making part. The conclusion and the ideas for future research will be provided at the end.

2. Complex event processing

Complex event processing (CEP) is an emerging technology that generates actionable knowledge from distributed message-based systems, data streams and historical data in real time or near real time. There are some CEP engines but only few are capable of integrating data from multiple sources and working with high event volumes. Environmental measurements processing and making wide-spread use of it is a big data problem. The emphasis on real-time data processing is becoming an essential requirement.

According to [1] we can abstract many different levels of CEP. For example if we consider trading and financial markets, at the lowest level there will be a stock trader responsible for executing trades. A trade might consist of several executed bids, offers, payments and other financial transfers. Complex event, indicating how much profit or loss the trader generated during some time interval has predictive power which can be used in following decisions [1, 3].

Complex event processing offers its users a way to automate the detection of anomalies or other detectable and exploitable phenomena. It is way too tedious for the auditor to correlate all the trades made by all the traders to detect all the various errors they might have done. On the other hand, a CEP system, if properly configured, is able to react faster than a human. At this point, we would like to use a power of adaptive rules which can speed up the process by an automatic set up of process on the fly as a response to a specific change in context. Traditional approach for detecting anomalies on stock market has been statistical analysis after the trades were completed. For fast reactions for frauds, real-time monitoring is a necessity and can be used as a complementary method. The approach based on streaming data proved to be several orders of magnitude faster than a traditional approach based on relational database and single issue queries [5].

2.1 Events

There are two parallel definitions for the concept of an „event“. First, it can mean anything that happens, or is happening, e.g. a financial trade is made. Second, it may be seen as the object acting as the manifestation of something that happens, a purchase order is sent [1]. Because of the nature of the CEP, we will use event as a representation-based definition. In complex event processing, multiple rules are applied to the events that flow through the system. These rules are applied in Event Processing Agents (EPA), which are the fundamental building blocks of CEP. EPAs monitor the patterns in event flows and react according to defined function.They take events as input and produce new events as output according to the given set of rules. Luckham [1] classfies agents as input filters, maps and constraints. Filters are event patterns that remove irrelevant events from the streams. Only relevant events are passed further to maps and constraints. Maps are used to create higher level complex events by aggregating multiple lower level events. These aggregations specify event hierarchies. Constraints can detect the presence or absence of an event or a complex event in a stream. They create notification events, when the constraint is violated (broken). Transformation is an abstract supertype of translate, aggregate, split and compose and never used by such alone. Translation means directly mapping one event to another event. This part was based on [5].



Fig. 1 Complex event processing reference architecture - adapted from [2]

On figure 1 a schema of a complex event architecture is presented. Processing of events is divided into several levels which conform to desired level of inference. At the lowest level, the event preprocessing runs – during this phase we clean the input data stream to produce comprehensible data. On the next level, the events that were detected in input data are refined and subsequently initial decisions and correlations are done. The main challenge is to find relevant data. Then, situation refinement and impact assessment follows. At the level of impact assessment, we may predict the intentions of subject or to estimate potential opportunities or threats. At the end, the process refinement is done. Information was taken from [7].

All the results of event processing and operational visualization at all levels are summed up in a human readable format via user interface.

Most of current CEP platforms solutions fall into one of these two categories:

1. Aggregation-oriented CEP, or

2. Detection-oriented CEP.

The first approach uses real time processing of event data which enters the system. As an example we might take an algorithm, performing some calculations within a moving window of a given size. On the other hand there is a detection-oriented solution which focuses on examination of data and detection of patterns or recurring behavior. Many applications use combination of both approaches.

2.2 Rules for real time processing

In [8] are presented rules for real time processing. As we want to design model for events processing which can react in real time we will design the model with respect to these requirements.

