Business Process Discovery in Real Time

João J. Oliveira

IST – Technical University of Lisbon
Av. Prof. Dr. Aníbal Cavaco Silva
2744-016 Porto Salvo, Portugal
joao.r.oliveira@ist.utl.pt

Abstract. The *process mining* research field is fairly recent, however it counts with numerous techniques that enable extraction of business process models based on event logs produced in information systems. Typically, these events are extracted to a log file and then analyzed with proper tools, such as ProM. This type of analysis is made over historical data, and by the time it is performed, results could, eventually, be outdated, in the case that business rules/routines had changed or evolved. The main objective of this dissertation is to develop a *process mining* tool that can be used in real time as the events are being produced. Furthermore, it is imperative to study the behavior of the tool, as the events occur, being incorporated in real time and updating the models instantaneously.

Key words: Process Mining, Complex Event Processing, Event-Driven Architecture, ECA Rules

1 Introduction

Monitoring business processes is important to improve the efficiency of organization processes. The Process-Aware Information Systems (PAIS) enable the recording of events – in the event log – that are being generated by the organization, through the execution of it. Enterprise Resource Planning (ERP), Customer Relationship Management (CRM) or Supply Chain Management (SCM) are examples of the referred ones, among the other existing systems at present [1,2,3].

Typically the discovery of business processes is accomplished throughout the usage of log files. These files contain historical data, being so the basis for the different analysis. Considering that events are generated all the time, it may not be possible to access older ones, however it could be of interest to provide access to the latest. This suggests that it would be very useful to apply techniques for the discovery process based on the events that occur in real time. Then we have the basis for this work whose main objective is to apply the traditional process mining techniques in real time, as events are being produced.

Given the above context, the objectives of this work are as follows:

- Acquire insight in the area of process mining and existing tools;
- Develop or adapt process mining techniques that can be used in real time;
- Study the behavior of these techniques as the events are being processed, in real time;
- Develop a tool for processing events in real time;
- Demonstrate the use of these techniques in a realistic case study.

This paper is organized as follows: Section 2 provides an overview of process mining and its perspectives. Section 3 provides an overview of related work involving event processing, giving focus to an engine named Esper, that is used in this work. Furthermore, Section 4 presents the global architecture of our proposed solution and the used technologies to implement it. Also, in Section 5 we demonstrate the approach in real-world case study where the goal was to understand the behavior of models, when processed in real time. Last but not least, Section 6 concludes this paper, describing the main contributes of this work.

2 Discovery and Analysis of Business Processes

The process mining area is concerned with the discovery, monitoring and optimization of real processes by extracting information from an event log [4]. Thus, it is assumed that it is possible to record events such that (i) each event is an *activity* (*i.e.*, a well-defined phase of the process), (ii) each refers to a *case* (*i.e.*, a process instance), (iii) each can have a *performer* also referred to as a *originator* (the person who performs or initiates the activity) and finally the events have a *timestamp* attribute and are totally ordered.

There are three main perspectives: (1) process perspective, (2) organizational perspective and (3) the case perspective [5,6,7]. The scope of this work regards only the two first perspectives, which we will address, as follows:

- The Process Perspective is concerned with control flow, *i.e.*, the sequence of activities. The purpose of mining, through this perspective, is to find a good characterization of all possible paths, for instance, expressed in terms of Petri nets or Event Process Chains (EPC) [8]. Examples of such techniques in this perspective are α and α++ algorithms [9,10], the Heuristic algorithm [11], genetic algorithm [12,13] and the Fuzzy Algorithm [14,15].
- The Organizational Perspective ([16]) gives focus to the attribute *originator*, *i.e.*, which actors are involved and how they relate. The goal here is to structure the organization by classifying people in terms of roles and organizational units that perform, or to present the relations between individual actors (*i.e.*, to build a social network).

