Domain Specific Reference Models for Event Patterns – for Faster Developing of Business Activity Monitoring Applications

Rainer v. Ammon[1], Christian Silberbauer[1], Christian Wolff[2]

[1]*Centrum für Informations-Technologie Transfer GmbH, D-93051 Regensburg, Germany, rainer.ammon@citt-online.com, christian.silberbauer@citt-online.de*

[2]*Media Computing, University of Regensburg, D-93040 Regensburg, Germany, christian.wolff@sprachlit.uni-regensburg.de*

Abstract

Business Process Management (BPM) and real-time Business Activity Monitoring (BAM) are newly discussed as the preconditions for a so-called predictive business and the competitiveness of enterprises in the future. Complex event processing (CEP) is an emerging technology that shall be the basis in order to achieve actionable, situational knowledge from distributed message-based systems, databases and applications in real-time or near real-time. Detecting event patterns in an event cloud or in one or more event streams [30] is a basic idea of the CEP technology. If low level events without any semantics occur in specific combinations, a complex event on a higher business level can be derived of them as well as of "historical" events stored in databases. First attempts of setting up CEP applications have shown that the potential adopters have major problems to define the needed event patterns. This is a reason why future CEP applications will delay to be set up. Therefore the availability of domain specific reference models for event patterns is needed. In the project DoReMoPat, a catalogue of reference models for selected domains like automotive, finance, logistics, telco are developed and implemented as customizable prototypes. A meta model is defined for faster developing reference models for other domains. Reference models for event patterns can dramatically reduce time and costs as well as improve the quality of BPM/BAM projects.

1. Introduction

In accordance with a broad understanding (see e.g. [7, 9, 24, 27], Complex Event Processing (CEP) is an emerging technology "…that creates actionable, situational knowledge from distributed message-based systems, databases and applications in real-time or near real-time. CEP can provide an organization with the capability to define, manage and predict events, situations, exceptional conditions, opportunities and threats in complex, heterogeneous networks. CEP will help advance the state-of-the-art in end-to-end visibility for operational situational awareness in many business scenarios. These scenarios range from network management to business optimization, resulting in enhanced situational knowledge, increased business agility, and the ability to more accurately and rapidly sense, detect and respond to business events and situations" [9], as a precondition of the competitiveness in the future.

Detecting event patterns is one of the basic ideas of the CEP technology. If so-called low level events occur in specific combinations, a complex event could be derived of them as well as of "historical" events stored in databases. This process of event hierarching and event abstracting for generating higher business level events is called "Complex Event Processing". All low level events exist in the "event cloud" of an enterprise. In order to define these matching combinations of events, event patterns are used. These patterns shall be monitored in "enterprise cockpits" as defined views in the sense of real-time Business Activity Monitoring (BAM) for a predictive business and a better business insight. BAM on the basis of CEP shall improve the existing, often complained IT blindness which is caused by thousands or millions of low level events per second without any semantics.

First attempts of setting up CEP applications have shown that the potential adopters have major problems to define the needed event patterns. This is a reason why future CEP applications will delay to be set up. Therefore the availability of domain specific reference models for event patterns is needed. In the project DoReMoPat, a catalogue of reference models for selected domains are developed and implemented as customizable prototypes. A meta model is defined for faster developing reference models for other domains.

1.1. The present situation

"Others manage and monitor their processes already, we think about it…" This typical situation was told by HypoVereinsbank of the Unicredit Group at the 4th Ex-

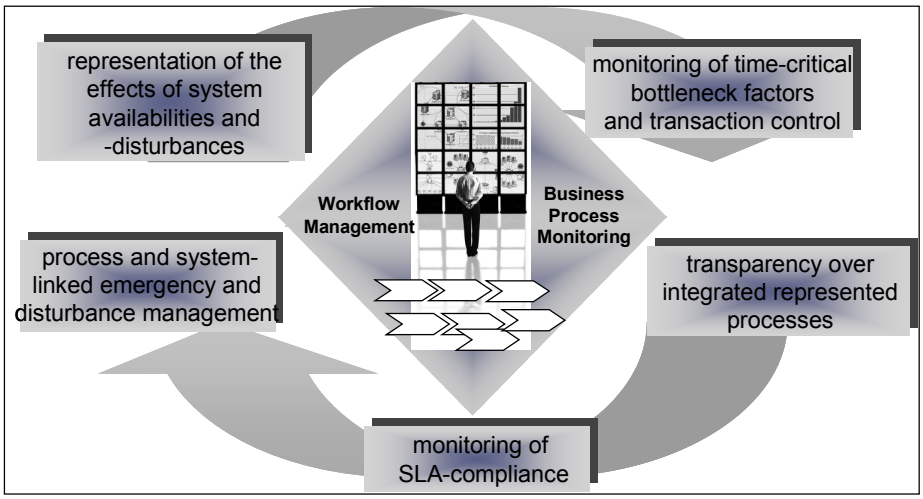


Figure 1. Aspects of an enterprise-wide business process management and monitoring

pert Meeting about "BPM/BAM/CEP/SOA/EDA" in December 2006 [26]. In some industries such as energy or automobile however processes are already for a long time automated managed and monitored. For the control of current supply early warning systems exist, which indicate overloads and in case of losses include emergency systems into the supplying processes in real-time or at least in near-time. Or - as another example from the domain of automotive - a car has already many thousand miles behind itself before it is built. The supply of the individual components takes place „just in time". Bottleneck situations are controlled in real-time or at least in near-time.

But this is only true for production or supply processes. On the other hand business processes are until today only individually managed and monitored - if at all, then usually only as a isolated solution (e.g. for algorithmic trading in the financial domain [10]). So-called enterprise cockpits, which cover all business processes of an organisation, are not available at present yet.

1.2. The future of predictive business

In time and even predictively recognizing of disturbances and the if possible automated and self-healing (re-) action to threatening difficulties will decide in the future on the competitive ability of organizations and of whole national economies [36]. Figure 1 shows the most important aspects of an enterprise-wide business process management and monitoring.

