A Blueprint for Event-Driven Business Activity Management

Christian Janiesch[1], Martin Matzner[2], and Oliver Müller[3]

[1] Institute of Applied Informatics and Formal Description Methods (AIFB),
Karlsruhe Institute of Technology (KIT), Englerstr. 11, Geb 11.40, Karlsruhe, Germany
christian.janiesch@kit.edu
[2] University of Münster, European Research Center for Information Systems (ERCIS),
Leonardo-Campus 3, 48149 Münster, Germany
martin.matzner@ercis.uni-muenster.de
[3] University of Liechtenstein, Hilti Chair of Business Process Management,
Fürst-Franz-Josef-Strasse 21, 9490 Vaduz, Liechtenstein
oliver.mueller@uni.li

Abstract. Timely insight into a company's business processes is of great importance for operational efficiency. However, still today companies struggle with the inflexibility of monitoring solutions and reacting to process information on time. We review the current state of the art of business process management and analytics and put it in relation to complex event processing to explore process data. Following the tri-partition in complex event processing of event producer, processor, and consumer, we develop an architecture for event-driven business activity management which is capable of delivering blueprints for flexible business activity monitoring as well as closed loop action to manage the full circle of automated insight to action. We close with a discussion of future research directions.

Keywords: Reference architecture, complex event processing, business process management, business process analytics, business activity management.

1 Introduction

Processes represent the core of an enterprise's value creation. Thus, it is intuitively comprehensible that their performance makes a – if not the – major contribution to an enterprise's success. Nowadays business process management (BPM) is not as simple as managing processes contained within one company but within a network of suppliers and customers [1]. This entails a multiplicity of companies with disparate locations and information technology. While process interoperability is obviously one major issue in this scenario [2], we want to focus on what is commonly perceived as a second step when realizing such a system: business process analytics that can provide automated insight to action.

By law, enterprise systems have to provide some support to log their transactions. Consequently, most process engines store data about the enactment of processes in audit logs. Single log entries indicate e.g. the start of a process, the completion of a

S. Rinderle-Ma, F. Toumani, and K. Wolf (Eds.): BPM 2011, LNCS 6896, pp. 17–28, 2011.

task or the invocation of an application [3]. This entails that a large amount of data is already available which can be used to gain insight into processes. Business activity monitoring (BAM) strives for providing real-time insights into running process instances in order to reduce the latency between the occurrence of critical audit events and decision making on effective responds [4, 5]. However, the common architecture of BAM systems does not cover communication with the process engine but only observe execution. Also BAM architectures usually require hardwiring the two systems to each other which makes it difficult to extend the architecture to a network of process engines.

We propose a more flexible approach of loosely coupling systems through business events. Business events represent state changes in process execution and can not only be monitored but managed by complex event processing (CEP) engines [6]. Hence, with an event-driven architecture, it is not only possible to observe process information and display it in (near) real-time but also to take immediate action as a result of the analysis which does exceed simple alerting capability.

We propose an architecture based on the ideas of BPM and CEP which is scalable and flexible so that it can also be used in a networked scenario. The architecture provides a starting point for the design of *business activity management* solutions as opposed to business activity monitoring. It can cover more elaborate scenarios such as context-aware BPM [7].

The structure of the paper is as follows: In the subsequent section, we do a thorough review of related work in BPM and include a wide array of approaches to business process analytics ranging from controlling and monitoring to predictive methods. We include CEP fundamentals in this discussion. We detail our reference architecture in Section 3 and discuss interaction options between BPM and CEP systems. The architecture enables real-time business activity management rather than mere observation of processes. We close with an outlook to future research opportunities.

2 Related Work

2.1 Processes and Activities

Processes are generally seen as activities performed within or across companies or organizations [8]. Activities are considered to be the smallest entity used to build a process. These entities may be seen as complex chunks, which do not need to be analyzed further, or atomic activities, which cannot be divided any further. Furthermore, an activity can either refer to work, which needs to be done manually, or to work, which can be done automatically. Other definitions of processes [9, 10] emphasize on the timely and local order of activities and also regard different kinds of inputs and value-added outputs.

Business Process Management (BPM) refers to a collection of tools and methods for achieving an understanding and then for managing and improving an enterprises' process portfolio [11]. It plans, controls, and monitors intra- and inter-organizational processes with regards to existing operational sequences and structures in a consistent, continuous, and iterative way of process improvement [12]. A BPM system or engine allows for the definition, execution, and logging of business processes.