Relatively to the organizational perspective, we will use two metrics in this work, to construct social networks, described below:

- *Metrics based on (possible) causality* monitor for individual cases, how the work is passed between people who perform it. One of the examples, and which is given focus in this paper, is the *Handover of Work*. The *Handover of Work* presents who delivers work to whom, leaving it possible, through the event log deduce this conclusion through two subsequent activities in the same case or process instance.
- *Metrics based on common cases* ignore causal dependencies, making the count on how often two individuals perform activities for the same case. *Working Together* Metric is an example of this approach. In the metric *Working Together*, if individuals work in the same case, then they have a stronger relationship compared with individuals who rarely work in the same case.

As previously mentioned, in this paper we will focus on the metrics of *Handover Work* and *Working Together*.

3 Event Processing in Real Time

The analysis techniques presented in section 2 are meant to process historical data in order to extract models based on complete instances. However, in his recent book [17], Professor Wil van der Aalst mentions the possibility of business process analysis to be performed in "*online*" fashion instead of the traditional "*offline*" way. This *online* mode corresponds precisely to the real time processing that is developed in this work, being closely related to the subject area of complex event processing (CEP), which also advocates the processing events in real time.

Event-driven information systems require an automated and systematic processing of events. The CEP encompasses methods, techniques and tools to process events in real time. Useful knowledge is obtained from high-level, taking as its starting point the low-level events (highly detailed).

There are several academic and commercial implementations in this field, which at present, vary greatly in the expressiveness of the rules and data queries, ease of use, performance and how they are integrated into the overall structure of the organization's information technology [18].

To query data streams several approaches have been developed, based on the SQL syntax, which has revealed to be the best successful approach, being also supported in an efficient and scalable manner for many products. The best known solutions at this level are the *Oracle CEP*, *Borealis* [19], *Aurora* [20], *Coral8*, *StreamBase*, *Aleri* and the open-source project *Esper* [21]. The general term to classify data stream query languages is *EPL*. The best known data stream query languages are CQL (Continuous Query Language) of STREAM project and EQL (Event Query Language) of Esper project. These languages have in common the use of basic clauses provided by the SQL syntax, for instance SELECT, FROM, WHERE. But in this case, data flows replace tables as data source and events replace the tuples as the basic unit of data. In our work, we used Esper to process events in real time, so we discuss it in more detail.

Esper engine is an open-source software, which combines two styles of event processing: ESP (Event Stream Processing) and CEP (Complex Event Processing). The CEP module was inspired on the behavior model *state machine*, being an intuitive basis for expressing CEP event patterns. Relatively to ESP, this was based on the *delta networks* (a formal definition can be found at [22]).

The Esper[1] works similarly to an inverted data base, *i.e.*, instead of storing the data and then make queries on data, it allows applications to store query expressions – which remain static – querying data with the instructions previously specified. The execution model, is thus, continuous instead of being performed only when the query is submitted [23].

The engine also provides two main mechanisms to process events: *event patterns* and *events flow queries*.

Its event query language is based on the SQL, named EQL, and is used to express clauses, as *filtering*, *joins*, *aggregation* over multiple data streams, whereas standard language is used to define more complex patterns in different types of events.

4 Proposed Approach

The fundamental concepts of our approach are (1) the events, a message containing information about an operation performed by/at a system (2) the event channels, flows of events that occur between producers and consumers, (3) the business processes, a set of activities whose objective is to produce value to customer and (4) the process mining techniques, which allow to extract useful information from an event log.

The architecture, shown in Figure 1, is separated into two steps: (1) production of events and (2) event processing. In the first stage of this architecture, these are referred to an event channel that supports the flow between the information systems that execute business processes and, event log file where events will be subsequently recorded. The number of event channels will depend on each information system specification, because it is in this step that is performed an analysis relative to the need of creating a new log, hence instancing a new event channel. Alternatively, we could reuse the components already in place. These channels are the source of input events to the tool developed in this work, which performs its processing. Through a publish/subscribe mechanism, events are published by the channel and subscribed by the processing events tool.

In the second stage, events are analyzed individually and in the case of existing a registered subscriber to the type of event under review, it will be sent to that subscriber, which subsequently applies the process mining technique, resulting in a model, that can either be a control flow or social networks type. It should be noticed the fact that in this work, the process mining is made in real time, in contrast to existing tools that process a static event log, which is the main contribution this work.