Complex Event Processing (CEP) is a paradigm, which is helpful to react in real-time to changes of states by corresponding information [27]. By CEP messages or data are correlated, aggregated, analyzed and evaluated in real-time. This newly generated information then provides the base for further decisions. Thus, a CEP platform becomes an intelligent BAM tool, which also offers the possibility of dynamic visualization. In the future a lot of applications will be realized with methods of CEP which can already be seen from the respective blogs of the CEP community [5, 8, 15].

2. Important challenges for real-time BAM

Before we can realize real-time BAM, some important challenges have to be mastered. Inter alia it concerns thereby the necessity for redesigning the business process models under SOA considerations as well as the technology challenge of combining BPM and CEP regarding the non-intrusive generation of BAM-relevant events and the challenge of multi-channeling and the BAM problem in the case of different business processes for each channel for the same use case, e.g. "credit application", as is shown in the following.

2.1. Redesign the business processes for SOA and BPM

For the real-time BAM approach it is necessary to redesign the architecture of the systems in the sense of a SOA [12][45]. As a principal difference to technologies like Enterprise Application Integration (EAI) [31] a SOA is based on the business processes (see Figure 2). At each process step another enterprise internal or external process or a service respectively a software component can be called and eventually a change in a database or in a legacy system, e.g. in an ERP system like R/3 or in a CRM system like Siebel, can be caused. Figure 2 shows the example of an online credit system that this architecture could arbitrarily cascade [1].

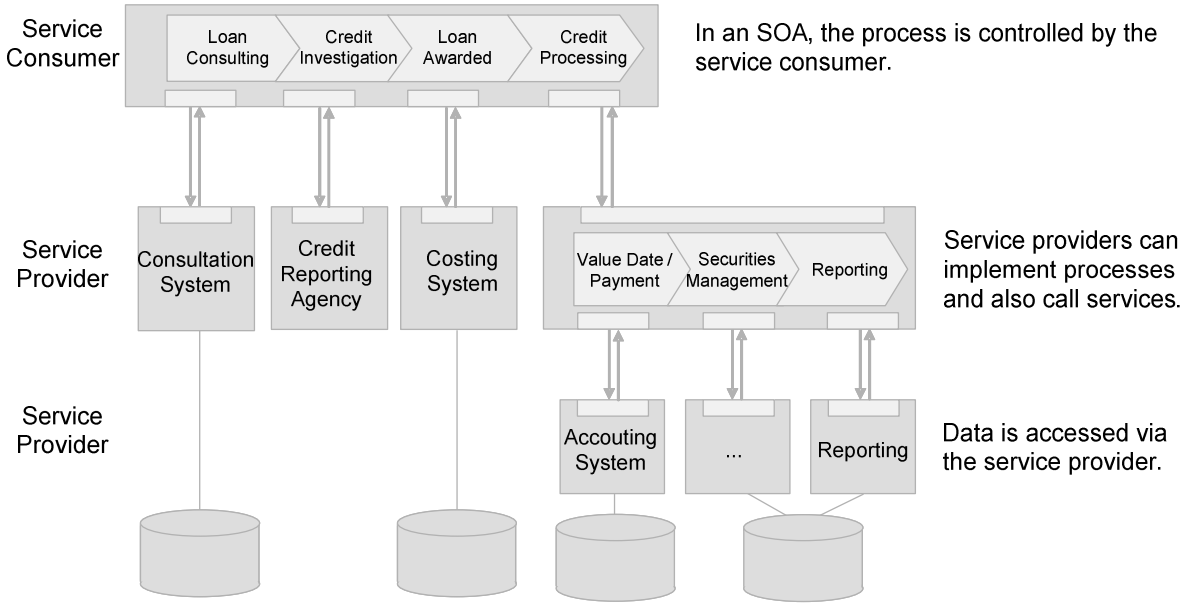


Figure 2. The SOA and BPM challenge - design horizontal and vertical coupling of services [1]

All services are defined, e.g. by the Web Service Definition Language (WSDL), and are bound as web service to a process step. However hereby it has to be guaranteed that for the aim of a real-time BAM no performance problems will be caused by still relatively heavy and slow XML-based protocols like SOAP [40] because of longer latency times.

This also goes for the use of a BPEL- respectively standard based workflow engine [32]. Though the business process can be standardized and flexibly implemented in this way and even be modified at runtime of the system on a high level, i.e. by the means of workflow design tools by non-IT-experts, if applicable even directly by a business department, for the realization of new business and marketing strategies. On the other hand numerous, concurrent business process instances could cause performance and scalability problems of the BPEL engine. This has to be considered in time towards the required real-time performance of the BAM at the dimensioning of the system and the resource planning (sizing project).

2.2. Events for the workflow engine versus events for CEP

The business process is controlled e.g. in the sense of an eEPC notation [19] of events, like "credit application received", "credit application checked for completeness", "credit protection agency information received" and so on. These events are manually caused by men, e.g. by an employee or by actions of the system. These events are needed by the workflow engine for flow controlling. These events, however, are no events, which are filtered from the event cloud or from an event stream by a CEP system and processed, e.g. for the visualization in a BAM. It is the job of a business process modeler to design the process model by interviewing the process owner or the business department as a chain of events and actions in such a way that the process model can be executed by a BPEL-based workflow engine (see Figure 3). With some enterprises in the meantime thousands of business processes are designed and archived by different notations like eEPC, BPMN (Business Process Management Notation [34] or as activity diagrams of the Unified Modeling Language (UML) [44]. But these process models are presently not designed as "executable workflows" and have to be redesigned more fine-grained – also under SOA prospects.

The Business Process Management System (BPMS) [39] generates at executing the business process instances partly autonomously BAM- respectively CEP-relevant information, e.g. timestamps for each single process step, whereby in the BAM throughput times can be monitored and analyzed. Additionally specific events can be generated through appropriate implementation of actions, e.g. for monitoring views "establishing the absolute amount of applications within the single pipeline sections of the credit fabric at defined times", "establishing, which application objects have newly reached the corresponding pipeline section within a defined time interval", or the calculation of a credit sum or of the interest rate. This can be realized in the Java EE environment via JMS as the Publish-/Subscribe-method [42] or as well with CORBA analogically via the "notification service" [35].