2.2 Business Process Analytics

Business process analysis is a means to fathom organizational behavior within BPM. From a process perspective, it investigates if the actual behavior of single process instances is compliant with the prescribed business process design. Further, business process analysis might reveal if resources are well utilized within processes, e.g. if work load is assigned purposefully among resources. Finally – taking an object-perspective – business process analysis helps to explore the life cycle of specific business objects and e.g. helps to find out if orders or enquiries have been processed in time and at appropriate costs [13].

Today, enterprise architectures provide various and rich sources for such information, e.g. workflow management systems, enterprise resource planning systems, customer relationship management systems, and product data management systems [14]. The emerging paradigm of context-aware BPM suggests that even more sources of information relevant to process execution [7] may be available in the future via the Web.

Single process instances can be analyzed through behavior data that informs on the state-of-operation of processes' constituting activities. Generally, this means that processes are analyzed with focus on the behavior of completed processes, currently running process instances, or predicting the behavior of process instances in the future [3]. Each of these perspectives corresponds to a certain set of methods that is yet to be established in practice [3, 13, 15]:

- *Process Controlling* is concerned with the ex-post analysis of completed business processes.
- *Business Activity Monitoring* strives to support the real-time monitoring of currently active business processes' behavioral properties.
- *Process Intelligence* is the post-execution forecast of the organization's future behavior through scenario planning or simulation.

There are two design options [13]. A top-down approach starts with conceptual models to reason about BPM activities. The bottom-up concept utilizes the limited analysis functionality of present BPM solutions specifically by integrating bypassed data warehouse concepts [16].

Process Controlling

(Historical) process information can be found in log files and event streams of workflow systems and where other transactional data of information systems is captured [3]. Information is then stored and processed in data warehouses to allow for ex-post analysis of this data. Data warehouses store event-related data permanently, which might be subject to user-specified analytical roll-ups later on [5]. Thus, BI typically looks at process data on an aggregated level from an external perspective [14].

Process mining has a conceptual position as it strives for the "automatic construction of models explaining the behavior observed in the event log" [17]. Its focus is on concurrent processes rather than on static or mainly sequential structures

[14]. To its very nature, process mining is an ex-post analysis of organizational behavior. Process mining informs real-time business process analytics as it poses comparable questions to the log, namely pointing to the "process perspective (How?), the organizational perspective (Who?), and the case perspective (What?)" [17] of a business process. Thus, it rather has the goal of "looking inside the process" [14].

Typical methods of data mining are association analysis to discover significant associations between attributes of (process) objects, cluster analysis to build heterogeneous groups of homogenous (process) objects, and classification to assign (process) objects to groups according to their attributes [18].

Business Activity Monitoring

Business Activity Monitoring (BAM) systems, in contrast, focus on capturing and processing events with minimum latency [5] and therefore have to be tightly integrated with operational data sources. BAM foremost strives for the monitoring of key operational business events for changes or trends indicating opportunities or problems, and enabling business managers then to take corrective actions [5]. Single events recorded are usually not very meaningful in a BAM context, "instead many events have to be summarized to yield so-called key performance indicators" (KPI) [19]. BAM systems are then typically engaged with updating KPI [20].

Therefore a BAM system must be able to detect events occurring in enterprise systems which are relevant to the activity monitored, calculate information for temporal processing, integrate event and contextual business information on the fly, delivering quality data, execute business rules to set thresholds according to key performance indicators and other business-specific triggers, and provide intuitive interfaces for presenting rules and metrics [4, 5].

In contrast, functionality offered by BAM tools in practice is limited to rather simple performance indicators such as flow time and utilization [17]. In parts, the BAM concept goes beyond tracking KPI. Instead, software systems contain some event processing functionality, namely "filtering, transformation, and most importantly aggregation of events" [15]. By applying rules engines to KPI, BAM systems can generate alerts and actions which inform managers of critical situations to possibly alter the behavior of the running process [3]. "This can be done in batch mode […] or in online mode so that the current value of the KPI can be continually tracked, typically on a dashboard" [15].