[1] http://esper.codehaus.org/

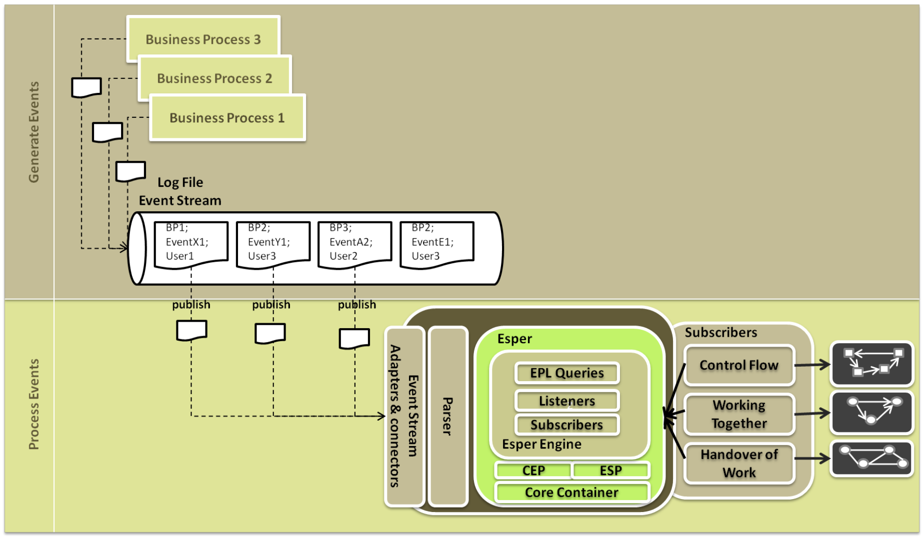


Fig. 1. Architecture of proposed approach

During the process execution it is possible to extract the events being produced (that allows constructing log seen in Figure 3). For this, we develop a periodic query (every 5 seconds), that verifies if new events were stored in a BizAgi database (supported by *Sql Server 2008*), and if so, return this new data to a private message queue (implemented in Microsoft Message Queue). For each event that is sent to a message queue, the timestamp of that event is recorded in a key of *Windows Registry*.

Concluding all the aforementioned processes, the resulting messages are sent to our tool, and for this purpose we have implemented an asynchronous mechanism (callback). The tool registers itself in the message, and when it receives new events from database, our tool is notified, and gets the new messages (Adapters component). When our tool receives the new messages, these must be parsed (Parser component) to an object recognized by our application, with a specific structure, to store all information that came within the message. The instantiated object is then sent to Esper engine, which processes events. This engine is configured to trigger the software modules, through the EPL statements. Each statement must be configured to only receive and process the subscribed events. This is accomplished through the implementation of a known software design pattern, named *observer pattern* (a subset of *publish / subscribe* pattern). In our tool, each Listener corresponds to an EPL statement. So, when a condition is verified, the events are sent to the associated subscriber. The subscribers are materialized in three modules, namely: (i) *Control Flow*, (ii) *Handover of Work* and (iii) *Working Together*. Each one of the referred modules after receiving the new events, and through the respective algorithms (see section 2), performs the processing. The last step is to convert the process models into graphical models. This is accomplished through the conversion of models processed to Dot language, and then this data is passed to GraphViz tool[2].

It should be noticed that, the produced models, are updated and at the same time the results are presented to the user, that is, each time a new event reaches our tool. Our implementation was developed in C# language, and using the NEsper library.

5 Case Study

This section presents the case study used to validate the approach described in section 4. This process was implemented in a commercial workflow system, named BizAgi. This tool allows to model, automate, execute and maintain a business process. The BPMN model in Figure 2, refers to activities that are in the domain of this process. The initial activity that triggers the business process is the fulfillment order. This request has information regarding the product to order, as well as its amount, among other attributes needed to the processing of order. Then, the document must be approved by the responsible manager. If the order receives approval, then the product is ordered. In the next step, the accounting department will make payment and in parallel, at the warehouse, the goods are recorded as received and upgraded the stock of products received. Finally, the accounting department closes the request. If the request does not receive approval, then the document is filed, in order to keep track of requests.