Altogether there can be on the network level a very large amount of events of different types in a certain time slot (1 minute, 1 hour, 1 day and so on) – metaphorically as an unordered "event cloud" or transformed as a chronologically ordered "event stream" [30].

The event modeller (see Figure 3) decides, which of these events have to be filtered for which BAM view from one or more streams, if necessary have to be aggregated and correlated as higher business level events (e.g. pipeline progress, wait time monitoring) and in which time slot these events have to be held and stored [28]. According to CEP systems, just entering the market, those event scenarios again shall be generated in the future very quickly and modifiable at any time by means of special high level tools without IT experts.

There are special, often already prefabricated adapters for the filtering of each event type (see Figure 3). Examples are SNMP-, email- and log file-adapters for searching for specified strings. The aggregation and correlation take place according to the event scenarios through the used CEP system by means of their Event Processing Language (EPL). The CEP discipline, just coming up [18], is presently discussing - still controversial -, how an appropriate language or an appropriate user interface for this task shall be designed. At present the first CEP platforms offer SQL-like languages, which, however, are extensively enhanced, and process events - precompiled and "in memory" – in a highly performing way, e.g. the "Event Query Language" (EQL) of the Open Source Platform Esper [20], the "Continous Computational Language" (CCL) of Coral8 [17] or StreamSQL of StreamBase [41]. Some CEP platforms additionally provide a user interface without any programming as far as possible, e.g. APAMA [4], AptSoft [6] or StreamBase, and some of them have a proprietary special non-SQL-like EPL, e.g. Apama, IBM/AMiT [23], BusinessBridge of Systar [43].

In the future an important, new task will be the modeling of appropriate event patterns by the new role of the event modeler according to Figure 3.

2.3. The challenge of multi-channeling: Same process for each channel and real time processing

What is also special for BPM/BAM/CEP is the challenge which is caused by multi-channeling.

This means: we have different channels for the same goal (e.g. selling credits) like internet, online terminals, call centers, branches and so on and each channel must lead to the same business process and must deliver the same result, in real-time.

This sounds evidently, but is not true in our reality at present. A bank told in the year 2007 that they still have different processes for the channels and the replication or reconciliation of the data and processes would take 13 days. This means: we cannot „bam" reasonably and in real-time, if we have not solved the problem of multi-channeling and the processes behind the channels.

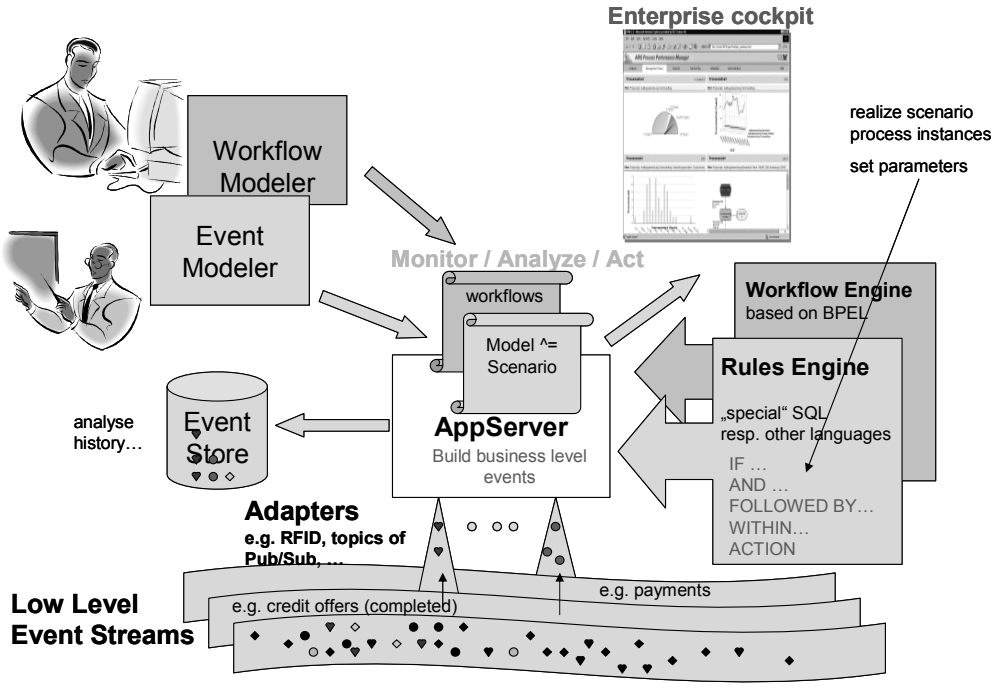


Figure 3. The technology challenge and the principle of BPM/BAM/CEP

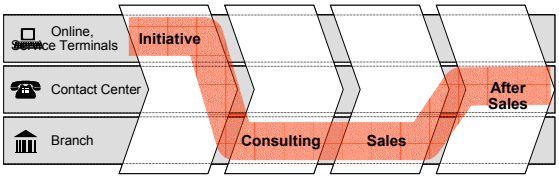


Figure 4. The Multi-channeling Challenge: same process for each channel and real time processing

So, the important steps for monitoring business processes and activities are:

1. Redesign the business processes for SOA and BPM, unify different processes in the case of multi-channeling.
2. Make a SOA, define services, build WSDL-interfaces.
3. Precise description of patterns of events.
4. Detecting patterns in the event cloud.
5. Abstraction of complex event pattern instances to higher level events.