Process Intelligence

Process Intelligence refers to the predictive analysis of process design to influence future process instances' behavior. This also includes the detection of patterns and links within processes. One can distinguish the process intelligence methods of simulation and optimization as well as – again – data mining [3].

Simulations are commonly used either before a process is deployed for production or after a process has been executed for a reasonable amount of time. In the former case, simulation can assist in assessing the adequacy of the new process for the designated task. In the later case, variants of the process model can be used for simulation in order to improve or optimize the process design. Simulation usually addresses resource allocation, process structure, and process execution context.

Simulation of process instances with methods such as Monte Carlo offers a broader range of possible process simulation results than deterministic what-if scenarios which focus on manually chosen best case, worst case, and common case scenarios. Trend estimation and forecasting, on the other hand, offer insight into the longer term horizon of process execution and can help identify future bottlenecks in a process or in the process landscape [21]. Simulation commonly uses log data batches for processing.

2.3 Complex Event Processing

Complex event processing (CEP), in general, comprises a set of techniques for making sense of events in a system by deriving higher-level knowledge from lower-level system events in a timely and online fashion [15, 19, 22].

The central concept in CEP is the event. In everyday usage, an event is something that happens. In CEP, an event is defined as a data object that is a record of an activity that has happened, or is thought of happening in a system [22]. The event signifies the activity.

In the context of BPM, events represent "state changes of objects within the context of a business process" [3]. Such objects comprise, among others, activities, actors, data elements, information systems, or entire processes operated in a BPM system. Typical events in the context of BPM are, for example, process started, work item assigned, or activity completed [for more example cf. 23]. Events comprise of several data components called attributes [22]. Attributes of an event can include, for example, the ID of the signified activity, start and end time of the activity, the system in which the activity has been executed, or the user who carried out the activity.

Another key idea of CEP is the aggregation of events. Through the definition of abstraction relationships it is possible to define complex events by aggregating a set of low-level events. Hence, a complex event signifies a complex activity which consists of all the activities that the aggregation of individual events signified [22]. The complex event sub-process completed, for example, is created when the completion of all activities of the sub-process has been signified by the corresponding low-level activity completed events.

Orthogonal to the concept of aggregation is the concept of causality. Also, low-level events might happen in dependence of each other [22]. This dependence relationship is called causality. An event is caused by another event if the former happened only because the latter event happened before. In a business process the event order declined might be caused by the event credit card check failed. Causality plays an important role in any kind of root cause analysis, either online or ex-post.

In doing so and extending in particular the BAM concept to more valuable, actionable business level information, CEP is a good candidate [24], as BPM engines generate events while executing business processes and recording them to logs [13].

Today's BI approaches to process analysis differ from event processing systems in that they are request-driven [15]. It is an ex-post analysis which can basically be distinguished into a *based on* and *not based on* formal representation of the business processes [3].

3 A Blueprint for Event-Driven Business Activity Management

3.1 Interaction Points of BPM and CEP

Only recently, so called event-based or event-driven BI approaches emerged, involving variants of business process analytics which can run online and be triggered by events. These systems are referred to as operational BI by some vendors [15] and by allowing for event-driven decision making, they overcome former shortcomings of BI approaches to event handling. However, event-driven BI falls short when it comes to the dynamic modeling and use of business rules to define actions upon event occurrences and especially in correlating events into metrics over time [5].

Current trends require BAM to extend processing functionality and call for a tighter alignment of BPM and CEP in business process analytics: There is (1) "the drive to provide richer types of observation which necessitates more event processing functions such as event pattern detection" and (2) "the need to provide more online observations when data comes from multiple event sources" [15]. However, even when this information is retrieved, "business processes remain static and cannot be changed dynamically to adapt to the actual scenario" [6] and therefore require guidance for translating insight into action.

BPM seeks to support the entire life cycle of business processes and therefore requires an integration of design and execution phase. Accordingly, business process analytics "undoubtedly involves post-execution analysis and reengineering of process models" [13]. Such analysis requires an exploration of "time, cause, and belonging relationships" [6] among the various activities embedded into a BPM as well as the environment the process is executed in.

Here, CEP emerges to be a valuable instrument for a more timely analysis of process-related events on a business-level perspective. As an independent system, CEP is "a parallel running platform that defines, analyses, processes events" [25]. BPM and CEP might interact via events. The interaction between BPM and event processing then features two facets as BPM can act as (a) a producer and as (b) a consumer of events [15].