The rules and their brief introduction:

1. **Keep the data moving** - to process messages “in-stream”, without any requirement to store them, to perform any operation or sequence of operations in order to achieve low latency. An additional latency problem arises for systems that are passive, that means the system requires applications to continuously poll for conditions of interest.
2. **Query using SQL on streams (*StreamSQL*, *CQL*)** – a traditional SQL system knows when the computing is finished when it gets to the end of a table, but the streaming data never ends so the stream processing engine must be instructed when to finish such an operation and output an answer. The window concept serves this purpose by defining the “scope” of a multimessage operator such as an aggregate or a join. Depending on the choice of window size and slide parameters, windows can be constructed as isolated or overlapping.
3. **Handle stream imperfections (delayed, missing and out-of-order data)** –we need to have built-in mechanisms to provide adaptability against stream “imperfections”, including missing and out-of-order data, which are commonly present in real-world data streams.
4. **Generate predictable outcomes** **-** a stream processing system must process time-series messages in a predictable manner to ensure that the results of processing are deterministic and repeatable. The ability to produce predictable results is also important from the perspective of fault tolerance and recovery.
5. **Integrate stored and streaming data** **-** requires to have the capability to efficiently store, access, and modify state information, and combine it with live streaming data. For seamless integration (without modifying the application code), the system should use a uniform language when dealing with either type of data.
6. **Guarantee data safety and availability** - to preserve the integrity of mission-critical information and avoid disruptions in real-time processing, a stream processing system must use a high-availability solution. We must ensure that the applications are running and available, and the integrity of the data is maintained at all times, despite of the failures.
7. **Partition and scale applications automatically** – to have the capability to distribute processing across multiple processors and machines to achieve incremental scalability. Stream processing systems should also support multi-threaded operation to take advantage of modern multi-processor (or multicore) computer architectures. Ideally, the distribution should be automatic and transparent.
8. **Process and respond instantaneously** – a stream processing system must have a highly-optimized, minimal-overhead execution engine to deliver real-time response for high-volume applications.

3. High frequency data

One of the ideal source of high frequency data are financial markets. For processing of these data we need to put together statistic, mathematic, economic and also informatic methods and algorithms. Statistical methods can predict time series well but the results are not so stable when there is noise in the time series - such as inaccurate or incomplete data. Market data is highly variable and each time interval (known as tick) generates new logical unit of data. Main focus of today’s research is not only the development of high-quality descriptive systems but also the ability to produce predictions of future movements of data. Information for this section were mainly taken and updated from [2].

In terms of adaptive rules generation, the real-time event processing is a key part we would like to focus on. Fast and automated data analysis won’t yield any advantage if every next step in processing requires human approval. But transforming a business system to react in real-time requires not only new technologies, but a new way of thinking as well.

Most apparent are applications to high frequency financial data, which are characterized by a set of contemporaneously correlated trade marks, many of them are discrete in nature at high or ultra high frequency. In empirical studies on financial market microstructure, characteristics of the multivariate time varying conditional densities (moments, ranges, quantiles, etc.) are crucial. For instance, with our model we are able to derive multivariate conditional volatility or liquidity measures.

4. ADAPTIVE business rules

In CEP, the processing takes place according to user-defined rules, which specify the relations between the observed events and the phenomena to be detected. We would like to focus on event processing from the decision point of view. After the data has been processed by a CEP engine we can distinguish recurring behavior in data. We can describe these phenomena by patterns. Designed model for decision making during processing of data takes into account these patterns and so the system may react to this behavior and may apply on data the most suitable rule with which the process will continue.

For example when the trader has to decide when to buy or sell, we might apply the following rule:

* If price\_of\_security reaches threshold\_value *→* open buy or sell position

Next step of processing will be chosen according to information obtained from historical data and information given by user. User may update the behavior according to preconditions, which will be in system described by business rules. Designed model of system will allow the user to specify his or her own input rules.

Next steps in decision making might also be correlated by learning of model on meaningful sample of data. But this is not the aim of this thesis.