3. The "event tornado": Additional events are needed for a better business insight

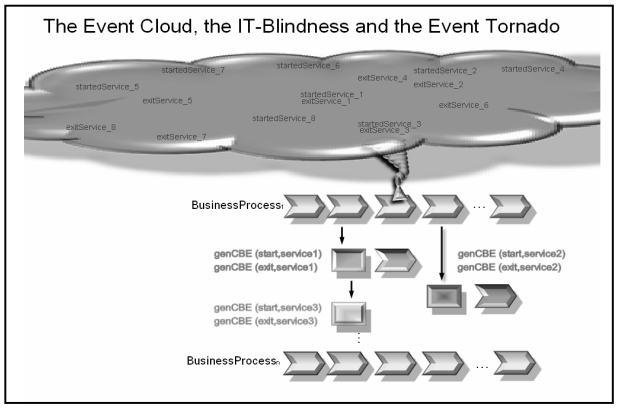


Figure 6. Additional events from business process steps and services for a better business insight and for drill down features

As first experiences showed in the meantime [25], it is not sufficient for the users that a KPI is not satisfyingly fulfilled or the entire type of a business process or a concrete instance of it is currently not well processed. In such cases additionally suitable events must be produced on service level. These will be the basis for drill down features in an enterprise cockpit, by which the cause of a disturbance or a problem can be determined and - proactively – compensating or also self-healing measures can be en-

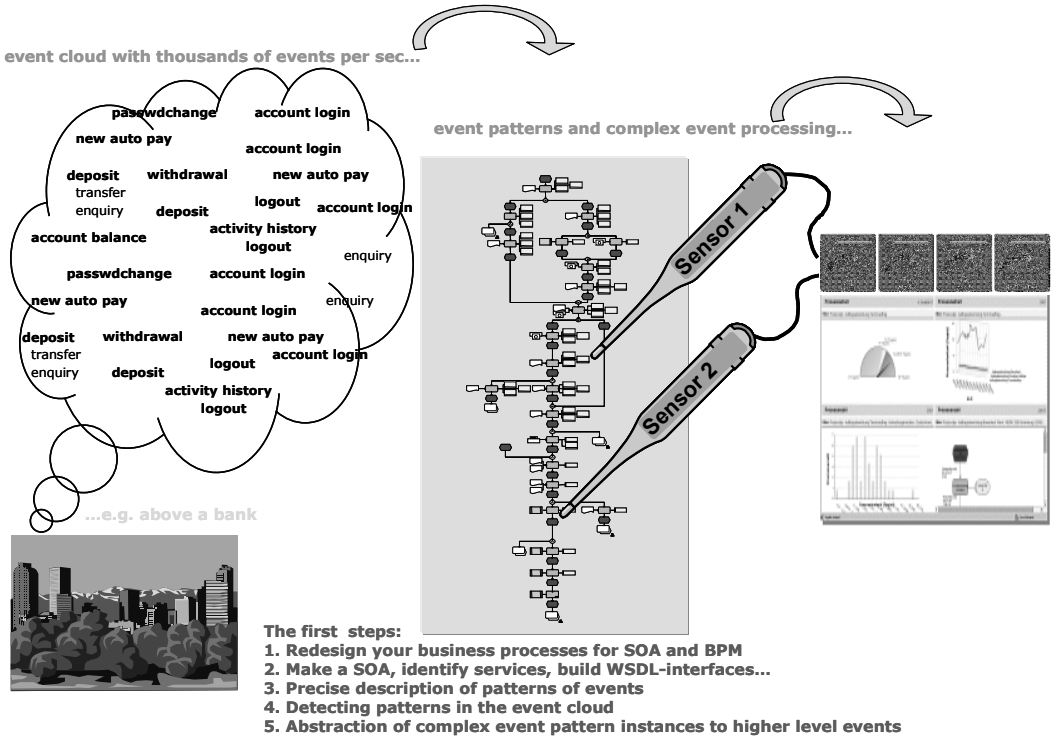


Figure 5. The important steps for the whole picture of monitoring business processes and activities

abled [33]. To this connection the term of the "event tornado" was introduced, which increases the existing event cloud of an organization still substantially by additional events [3]. These events must be produced in the future by the IT-infrastructure and/or a SOA framework or an Enterprise Service Bus (ESB) automatically. Otherwise thousands of additional statements would have to be inserted in the applications intrusively. Such an intrusive procedure would be not passable in practice - often already for legal reasons - and the quality of the implementation of the business logic would beyond that be destroyed. Appropriate non-intrusive approaches are tested since 2006 in the SOA framework of Deutsche Post [25].

4. The solution: Domain specific reference models for event patterns

Therefore also the definition of event patterns for BAM views, which contain suitable drill down features, becomes accordingly complicated and complex. The event patterns must be compiled thereby together with the several roles of an organization. With the management must be specified, what shall be monitored and how indicated or how is to be compensated. With the IT department must be clarified, which event types are needed for it and have to be filtered from the event cloud as well as from data bases with stored "historical" events and how they have to be suitably correlated. Prefab, domain-specific reference models for event patterns substantially would facilitate an introduction of enterprise cockpits and help the dialog between the management and IT departments.

This new profile of an "event modeler" is not or rarely trained at present. This is already predicted by Gartner Group since 2006 as one of the causes of impediment for the broad introduction of CEP applications and/or enterprise cockpits [37]. Therefore we need special courses of studies at our universities. There is a roadmap of the Stanford University regarding the research and the development of CEP technologies, which goes up to the year 2015 [29]. This could be one of the bases for developing such a new course of studies about event processing. At first conferences in 2007 the necessity of special courses of studies are discussed [18].

5. Defining and describing reference models for event patterns

For the representation of reference models for event patterns a standard method is needed. In order to reach a formalization of the reference model, an orientation on the theory of design patterns [14, 22] might be applicable. For the definition of design patterns the authors offer different structures [38]. Table 1 shows that the structures exhibit similarities as well as differences in content wise points and arrangement of the points.

For the description of the reference model the description structure of Buschmann [14] was selected. The structure of Gamma would likewise offer a suitable structure for the description of reference models. For a better understanding of the reference model the particular elements of the description structure of Buschmann are described in table 2.