There are basically two challenges [13] in such tightly integration of BPM and CEP that are not addressed by practice yet: First, how to move from low-level monitoring and representation of rather technical events reported by various enterprise systems to higher-level business information. Second, business insights claim for instant actions in terms of round tripping to the BPM system for either human or automated response that is supported by a tightly aligned BPM-CEP concept.

This means that BPM systems generate events which signify state changes within the process of a BPM system. Events are then analyzed by the event processing system and the resulting events are either returned to the BPM system or sent to other systems.

BPM systems can then react to situations detected by the event processing system after analyzing events, e.g. an event can trigger a new instance of a business process or it could affect a decision point within the flow of a business process that is already running. Also, it could cause an existing process instance to suspend or stop executing altogether.

The state-of-the-art of business activity monitoring as introduced above primarily deals with the observation of events and the dissemination of results in terms of diagram/ dashboard updates, as well as alerting. However, automated insight to action is out of scope for these systems. Thus, we extend upon the real-time concept of business activity monitoring by leveraging the capabilities of CEP to include decision-making and feedback mechanisms to the originating BPM engine or other systems. This effectively means that the CEP engine, in parts, manages rather than observes the execution of the business process. Hence, we deem it more appropriate to speak of *business activity management* than of monitoring when describing our architecture in the following Section.

3.2 Event-Driven Business Activity Management

In the following, we describe how CEP can be applied to monitor and control business processes in real-time. Fig. 1 depicts the proposed blueprint for an event-driven business activity management architecture using the TAM notation. The architecture is based on the usual tri-partition in CEP. In most CEP applications, there is a distinction among the entities that introduce events into the actual CEP system (event producers), the entities that process the events (event processors), and the entities that receive the events after processing (event consumers).

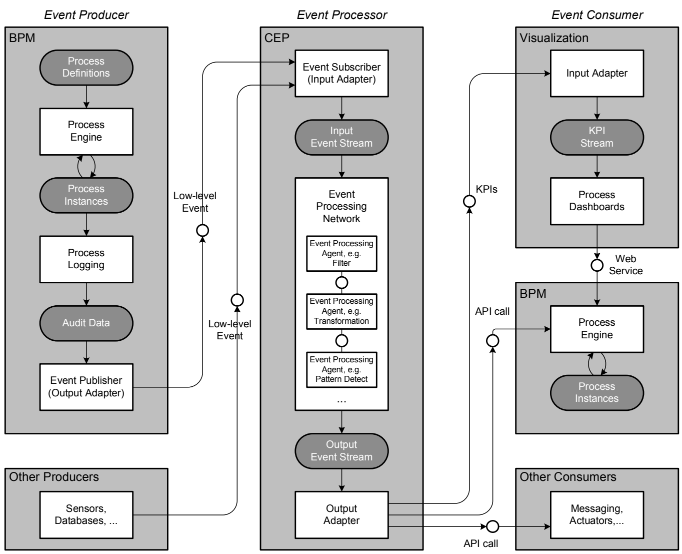

Fig. 1. Blueprint for Event-driven Business Activity Management

Event Producer

The major event producers in our scenario are BPM systems themselves. For analytic purposes, two components of a BPM system are of main interest: The process engine provides an environment for the execution of process instances, based on existing process definitions. The process monitoring component offers functions to track the progress of single process instances and to log corresponding audit data including data on occurred state transitions of process-relevant objects (e.g. processes, activities, resources, work items).

This audit data constitutes a rich source of low-level process events for subsequent analysis, but is usually not accessible from outside the BPM system. Hence, a third component – an event publisher – is required to make the data available for external systems. The main functions of this output adapter is sniffing, formatting, and sending of events [22].

In the sniffing step, the adapter detects events in the process audit data and other relevant sources (e.g. work item lists). Typically, the sniffer maintains a list of filters that defines which events should be observed and subsequently extracted to prevent that sensible data from being published or simply to reduce the volume of events. Such filters can be defined on type level (e.g. only events signifying that an activity has been completed shall be extracted; events of type running and cancelled shall be filtered out) or instance level (e.g. only events with a process definition ID from 4711 to 4742 shall be extracted). It is important that event sniffing must be benign, i.e. it must not change the target system's behavior.