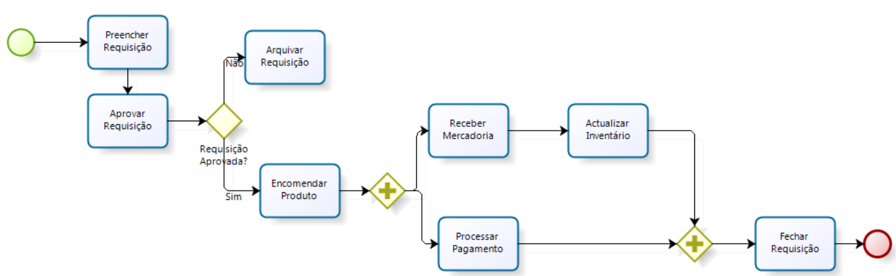
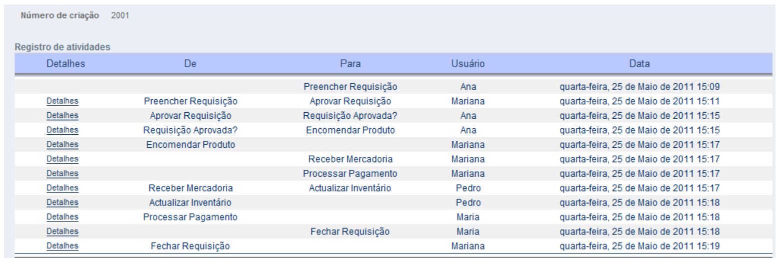


Fig. 2. Model BPMN of case study[3]

[2] http://www.graphviz.org/

[3] Adapted from Ferreira, D.R.: Mineração de processos - o elo que faltava na gestão de processos de negócio (Junho 2010), Marabá, Pará, Brasil, keynote presentation at VI Simpósio Brasileiro de Sistemas de Informação (SBSI 2010)

The implementation of this process gives rise to an event log similar to that shown in Figure 3. In the log, the information key that is necessary to do process mining can be identified, namely: the process instance identifier, name of activity, name of the actor and the timestamp that refers to the conclusion of the activity. It is from this event log that we can extract models of control flow and social network associated with this process.



Número de criação 2001

Registro de atividades

Detalhes	De	Para	Usuário	Data
		Preencher Requisição	Ana	quarta-feira, 25 de Maio de 2011 15:09
Detalhes	Preencher Requisição	Aprovar Requisição	Mariana	quarta-feira, 25 de Maio de 2011 15:11
Detalhes	Aprovar Requisição	Requisição Aprovada?	Ana	quarta-feira, 25 de Maio de 2011 15:15
Detalhes	Requisição Aprovada?	Encomendar Produto	Ana	quarta-feira, 25 de Maio de 2011 15:15
Detalhes	Encomendar Produto		Mariana	quarta-feira, 25 de Maio de 2011 15:17
Detalhes		Receber Mercadoria	Mariana	quarta-feira, 25 de Maio de 2011 15:17
Detalhes		Processar Pagamento	Mariana	quarta-feira, 25 de Maio de 2011 15:17
Detalhes	Receber Mercadoria	Actualizar Inventário	Pedro	quarta-feira, 25 de Maio de 2011 15:17
Detalhes	Actualizar Inventário		Pedro	quarta-feira, 25 de Maio de 2011 15:18
Detalhes	Processar Pagamento		Maria	quarta-feira, 25 de Maio de 2011 15:18
Detalhes		Fechar Requisição	Maria	quarta-feira, 25 de Maio de 2011 15:18
Detalhes	Fechar Requisição		Mariana	quarta-feira, 25 de Maio de 2011 15:19

Fig. 3. Event Log produced in BizAgi

This section also discloses the results achieved through the proposed approach, of the events that were processed in real time, by our tool. To note that, in this analysis were processed 91 events of 8 process instances.