The definition of the reference model was exemplified in a project for the German TeamBank in 2007 [11]. It was tried to generalize the concrete use case "loss of sales volume because of cancellations of credit applications" as reference model for any web shops, web travel agencies and similar Web applications, which have all the same problem of monitoring cancelled applications or sales and the need to react in real-time respectively to start an appropriate action in order to hold the customers. Therefore project-specific metrics are generalized and general solutions and implementation techniques are pointed out and ordered into the description structure of Buschmann [38]. In the project DoReMoPat reference use cases for different domains like finance, logistics, telco or automotive are

Table 1. Design pattern structure elements of Gamma and Buschmann [12, 19, 36]

Structure elements of Gamma	Structure elements of Buschmann
Name	Name
Intent	Also Known As
Also Known As	Example
Motivation	Context
Applicability	Problem
Structure	Solution
Participants	Structure
Collaborations	Dynamics
Consequences	Implementation
Implementation	Example Resolved
Sample Code	Variants
Known Uses	Known Uses
Related Patterns	Consequences
	See Also

Table 2. Design pattern structure elements of Gamma and Buschmann [36]

Structure element	Description
Name	The name and a short summary of the pattern.
Also Known As	Other names for the pattern, if any are known.
Example	A real-world example demonstrating the existence of the problem and the need for the pattern. Throughout the description we refer to the example to illustrate solution and implementation aspects, where this is necessary or useful.
Context	The situations in which the pattern may apply.
Problem	The problem the pattern addresses, including a discussion of its associated forces.
Solution	The fundamental solution principle underlying the pattern.
Structure	A detailed specification of the structural aspects of the pattern.
Dynamics	Typical scenarios describing the run-time behavior of the pattern.
Implementation	Guidelines for implementing the pattern.
Example Resolved	Discussion of any important aspects for resolving the example that are not yet covered in the solution, structure, dynamics and implementation sections.
Variants	A brief description of variants or specializations of a pattern.
Known Uses	Examples of the use of the pattern, taken from existing systems.
Consequences	The benefits the pattern provides and any potential liabilities.
See Also	References to patterns that solve similar problems and to patterns that help us refine the pattern we are describing.

investigated.

6. Related work

First attempts of setting up CEP applications, with the special focus on BPM and BAM [2], have shown that the potential adopters have major problems to define the needed event patterns. This is a reason why future CEP applications might be delayed to be set up. Therefore the availability of reference models for event patterns is needed.

Reference models are already known from other and similar applications, the most important related field is the research about reference models for business processes [13, 21]. Reference models for business processes have been a successful means for designing, redesigning, tailoring, and implementing business processes. Still there is no common understanding of reference models for business processes:

- What is a reference model?
- What makes them different from a business process model?
- What should be covered by a reference model?
- What is their purpose and how should they be used?
- How should they be designed and presented?

Similar questions have to be answered regarding reference models for event patterns as well. We believe that the orientation at the theory of design patterns is well suitable.

7. Conclusion

Only in order to obtain an impression, how extensive the description of such reference model for event patterns is, one can find an example with [38]. As another example for an impression of dimensions what will be to do: a bank told in July 2007 that they want to monitor in the immediate future circa 40 KPI's with approximately 140 measuring points. At present many potential adopters of CEP/BAM/BPM do not already have an idea of how to monitor how many KPI's in which business processes. Catalogs of domain-specific reference models would dramatically reduce time and costs as well as improve the quality of BPM/BAM projects. Well known scientists and Gartner analysts forecast at present: „The work of IT during the next twenty years will be to complete the evolution of business processes from sequences of slow-moving, disjointed applications to more responsive end-to-end, event-based straight-through flows of action." (Prof. Mani Chandy (Caltech University), Roy Schulte (Gartner) at the Event Processing Summit, Orlando September 2007 [16]).

8. References

[1] R. v.Ammon, W. Pausch, and M. Schimmer, "Realisation of Service-Oriented Architecture (SOA) using Enterprise Portal Plattforms taking the Example of Multi-Channel Sales in Banking Domains", Wirtschaftsinformatik 2005, O. Ferstl et al. (Publ.), Heidelberg, Physica-Verlag 2005, pp. 1503-1518.

[2] R. v. Ammon, H.-M. Brandl, H.-W. Düster et al., "Business Activity Monitoring of norisbank - Taking the Example of the Application easyCredit and the Future Adoption of Complex Event Processing (CEP)", Proceedings of PPPJ'06, Mannheim, 2006, http://www.citt-online.com/downloads/Ammon-Greiner-ACM-PPPJ06-vs5-engl-final.doc, downloaded 2007-08-19

[3] R. v. Ammon, H.-M. Brandl, T. Greiner, D. Guschakowski, "Complex Event Processing in the Context of Business Activity Monitoring: An evaluation of different approaches and tools taking the example of the Next Generation easyCredit.", Preworkshop DEBS'07, H.-A. Jacobsen, Toronto, 2007, http://www.debs.msrg.utoronto.ca/collaborate/Workshop_Agenda, downloaded 2007-08-21

[4] Apama, http://www.progress.com/realtime/products/apama/apama_technology/index.ssp, downloaded 2007-08-19.

[5] Apama's Blog, http://apama.typepad.com, downloaded 2007-08-19

[6] Apama, "Complex Event Processing: A New Computing Model", http://www.eventstreamprocessing.com/index.htm, downloaded 2007-08-21.

[7] AptSoft, http://www.aptsoft.com/, downloaded 2007-08-20.

[8] Tim Bass' Blog http://thecepblog.com, downloaded 2007-08-19/

[9] T. Bass, "What is Complex Event Processing?", http://thecepblog.com/2007/05/14/what-is-complex-event-processing-part-1/, downloaded 2007-08-21

[10] [Bates2007] J. Bates, "Algorithmic Trading - The use of algorithms in automated trading", Dr. Dobbs Journal, March 09, 2007, http://www.ddj.com/hpc-high-performance-computing/197801615, downloaded 2007-08-19

[11] H.-M. Brandl, David Guschakowski, "Complex Event Processing in the context of Business Activity Monitoring - An evaluation of different approaches and tools taking the example of the Next Generation easyCredit", diploma thesis, University of Applied Sciences, Regensburg, May 2007, http://www.citt-online.com/downloads/Diplomarbeit_BaGu_Final.pdf, downloaded 2007-08-19

[12] M. Brandner et al.: Web services-oriented architecture in production in the finance industry", Informatik Spektrum, Volume 27, No. 2, 2004, pp. 136-145.