After detecting an event, the corresponding data has to be extracted and transformed to a specific event format. Most BPM systems use proprietary formats for event representation in their log. Recently, there have been attempts to standardize event formats, e.g. the XML-based Business Process Analytics Format (BPAF) [23].

After an event has been detected and formatted, a corresponding message has to be sent to the CEP system. Communication can be done in a direct point-to-point fashion or via some kind of message-oriented middleware (for reasons of complexity not depicted in Figure 1). Either way, the use of a publish/ subscribe messaging mechanism is a promising way to realize a loosely coupled and scalable communication that allows for pushing events to processors.

Besides BPM systems, there are a number of other systems that can act as event sources in the context of a business process. Sensors, for instance, can produce events that signify the actual execution of a business process activity or report on real-world events that are happening in the context of a business process (e.g. weather change, traffic conditions, position and movement of objects). Transactional systems, such as ERP or CRM systems, can produce events that are related to a business process but lie outside of the control sphere of a BPM system (e.g. low stock alert, creation of a master data record).

Event Processor

The raw events sent out by the various event producers are received and processed by a central event processor, i.e. the actual CEP system.

A typical CEP system comprises three logical components: an input adapter, an event processing network consisting of a set of connected event processing agents, and an output adapter.

The input adapter is responsible for transforming incoming events into the internal event format of the CEP system. Adapters are essential to configure a CEP system to run in a given IT landscape. Usually, there is an input adapter for each connected event producer and/ or each possible incoming event format. After adaptation, events are queued into an input event stream, ready to be processed by an event processing network.

An event processing network (EPN) consists of a number of connected event processing agents (EPA). An EPA is an active component that monitors an event stream to detect and react on certain conditions. Its behavior is defined by a set of so-called event pattern rules. These reactive rules trigger on specific instances or patterns of input events. As a result of executing a rule, the agent creates output events and changes its local state. Three basic classes of EPAs can be distinguished [15]:

A filter EPA performs filtering only. It passes on only those input events that are defined in its event pattern rules. A filter does not perform any transformation of events.

A transformation EPA processes input events that are defined in its event pattern rules and creates output events that are functions of these input events. A transformation EPA can be stateless or stateful. A special case of transformation EPA is an aggregate EPA which takes as input a collection of events and creates a single derived event by applying a function over the input events. Other specializations of transformation EPAs are translate, enrich, project, split, and compose.

A pattern detect EPA performs a pattern matching function on a stream of input events. It emits one or more derived events if it detects a combination of events in the input stream that satisfies a specified pattern.

A basic EPA is usually realized by programming it in a query language such as an SQL variant. In addition, functionality can be made available by using scripting or programming languages inside as an EPA inside the EPN. In certain cases it can be beneficial to even distribute this functionality and have EPAs call remote applications such as a business rules engine to extend their functionality.

Having defined a number of EPAs, EPNs can be composed by linking EPAs so that the output of one agent is the input of another agent. The EPAs communicate by sending their output events to other agents. This allows for building complex event processing logic out of lightweight and reusable building blocks.

Also, EPNs can be composed into even larger networks of event processors for scalability or query distribution purposes. This way one EPN acts as the producer for another EPN which is the consumer.

After processing, events are queued into an output event stream, ready to be forwarded to possible event consumers. An event output adapter is responsible for transforming the outgoing events into external formats that can be processed by the connected event consumers. After adaptation the outgoing data does not necessarily have to represent an event anymore; it can, for example, be in the form of metrics, messages or function calls (see below).

Event Consumer

An event consumer is an entity that receives and visualizes or acts upon events from a CEP system. It is the logical counterpart of the event producer. As depicted in Fig. 1, dashboards, BPM systems, and messaging services are the main event consumers in the context of business activity management.

"A dashboard is a visual display of the most important information needed to achieve one or more objectives; consolidated and arranged on a single screen so the information can be monitored at a glance" [26]. Dashboards provide high-level summaries of information in a concise, clear, intuitive way and often in real-time [26]. This real-time aspect makes dashboards especially appealing in context of event-driven business activity management, as it fits the concept of event streams. Typical information to be displayed in a process dashboard include KPI (e.g. runtime, idle time), occurred or anticipated exceptions (e.g. when certain thresholds of KPI are or will likely be exceeded), or visualizations of the actual flow/ path of process instances. The visualized information is often averaged on the basis of the last n events of an event output stream or on the basis of a certain time window.