4.1 DECISION MAKING

For decision making of some more complex situation requiring calculation we might use an engine which can communicate with our CEP solution. We might use existing tool which can generate action as an output based on a given set of data (eg. FICO Blaze Advisor) or we can implement our own solution. We still get a result which is set up on the fly without need of redeployment of the running process. Communication with external solution might be provided via web services – this design is used in Service Oriented Architecture (SOA) approach. Service oriented architecture (SOA) models were created to facilitate the design of enterprise software.

SOA addresses the following concerns:

1) many systems need to be integrated to a single interoperable entity and serve as a service for other systems

2) the existing components don’t have to be implemented strictly in the same language in order for them to communicate to each other

3) businesses implement new products rapidly. Another source of integration requirements are mergers, which bring new, incompatible systems to the ecosystem. Desinged model must scale to support high volumes of events [5].

4.2.CEP PATTERNS

As we have historical data we might simulate the designed solution with dynamically set up rules on these data. After the run we will compare the experimental results with real data processing. Pattern detection works always in some context. The context defines the relevant events impacting the pattern matching. It can be temporally or spatially bounded. Context can also be based on semantics of mutually referenced objects or entities. This context is called a window. According to the [5] the patterns are divided into two categories - basic patterns and dimensional patterns. They consist of

* logical operations (conjunction, disjunction, negation)
* threshold patterns - triggers when a number of events of some type has been processed.
* subset selection patterns – can select significant values in a set
* modal patterns - check if some assertion is true.

For example, there are patterns to detect if one instance of each type of the participant set (or none of them) has been seen. These patterns without clearly defined rules can be very ambiguous. Pattern policies allow us to express evaluation, cardinality, repeated type, consumption and order policies. Evaluation policy defines whether we want to evaluate the pattern every time a new event is observed. Cardinality policy determines how many times one event can be part of a pattern matched [3,5].

Traditional approach for detecting anomalies in the stock market has been statistical analysis, after all the trades have been completed. CEP solution has the benefit that analyses according patterns can be run when needed. CEP is suitable for fast reactions for frauds, real-time monitoring is a necessity and can be as a complementary method. For example we may detect a fraud for payment by credit cards according to a spatial restriction – if we encounter two payments from different places with very long distance in short period of time, we can be almost sure about the fraud.

CEP systems are often developed bottom-up by first identifying the event information available. However, in [10] a top-down approach is described. First of all, the key performance indicators and other abstract measures are defined and then we hierarchically proceed down to find the correct low level events in a changing environment to calculate them. CEP distinguishes several scalability attributes:

* Events volume
* Event processing agents
* Producers and consumers
* Window size
* Computational complexity
* Environment
* Constants

For stream analytics it is a key capability that complex event processing systems are able to scale out in order to process all incoming events in a timely fashion as required by the application domain.

4.3 use of business rules patterns

By defining the business rules patterns, we will control the events flow. According to the context, we dynamically update and apply the restrictions on business rules. These restrictions might be defined as in the following example.

For simplicity let‘s consider we have 4 event types occuring during data processing – *A*, *B*, *C*, *D*. Then the set of restrictions could be defined:

1. *A* – event *A* must occur during data processing
2. *NOT A* – event *A* must not occur
3. *AB* – events *A* and *B* must occur during data processing
4. *A + B* – one of the event *A* or *B* must occur
5. *A→B* – event *B* will occur only after the processing of event *A*
6. *AB→C* – event of type *C* will occur only if events *A* and *B* were processed before

Also, CEP distinguishes a number of event processing patterns, for instance:

* adoption patterns
* business patterns
* integration patterns
* workflow patterns, etc.

More detailed information can be found in [9].

The use of CEP patterns might be helpful if we want to generate the rules automatically during the process run.