[13] BPRM05 Workshop on Business Process Reference Models, Nancy, September 5, 2005, http://events.deri.at/bpm2005/, downloaded 2007-08-19

[14] F. Buschmann, R. Meunier, H. Rohnert, P. Sommerlad, *A System of Patterns. Pattern-Oriented Software Architecture*, Wiley & Sons, 1996

[15] CEP-Interest Group, http://tech.groups.yahoo.com/group/CEP-Interest/, downloaded 2007-08-19

[16] M. Chandy, R. Schulte, „What is Event Driven Architecture (EDA) and Why Does it Matter", July 15, 2007, http://complexevents.com/wp-content/uploads/2007/07/EDA%20article%20long%20Chandy%20and%20Schulte%2015%20July%202007%20final.pdf, downloaded 2007-08-21

[17] Coral8, http://www.coral8.com/, downloaded 2006-04-24.

[18] EDA PS workshop 2007, Vienna, September 24, 2007, http://www-static.cc.gatech.edu/projects/disl/conferences/EDAPS/, downloaded 2007-08-19

[19] eEPK – erweiterte Ereignisgesteuerte Prozesskette, http://de.wikipedia.org/wiki/Erweiterte_ereignisgesteuerte_Prozesskette, downloaded 2007-08-20

[20] Esper, http://esper.codehaus.org/, downloaded 2007-08-19.

[21] P. Fettke; P. Loos; J. Zwicker, "Business Process Reference Models - Survey and Classification", *Proceedings of the Workshop on Business Process Reference Models*. E. Kindler; M. Nüttgens (Publishers): Business Process Reference Models (BPRM) - Proceedings of the Workshop on Business Process Reference Models (BPRM 2005), Satellite workshop of the Third International Conference on Business Process Management (BPM), Nancy, France, September 5, 2005, pp. 1-15

[22] E Gamma,.R. Helm, R. E.Johnson, *Design Patterns. Elements of Reusable Object-Oriented Software,* Addison-Wesley Longman, Amsterdam, 1995

[23] IBM/AMiT, http://www.haifa.il.ibm.com/dept/services/soms_ebs.html, download 2007-08-18

[24] IBM, "Complex Event Processing (CEP) – Innovation Matters", http://domino.watson.ibm.com/comm/research.nsf/pages/r.datamgmt.innovation.cep.html, downloaded 2007-08-21.

[25] D. Jobst, G. Preissler,. „Study on Mapping the Clouds of Low Level Events and Business Level Events for an Enterprise Cockpit", Workshop on Java-Based Distributed Systems and Middleware, ACM International Conference on Principles and Practices of Programming in Java (PPPJ 2006), September 2006.

[26] E. Jung, "Anforderungen an BPM, BAM, Service-Repository und Service-Registry aus Sicht der HypoVereinsbank als Mitglied der Unicredit Group", 4th Expert Meeting, Regensburg, Dec. 12-13, 2006, http://www.citt-online.com/index.php?id=veranstaltungen&id3=exp4&id4=more, downloaded 2007-08-20.

[27] D. Luckham, *The power of events*, Addison Wesley , Boston, San Francisco, New York et al., 2002.

[28] D. Luckham, "The Beginnings of IT Insight: Business Activity Monitoring", http://www.ebizq.net/topics/bam/features/4689.html, 2004, downloaded 2007-08-20.

[29] D. Luckham, "Roadmap – The ascent of CEP", slide 30 in the lecture of R. v. Ammon, "Mapping Clouds of SOA- and Business-related Events for an Enterprise Cockpit", SOA Days, Bonn, September 20-21, 2006, http://portal.acm.org/ft_gateway.cfm?id=1168089&type=pd,. downloaded 2007-08-21

[30] For terms like “event”, “event type”, “complex events” see: D. Luckham, R. Schulte, Event Processing Glossary, http://complexevents.com/?cat=15, downloaded 2007-08-20
[31] G. Meinhold, „EAI und SOA: Die Komponenten fallen nicht vom Himmel“, *Objektspektrum*, No. 2, 2004, pp. 33-36.
[32] OASIS: Web Services Business Process Execution Language Version 2.0, April 11, 2007, http://docs.oasis-open.org/wsbpel/2.0/serviceref, downloaded 2007-08-20.
[33] OASIS: Web Service Distributed Management. WSDM Introduction. 2006. http://www.oasis-open.org/committees/download.php/16998/wsdm-1.0-intro-primer-cd-01.doc. downloaded 2007-03-02
[34] OMG, Business Process Modeling Notation (BPMN), http://www.bpmn.org/, downloaded 2007-08-20.
[35] OMG, Notification Service, http://www.omg.org/technology/documents/formal/notification_service.htm, downloaded 2007-08-20
[36] V. Ranadivé, *The power of pr predict*, McGraw-Hill, New York, Chicago, San Francisco et al., 2006.
[37] R. Schulte, http://www.computerwoche.de/bizone/579005/index4.html, downloaded 2007-08-20.
[38] Ch. Silberbauer, R. v. Ammon, H.-M. Brandl, D. Guschakowski et al., „Loss-Pattern“, 5th Expert Meeting, Regensburg, June 25-26, 2007, http://www.citt-online.com/downloads/exp5/losspattern.doc, downloaded 2007-08-20.
[39] SixSigma, “BPMS”, http://www.isixsigma.com/dictionary/BPMS-536.htm, downloaded 2007-08-20.
[40] SOAP, http://de.wikipedia.org/wiki/SOAP, downloaded 2007-08-20.
[41] StreamBase, http://streamsql.org/pages/documentation.html, downloaded 2007-08-20
[42] Sun Developer Network (SDN), http://java.sun.com/products/jms/, downloaded 2007-08-20
[43] Systar, http://www.systar.com/main/main.asp, downloaded 2007-08-20
[44] Unified Modeling Language 2.0 (UML), http://www.uml.org/#Links-General, downloaded 2007-08-20
[45] D. Woods, *Enterprise Services Architecture*, O’Reilly: Gravenstein, 2003.