Apart from being an event producer, BPM systems also represent a key event consumer in our reference architecture. Based on the insights derived from processing events produced by a BPM system, an event processor might initiate actions that influence processes in that very system (or other BPM systems respectively). An event output adapter of a CEP system might, for instance, make an API call to start new processes or suspend, resume, or cancel running process instances. Likewise, gateways that control the flow of a process instance might be changed by manipulating or updating the conditions of a business rule.

The last class of event consumers in our reference architecture comprises various messaging services that can be leveraged to send alert messages to proactively inform process owners about exceptions or disturbing KPI. A CEP engine can basically use all available channels of communication such as mobile phone short messaging services (SMS), instant messaging, e-mail, or mass messaging services such as Twitter to actuate reactions.

4 Conclusion

Timely insight into a company's value chain is of great importance to be able to monitor and improve its operations. Business processes are the core building blocks of a company's operation and provide a plethora of information that can be tapped into. However, business process analytics are often only the second step in BPM. Furthermore, real-time or at least near real-time insight into processes is almost non-existent in practice. In the past, custom applications have been developed for this requirement. Recently, the topic of operational BI has gained attention through the emergence of more sophisticated CEP engines. However, the dedicated application to business process management is only in its infancy. Nevertheless, it promises cost saving due to the ability to react in a reasonable timeframe to errors, exceptions, or other indicators such as workload, waiting times, etc.

We reviewed the current state of the art of BPM, CEP and the different facets of business process analytics. Based on the existing work, we developed an architecture blueprint for business activity management. Following the tri-partition in CEP we introduced an architecture that is capable of delivering applications ranging from real-time business activity monitoring to context-aware BPM.

It consists of event producers, event processors, and event consumers. We elaborated on the requirements and different options for event producers which provide events to the CEP engine. We also highlighted the requirements for event processors and their integration options with an additional system such as business rules management or applications, e.g. for data mining and simulation. When introducing event consumers, we elaborated on the different relevant output options for BPM and presented detailed insight for closed loop action to performing the full circle of activity management.

The reference architecture is a software independent conceptual model that can be instantiated with basically any combination of BPM and CEP system that is capable of communicating with each other. It enables the monitoring and management of multiple BPM engines through its loose coupling and can also serve as the basis for future context-aware BPM systems.

We have implemented a proof of concept of this reference architecture using multiple SAP NetWeaver BPM servers as BPM engines, the Sybase Aleri Streaming Platform as a CEP engine, and SAP BusinessObjects Xcelsius Enterprise as a visualization frontend. All of the above can be downloaded for trial and evaluation and represent state-of-the-art BPM and CEP technology. The implementation is able to showcase the majority of the described use cases. Limitations are due to the shortness of the project and not yet existing standard interfaces of the BPM engine to receive and understand process events. We refrained from hardwiring as much as possible in order to retain generalizability.

We observed several shortcomings in the process: These include but are not limited to the lack of a common business event format, the lack of a standardized set of logging events, and the lack of standardized operations to trigger BPM engines for automated insight to action. On the conceptual side there is a clear need for an integrated or at least synchronized modeling paradigm for business processes and reporting on the one hand and business processes and their interaction with event processing networks on the other hand

References

1. Word, J.: Business Network Transformation: Strategies to Reconfigure Your Business Relationships for Competitive Advantage. Jossey-Bass, San Francisco (2009)
2. Janiesch, C.: Implementing Views on Business Semantics: Model-based Configuration of Business Documents. In: 15th European Conference on Information Systems (ECIS), St. Gallen, pp. 2050–2061 (2007)
3. zur Muehlen, M., Shapiro, R.: Business Process Analytics. In: vom Brocke, J., Rosemann, M. (eds.) Handbook on Business Process Management: Strategic Alignment, Governance, People and Culture, pp. 137–157. Springer, Berlin (2010)
4. Costello, C., Molloy, O.: Towards a Semantic Framework for Business Activity Monitoring and Management. In: AAAI 2008 Spring Symposium: AI Meets Business Rules and Process Management, pp. 1–11. AAAI Press, Stanford (2008)
5. Nesamoney, D.: BAM: Event-driven Business Intelligence for the Real-Time Enterprise. DM Review 14, 38–40 (2004)
6. Hermosillo, G., Seinturier, L., Duchien, L.: Using Complex Event Processing for Dynamic Business Process Adaptation. In: 7th IEEE International Conference on Service Computing (SCC), Miami, FL, pp. 466–473 (2010)
7. Rosemann, M., Recker, J., Flender, C., Ansell, P.: Understanding Context-Awareness in Business Process Design. In: 17th Australasian Conference on Information Systems (ACIS), Adelaide, pp. 1–10 (2006)