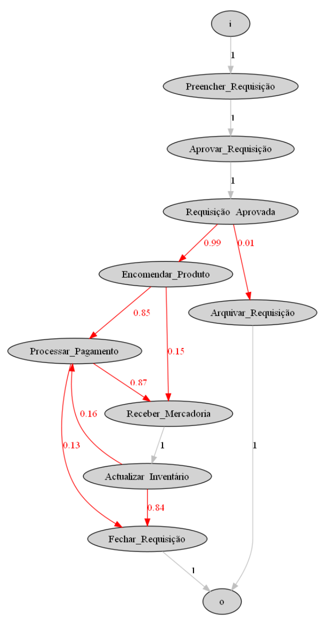


Fig. 4. Control Flow Model

The control flow model can be seen in Figure 4, showing the red color differences compared to the preceding matrix produced. Through the percentages shown in the model, it is possible to realize what is the most frequent transitions between activities.

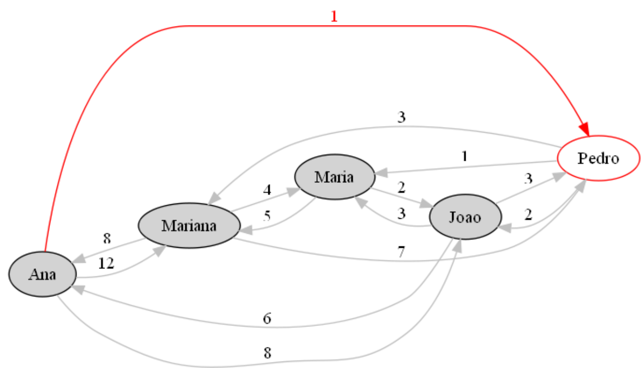


Fig. 5. Handover of Work Model

Relatively to the technique of *Handover of Work*, *i.e.*, the behavior observed in relation to the supply of labor between pairs of actors (see section 2), can be viewed in in Figure 5. Finally, the model built to *Working Together*, is exhibited in Figure 6. This model, represents the interactions between pairs of actors, as well as in how many cases they work together.

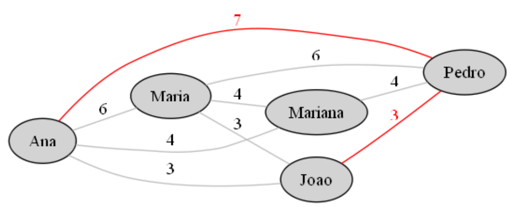


Fig. 6. Working Together Model

6 Conclusion

In this work, we established a relationship between process analysis in real time and complex event processing, an emerging theme in the area of information systems. Furthermore, extraction algorithms have been adapted for control flow and social network, to allow the processing of an event in real time, as they are being produced. An architecture to support the proposed approach was defined, which includes functionalities to query a database of a workflow system, and the transmission of data via a message queue for the events engine, which then feeds the tool construction and visualization of models. Mechanisms were developed not only to visualize the extracted models, but also to analyze the differences between a model and its earlier versions, as new events are being included in these models. The ultimate contribution made in this work, was to show that the proposed approach and its tools can be integrated with a commercial workflow system and thus becoming possible to monitor and analyze the events of process execution in real time, on a platform that was not originally designed for this function.