Appendix

Proposal for a Reference Model for Event Patterns
Taking the Example "Loss Pattern"

Final paper of the conjoint course "Complex Event Processing"
University of Regensburg / University of Applied Sciences of Regensburg
http://www.citt-online.com/index.php?id=lehre&id3=course_eventprocessing&id4=more

Presented by Christian Silberbauer at the 5th Expert Meeting,
June 25-26, 2007, Regensburg/Germany
http://www.citt-online.com/index.php?id=veranstaltungen&id3=exp5&id4=more

Loss

The Loss pattern enables detecting cancellations of an online order process. The corresponding online platform has to fire events after each significant user activity. A loss occurs when those events are not invoked after a certain timeout. The losses are aggregated according to number and amount. Additionally they are grouped by the location where they have been recognized. The Loss pattern allows determining weak points in the order process, detecting hardware problems, and to react on concrete losses in real-time.

Authors

Christian Silberbauer, Rainer v. Ammon, Hans-Martin Brandl, David Guschkowski, Torsten Greiner, Rolf Kintscher et. al.

Example

A credit application is a pretty extensive order process. A customer has to reveal numerous personal data about his financial circumstances in order to decide whether a bank credit can be granted and under which conditions. This process will not always be completed successfully which will lead to the loss of a potential credit contract. For the operators of such a platform it would be interesting to detect these losses and figure out the causes as accurate as possible. Those reasons could be, e.g.:

- The customer considers the credit application as too long-winded.
- The customer decides spontaneously that he does not want to have a credit anymore.
- The customer does not have all required data available, e.g. about rental income, cancels the application and intents to restart the application later.
- The customer does not have enough time to complete the application and wants to restart it later.
- The customer first wants to discuss with his spouse whether the credit is really necessary.
- The system is overloaded.
- The system fails and throws an exception which forces the process to abort.
- A server- or client-side hardware problem occurs.

Of course, it is not possible to recognize each detail of an abort, especially when the reason is founded in the psyche of the customer. However it would be helpful to determine where in the order process the cancellation happened. Thus, e.g. design errors of the order process or programming errors can be located.

Context

Customers use web applications to order products. The order process encompasses several web pages. The actual order is released at the end of this process. During this process there can be various reasons to cancel.

Problem

The operator of an internet sales platform is interested in determining cancellations of the order process and to obtain further information. The following questions are in particularly relevant:

- Where did the order cancellation occur?
- How often was it canceled?
- What is the economic dimension that the losses caused?

Solution

The web application fires events to the CEP platform for all relevant user actions. These events contain the following information:

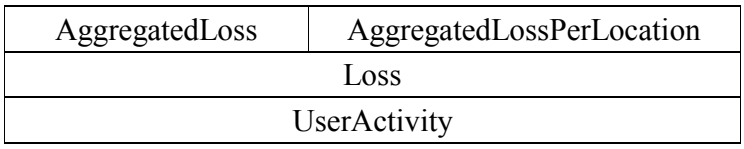
- Key identifying an order
- Location in the program where the events were fired
- Current order amount

If no further events are fired during a defined time slot (*activityTimeout*), a "loss" is determined.

The losses are aggregated over a particular time slot (*observationPeriod*), regarding number and amount. Additionally both will be grouped by their locations.

Structure

Different types of events are necessary for the solution. In the following these are ordered according to their abstraction level:



On the lowest abstraction level there is the event type *UserActivity*. Corresponding events are fired at user activities during the order process. They include:

- *sessionId*: key identifying the order process,
- *location*: location in the program where the event was fired and
- *amount*: current amount of the order.

The event type *Loss* is based on *UserActivity*. It defines the actual loss of an order and contains the following attributes:

- *sessionId*: the key of the order process,
- *location*: location of the last fired *UserActivity* event and
- *amount*: the current amount of the order.

A BAM view receives an aggregated prospect of the losses in two different ways:

The event type *AggregatedLoss* includes for a defined observation period (*observationPeriod*):

- *number*: the number of all losses and
- *amount*: the total amount of orders which were lost.

Both values are additionally grouped by their locations. Thus the so-called ***AggregatedLossPerLocation*** has this attributes:

- *location*: location the losses occurred,
- *number*: number of losses per location and
- *amount*: total amount of the orders which were lost per location

Dynamics

Each significant user activity, particularly click events and user inputs, invokes a *UserActivity* event.

Once for a certain time slot *(activityTimeout)* no more *UserActivity* events have been invoked, a *Loss* event is fired.

The time slot, while an order process is active and a loss can be determined, starts with the first *UserActivity* event of an order, i.e. the first time a certain *sessionId* appears. The time slot ends with the *UserActivity* event of a predefined *location*, namely the so-called *lastLocation.* This is the last *UserActivity* event of an order. It has to be sent as soon as an order is completed successfully.

Each loss event causes a new *AggregatedLoss* event and also a new *AggregatedLossPerLocation* event, i.e. the aggregated values will be refreshed on every single update immediately.

Implementation

In order to implement the pattern the following steps have to be proceeded:

- *Fire an event of the type UserActivity on each essential user activity in the program.* The invocation of the events either can be realized directly in the existing code or can occur non-intrusive, e.g. by interceptors in Java or by other aspect-oriented approaches depending on the programming language.

- *Normalize the location's value.* The type of *location* is string. It indicates where a *UserActivity* event was fired, e.g. in which website, in which method/function, in which class or on which

server. The composition of the *location* is specific to the context. It is recommended to build the *location* in terms of an URI.

- *Define the parameter lastLocation.* It reflects *where* the order is completed successfully. *lastLocation* has to be unique and exists typically once; there are no different last locations.