8. Object Management Group Inc.: Business Process Modeling Notation (BPMN) Specification 1.0 (2006), http://www.bpmn.org/Documents/OMG%20Final%20Adopted%20BPMN%201-0%20Spec%2006-02-01.pdf
9. Davenport, T.: Process Innovation: Reengineering Work Through Information Technology. Harvard Business School Press, Boston (1993)
10. Johansson, H.J., McHugh, P., Pendlebury, A.H., Wheeler, W.A.: Business Process Reengineering: Breakpoint Strategies for Market Dominance. John Wiley & Sons, Chichester (1993)
11. zur Muehlen, M., Indulska, M.: Modeling Languages for Business Processes and Business Rules: A Representational Analysis. Information Systems 35, 379–390 (2010)
12. Becker, J., Kugeler, M., Rosemann, M. (eds.): Process Management: A Guide for the Design of Business Processes. Springer, Berlin (2003)
13. Pedrinaci, C., Domingue, J., de Medeiros, A.K.A.: A Core Ontology for Business Process Analysis. In: Bechhofer, S., Hauswirth, M., Hoffmann, J., Koubarakis, M. (eds.) ESWC 2008. LNCS, vol. 5021, pp. 49–64. Springer, Heidelberg (2008)
14. Verbeek, H.M.W., Buijs, J.C.A.M., van Dongen, B.F., van der Aalst, W.M.P.: XES Tools. In: CAiSE Forum 2010. CEUR, vol. 592, pp. 1–8 (2010)
15. Etzion, O., Niblett, P.: Event Processing in Action. Manning Publications, Cincinnati, OH (2010)
16. Grigori, D., Casati, F., Castellanos, M., Dayal, U.: Business Process Intelligence. Computers in Industry Journal 53, 321–343 (2004)
17. van der Aalst, W.M.P., Reijers, H.A., Weijters, A.: Business Process Mining: An Industrial Application. Information Systems 32, 713–732 (2007)
18. Han, J., Kamber, M.: Data Mining: Concepts and Techniques, 2nd edn. Morgan Kaufmann, Amsterdam (2006)
19. Eckert, M.: Complex Event Processing with XChange EQ: Language Design, Formal Semantics, and Incremental Evaluation for Querying Events. LMU München: Faculty of Mathematics, München (2008)
20. McCoy, D.W.: Business Activity Monitoring: Calm Before the Storm. Gartner Research Note LE-15-9727 (2002), http://www.gartner.com/resources/105500/105562/105562.pdf
21. Bianchi, M., Boyle, M., Hollingsworth, D.: A Comparison of Methods for Trend Estimation. Applied Economics Letters 6, 103–109 (1999)
22. Luckham, D.: The Power of Events: An Introduction to Complex Event Processing in Distributed Enterprise Systems. Addison-Wesley Professional, Boston (2002)
23. Workflow Management Coalition (WfMC): Business Process Analytics Format Draft Version 2.0 (2008), http://www.wfmc.org/index.php?option=com_docman&task=doc_download&gid=437&Itemid=72
24. Zang, C., Fan, Y., Liu, R.: Architecture, Implementation and Application of Complex Rvent Processing in Enterprise Information Systems Based on RFID. Information Systems Frontiers 10, 543–553 (2008)
25. von Ammon, R., Ertlmaier, T., Etzion, O., Kofman, A., Paulus, T.: Integrating Complex Events for Collaborating and Dynamically Changing Business Processes. In: 2nd Workshop on Monitoring, Adaptation and Beyond (MONA+), Stockholm, pp. 1–16 (2009)
26. Few, S.: Dashboard Confusion (2004), http://intelligent-enterprise.informationweek.com/showArticle.jhtml;jsessionid=BJXFCGOZCMUE3QE1GHPCKHWATMY32JVN?articleID=18300136