References

1. M. Dumas, W. M. van der Aalst, and A. H. ter Hofstede. *Process-Aware Information Systems: Bridging People and Software through Process Technology*. Wiley & Sons, Inc., 2005.
2. Wil M. P. van der Aalst. Process-aware information systems: Lessons to be learned from process mining. *T. Petri Nets and Other Models of Concurrency*, 2:1–26, 2009.
3. Nicholas Charles Russell. *PhD Thesis: Foundations of Process-Aware Information Systems*. Queensland University of Technology, 2007.
4. W. M. P. van der Aalst and A. J. M. M. Weijters. Process mining: a research agenda. *Computers in Industry*, 53(3):231 – 244, 2004. Process / Workflow Mining.
5. W van Der Aalst, T Weijters, and L Maruster. Workflow mining: discovering process models from event logs. *IEEE Transactions on Knowledge and Data Engineering*, 16(9):1128–1142, 2004.
6. W. M. P. van der Aalst, H. A. Reijers, A. J. M. M. Weijters, B. F. van Dongen, A. K. Alves de Medeiros, M. Song, and H. M. W. Verbeek. Business process mining: An industrial application. *Inf. Syst.*, 32:713–732, July 2007.
7. Chen Li. *Mining process model variants : challenges, techniques, examples*. PhD thesis, University of Twente, Enschede, November 2010. SIKS Dissertation Series No. 2010-47.
8. Ana Karla Alves de Medeiros. *Genetic Process Mining*. PhD thesis, Technische Universiteit Eindhoven, 2006.
9. A.K. Alves de Medeiros, B.F. van Dongen, W.M.P. van der Aalst, and A.J.M.M. Weijters. Process mining: Extending the a-algorithm to mine short loops. BETA Working Paper Series, WP 113, Eindhoven University of Technology, Eindhoven, 2004.
10. Lijie Wen, Jianmin Wang, and Jia-Guang Sun. Detecting implicit dependencies between tasks from event logs. In Xiaofang Zhou, Jianzhong Li, Heng Tao Shen, Masaru Kitsuregawa, and Yanchun Zhang, editors, *APWeb*, volume 3841 of *Lecture Notes in Computer Science*, pages 591–603. Springer, 2006.
11. A J M M Weijters, Wil M P van Der Aalst, and A K A Medeiros. Process mining with the heuristicsminer algorithm. *Technische Universiteit Eindhoven Tech Rep WP*, 166(WP 166):1–34, 2006.
12. Wil M P van Der Aalst, A K A De Medeiros, and A J M M Weijters. Genetic process mining. *Application and theory of Petri nets*, 3536:48–69, 2005.

13. A.K. Alves de Medeiros, A.J.M.M. Weijters, and W.M.P. van der Aalst. Genetic process mining: An experimental evaluation. *Data Mining and Knowledge Discovery*, 14(2):245–304, 2007.
14. C W Gunther and W M P Van Der Aalst. *Fuzzy Mining - Adaptive Process Simplification Based on Multi-perspective Metrics*, volume 4714, pages 328–343. Springer-Verlag, 2007.
15. Gabriel M. Veiga and Diogo R. Ferreira. Understanding spaghetti models with sequence clustering for ProM. In *Business Process Management Workshops, BPM 2009 International Workshops, Ulm, Germany, September 7, 2009. Revised Papers*, volume 43 of *Lecture Notes in Business Information Processing*, pages 92–103. Springer, 2010.
16. Minseok Song. Mining social networks: Uncovering interaction patterns in business processes. In *International Conference on Business Process Management (BPM 2004), volume 3080 of Lecture Notes in Computer Science*, pages 244–260. Springer-Verlag, 2004.
17. Wil M. P. van der Aalst. *Process Mining: Discovery, Conformance and Enhancement of Business Processes*. Springer, 2011.
18. Josef Schiefer, Heinz Roth, Hannes Obweger, and Szabolcs Rozsnyai. Event data warehousing for complex event processing. In Pericles Loucopoulos and Jean-Louis Cavarero, editors, *RCIS*, pages 203–212. IEEE, 2010.
19. D J Abadi, Y Ahmad, M Balazinska, Ugur Cetintemel, M Cherniack, Jeong-Hyon Hwang, W Lindner, A S Maskey, A Rasin, E Ryvkina, and et al. The design of the borealis stream processing engine. *Proc of CIDR*, pages 277–289, 2005.
20. D J Abadi, D Carney, U Cetintemel, M Cherniack, C Convey, S Lee, M Stonebraker, N Tatbul, and S Zdonik. Aurora: a new model and architecture for data stream management. *The VLDB Journal The International Journal on Very Large Data Bases*, 12(2):120–139, 2003.
21. Michael Eckert and François Bry. Complex event processing (cep). *Informatik-Spektrum*, 32(2):163–167, 2009.
22. D.M. Dias and J.R. Jump. Analysis and simulation of buffered delta networks. *Computers, IEEE Transactions on*, C-30(4):273 –282, 1981.
23. EsperTech. Esper reference documentation version: 3.4.0. Technical report, EsperTech Inc, 2009.