- *Define the parameter activityTimeout.* Choose this time slot large enough, so that the next *UserActivity* event occurs before the timeout happens. Otherwise a "loss" would be propagated. So the *activityTimeout is* the maximum time interval between two *userActivity* events.

- *Define the observationPeriod*. It specifies the time slot, on which the aggregated loss events refer to. Typical values for this parameter are a day, but also an hour, depending on how often "losses" usually occur, how strong "losses" fluctuate and how fast you like to react on changes.

- *Realize the different complex event types Loss, AggregatedLoss and AggregatedLossPerLocation in the CEP platform.*

Example Resolved

This solution shows how the single complex event types are defined using the Coral8 syntax. They are based on *UserActivity* events. They are fired directly by the web application.

First, the Loss-Event:

```
INSERT INTO Loss

SELECT   Old.sessionId              AS sessionId,
         Old.location               AS location,
         Old.amount                 AS amount

FROM     UserActivity AS Old, UserActivity AS New

MATCHING [$activityTimeout: Old, !New] ON Old.sessionId =
New.sessionId

WHERE    Old.location != $lastLocation;
```

The MATCHING clause realizes the timeout. If no new *UserActivity* event of the same session occurs over the time slot *activityTimeout*, a *Loss* event will be fired. Once an event arrives with the location corresponding to the *lastLocation* of an order, this order is finished successfully.

The implementation of *AggregatedLoss* and *AggregatedLossPerLocation* are quite simple. Their data is relevant to the time slot that is set by the parameter *observationPeriod*.

```
INSERT INTO AggregatedLoss

SELECT   COUNT(sessionId)            AS number,
         SUM(amount)                 AS amount

FROM     Loss KEEP $observationPeriod;



INSERT INTO AggregatedLossPerLocation

SELECT   location                     AS location,
         COUNT(sessionId)             AS number,
         SUM(amount)                  AS amount

FROM     Loss KEEP $observationPeriod

GROUP BY location;
```

Variants

Difference amounts instead of absolute amounts

Sometimes it is not possible or inefficient determining the current total amount if it consists of several single amounts in order to fire a *UserActivity* event, e.g. by Interceptors. Especially, this fact is related to *UserActivity* events, which do not designate a change of the amount. Hence it is recommendable to paste the difference amount into the event instead of the absolute amount. This more primitive event type is called *UserActivityBasic* event. The CEP-Platform derives the actual *UserActivity* event from these *UserActivityBasic* events.

The following listing shows how to create the *UserActivity* event out of a *UserActivityBasic* event according to the Coral8-Syntax:

```
INSERT INTO UserActivity

SELECT   UserActivityBasic.sessionId              AS sessionId,
         UserActivityBasic.location               AS location
         COALESCE(UserActivity.amount, 0)
         + UserActivityBasic.amount               AS amount,

FROM     UserActivity KEEP 1 ROW PER sessionId KEEP $sessionTimeout,
         UserActivityBasic;
```

A *UserActivity* event is based on a *UserActivityBasic* event, which contains the difference amount, and the previous *UserActivity* event of the order, containing the current total amount. The new total amount is determined by the sum of these two values.

In particular you should consider the following aspects while implementing:

- The first event of an order has no precedent event. Thus the current total amount is 0. In the listing above the COALESCE function is used for this case.
- And: There is no need to keep the last *UserActivity* event in memory forever, but only for the time slot the order process is still active. Therefore the so-called *sessionTimeout* is introduced. It specifies the maximum time the last *UserActivity* event is retained.

Different activity timeouts

Instead of only one *activityTimeout* you can define different timeouts depending on the step of the order. This considers the aspect that not each step of an order takes the same length of time. A global timeout has to be geared to the most time-consuming step of the order whereas the individual timeouts can be parameterized to the actual duration of the step of an order. Hence a "Loss" can be detected faster. This might be necessary if a reaction to a Loss should occur in real-time.

Exceptions initiate loss events directly

If an application throws an exception and leads to the cancellation of the session, a Loss will be detected after the timeout as described. The exception leads to the cancellation of the order process, the user cannot proceed, the *activityTimeout* occurs and thereupon a *Loss* event will be fired. Alternatively Exceptions could fire *Loss* events immediately. This would have two benefits:

- The Loss can be recognized faster, because it happens immediately and not after the timeout.
- The event attribute *location* can be specified more accurate. Instead of using the location of the precedent *UserActivity* event, the exception itself can be used to specify the location.

Ignore Losses with small amounts

Maybe you do not want to take into account losses of orders having only small amounts or even those which have no amount at all. In that case define a parameter *amountLimit* and consider this parameter by specifying a *Loss*. In Coral8 you muss add the following WHERE clause to the implementation of the event type *Loss*:

```
WHERE    amount >= $amountLimit
```

Additional Aggregations of the Loss event

The two composed Loss events named *AggregatedLoss* and *AggregatedLossPerLocation* are not the only useful aggregations. Here is a further example:

- *AggregatedLossPerAmount*: Determine the Number of losses grouped by ranges of amounts.

Known Uses

The *Teambank* uses the Loss pattern in its *Next Generation easyCredit* platform. This is an online platform for simple and fast granting of bank credits. The Loss pattern is quite useful to ensure high availability, to detect aborts of high number or amount quickly and to allow proceeding appropriate reactions immediately.

Consequences

The Loss pattern provides fundamental benefits:

It is possible to analyze and to correct weak points of an order process. If losses appear frequently at a specific location it leads to the assumption that the user might be deterred of entering the data. It might be that the input is too extensive or too long-winded. Consequently, the operator can remove – if possible – or at least reduce the corresponding user inputs or he can move them to a later point in the process where the user is less likely to abort.

Appropriate measurements against a concrete Loss can be triggered. Immediately after a certain Loss corresponding arrangements can be initiated, e.g. if applicable the operator can personally get in touch with the user or can apply appropriate advertising.

Bugs can be detected. Bugs in a program that lead to the cancellation of an order process are also leading inevitably to a loss. Not only exceptions can be detected, that usually are written in log files anyway, but also e.g. infinite loops will be recognized as losses.

See also