4.4 Use of workflow patterns for complex event processing

The decision making in nodes might be controlled similarly like we do when using the workflow patterns when controlling business process. We will focus on use of following basic patterns (also known as sequence patterns):

* AND-join pattern
* AND-split pattern
* XOR-join pattern: Simple Merge
* XOR-split pattern: Exclusive Choice

In process-aware information systems various perspectives can be distinguished. The [control-flow perspective](http://www.workflowpatterns.com/patterns/control/index.php) captures aspects related to control-flow dependencies between various tasks (e.g. parallelism, choice, synchronization etc). Originally, twenty patterns were proposed for this perspective, but this number has grown to over forty patterns. The [data perspective](http://www.workflowpatterns.com/patterns/data/index.php) deals with the passing of information, scoping of variables, etc, while the [resource perspective](http://www.workflowpatterns.com/patterns/resource/index.php) deals with resource to task allocation, delegation and others. Finally the patterns for the [exception handling perspective](http://www.workflowpatterns.com/patterns/exception/index.php) deal with the various causes of exceptions and the various actions that need to be taken as a result of exceptions occurring. This information has been adapted from [11].

5. Conclusion and future work

In this article we introduced the idea of processing of big data by using the complex event platform with adaptive business rules for decision making. We mentioned the advantages of its usage and the overview of methods and theoretical basis.

The next step of our research is to formally design and consequently implement model for decision making of processing events by using CEP with adaptive model of business rules. After that we will perform test measurments and compare experimental results with real result. Simulation will be conducted on real historical data on relevant sample of data implementation of system according to adaptive business rules. We will experiment with some scalable attributes such as the size of window, the volume of event and others and compare partial results.

6. Acknowledgement

This work was partially supported by the BUT FIT grant FIT-S-14-2299, "Research and application of advanced methods in ICT".

References

[1] LUCKHAM D., The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems, Addison Wesley Professional, May 2002, ISBN: 0201727897.

[2] STEINBERG A., BOWMAN C., Handbook of Multisensor Data Fusion, JDL Laboratories, CRC Press, 2001.

[3] GENCAY R., DACOROGNA M., MULLER U.A., PICTET O. OLSEN, R., An Introduction to High-Frequency Finance, London, Academic Press, 2001, ISBN 978-0-12-279671-5

[4] ETZION O., NIBLETT P., Event Processing in Action, Manning Publications Co., Greenwich, CT, USA, 2010, 1st edition.

[5] AALTO A., Scalability of Complex Event Processing as a part of a distributed Enterprise Service Bus, master thesis, Aalto University, Esbo, 2012.

[6] What is Complex Event Processing?, 2007, [online, visited June 2014] <<http://www.thecepblog.com/what-is-complex-event-processing/>>

[7] ZÁMEČNÍKOVÁ E., KRESLÍKOVÁ J., Time Series Prediction Based on Event Driven Business Process Management.International Journal of Computational Engineering Research (IJCER), Ghaziabad, Uttar Pradesh: International Journal of Computational Engineering research, 2014, Issue 4, vol. 4, s. 1-6. ISSN 2250-3005.

[8] STONEBRAKER M., [ÇETINTEMEL](http://dl.acm.org/author_page.cfm?id=81316487754&coll=DL&dl=ACM&trk=0&cfid=359397040&cftoken=97296218) U., ZDONIK S., Newsletter, ACM SIGMOD Record, Volume 34 Issue 4, December 2005, Pages 42 - 47, ACM New York, NY, USA, [online, visited June 2014] <<http://cs.brown.edu/~ugur/8rulesSigRec.pdf>>

[9] PASCHKE A.: Design Patterns for Complex Event Processing, 2nd International Conference on Distributed Event-Based Systems (DEBS'08), Rome, Italy, 2008. [online, visited June 2014] <<http://arxiv.org/ftp/arxiv/papers/0806/0806.1100.pdf>>

[10] KELLNER I., FIEGE L, Viewpoints in complex event processing: industrial experience report, In Proceedings of the Third ACM International Conference on Distributed Event-Based Systems, DEBS 2009, pages 9:1 - 9:8, New York, NY, USA, ACM.

[11] AALST W., Workflow Patterns, [online, visited June 2014] <http://www. workflowpatterns.com/>

1. \* Faculty of Information Technology, Brno University of Technology, {izamecni|kreslika}@fit.vutbr.cz [↑](#footnote-ref-1)