Learning Semantic Models from Event Logs

Vasiliki Sfyrla[1], Ioannis Partalas[1], Richard Yann[2], Sebastian Maunoury[2]

[1] Viseo Technologies R&D, 4 avenue Doyen Louis Weil, Grenoble, France
{vasiliki.sfyrla,ioannis.partalas}@viseo.com
[2] MMA, 160 rue Henri Champion, Le Mans, France
{yann.richard,sebastien.maunoury}@groupe-mma.fr

Abstract. In this work we present an approach to extract the business logic of an application based on its generated logs. To do so we use process mining techniques to extract the model from the logs of an industrial application which are often large and hence difficult to analyze. We propose here a methodology to group the log according to similar elements, so that it can be presented to the user in separate fragments. We enrich this view of the model with semantic information based on the ontology we built.

Keywords: process mining, clustering, ontologies, business process

1 Introduction

One of the biggest challenges of information systems concerns the modernization of applications by introducing new technologies and functionalities. Modernizing an application can be a costly and time consuming task, especially when applications are developed using obsolete technologies and they lack a clear description of the underlying business logic.

The business logic of a system is described in the conceptual model which is a composition of the different entities and the relations between them. The conceptual model supports developers to understand different aspects of the application, independently of any implementation issues. Unfortunately, many of the modern systems lack of this conceptual model or are based on models which are neither maintained nor documented.

One of the most common approaches for extracting the different models of an application is based on the analysis of the source code using reverse engineering technologies. The extracted models can then be used to reconstruct different view points of the system. However analyzing the source code can be very expensive especially when the system is developed with an emphasis on perspectives different than the business process (e.g. COBOL). The business logic is spread out across the entire system which can encompass millions of lines of source code. As a result analyzing the integrality of the source code is necessary in order to create the model describing the business logic of the application.

In this work we use the events logs that are generated during the execution of an application in order to discover a model that describes the business logic.

An event log is a recorded event that is related to an activity, at a particular timestamp, from a particular user. Existing techniques from the domain of process mining can extract the process of a system. However, this model, especially when it comes from large logs is difficult to analyze and to extract information due to its big size. Our goal is to improve the quality of the extracted data and facilitate their interpretation. The proposed approach, extracts an abstract representation of the initial model enriching it with semantic knowledge. To achieve that, we use clustering analysis and ontologies.

The work presented in this paper is part of the ITM factory[3] collaborative project. The project solutions to the industries for the rapid maintenance of information systems and for software modernisation. The goal is to improve or create new business value from existing applications, especially when the software becomes outdated and incompatible with new technologies. The solutions are based on reverse engineering methodologies. That is, the model of the application is extracted, analysing the source code. Based on the extracted application model, we can derive different view points such as the architecture, the process and the business logic. The extracted models can then be enriched or modified to meet the objectives of the given project. The logic of the system can be enhanced with new business rules making use of the old ones, that is avoiding rewriting existing parts of the application. Similarly, new technologies and techniques can be applied to the corresponding model. This project joins together French specialists of the domain, Sodifrance (http://www.sodifrance.fr/) and Mia-Software (www.mia-software.com), leaders in the project of modernisation, offering methodologies and tools as well as the public research laboratory Inria-AtlanMod (http://www.inria.fr/en/teams/atlanmod), expert in the domain of model driven engineering. The project uses data from the French insurance company MMA[4] which is the industrial partner of this project. MMA is an insurance company proposing home, auto and business insurances.

Recent work has shown that one can rely on clustering techniques for extracting a model from user behavior. More specifically, Song et al. [11] define different types of profiles where a profile is a set of related items each of which describes the trace from a specific perspective. We also incorporate a clustering approach which is coupled with semantic knowledge through the use of an ontology in the level of clusters. The proposed model is used to describe the business logic of the system under consideration.

The rest of the paper is structured as follows. The background to this work is described in Section 2. Section 3 describes the case study. The methodology we follow is detailed in Section 4. Finally, Section 5 concludes this work and discusses directions for future work.

[3] http://www.viseo.com/fr/offre/le-projet-itm-factory
[4] http://www.mma.fr/

2 Background

Logging is a means of tracking actions that took place during the execution of a system from a user at a specific timestamp. Information such as actions, originators of this action, used resources and timestamp are some of the information that can be stored in an event log when executing an information system. Process Mining [14] is a model-driven approach aiming at constructing process models based on available events logs. A process model describes in a structured way and using well defined formalisms several information related to the process like action, originator and timestamp, etc. Process models can be described in different formalisms like modeling languages e.g. BPMN [8], or formal models e.g. Petri Nets [7].

A fundamental goal of process mining is the discovery and extraction of the model describing the system process. The reconstruction of the model is the aim itself but process mining it is not limited to that. Conformance checking [13] can be performed to compare the derived model with the process log and monitor possible deviations. Notice that while in discovery the only algorithm's input is the log, in conformance two inputs are considered: the log and the process model, the latter can be either obtained by a discovery technique or manually designed from an expert. Analysis might aim also at additional objectives including optimization of the process, testing for satisfiability of safety properties, performance analysis etc. In contrast to data mining, process mining techniques focus on the process perspective, and hence the causal relations between different events of a process are identified. Process mining is applicable to systems that record their behavior and produce process logs (e.g. ERP systems [10]).

The next section illustrates an example of an event log and how process mining can be used to extract the corresponding process model.

3 A Case Study

In this section, we present a case study that will be used throughout the paper to demonstrate our methodology. The event log chosen for the purpose of this case study is an extract of logs from an insurance company application. It describes a process of rescinding an insurance contract. The application is written in french language, so some of the terms in the rest of the paper will be presented in French.

The log contains several traces, each of which corresponds to a process instance. Each trace is a sequence of events represented by an action, the action originator, its timestamp, the resources, the classes etc. Some examples of identified actions are "register invoice", "denounce contract", "register new case", etc.

For the purpose of this work, we keep information about the action, the originator and the timestamp. Table 1 shows a fragment of this log. It illustrates three traces, denoted as "case id". For each trace, different activities are executed by different originators at given timestamps. For example, in case 1, the originator

"A6926A04" executes an instance of the activity "RegisterModifyPerson" at the timestamp "2013-12-19 13:59:33".

Case id	Originator	Timestamp	Activity
1	S051369	2013-12-19 11:55:09	ConcretizeProject
1	A6926A04	2013-12-19 13:59:33	RegisterModifyPerson
1	S055774	2013-12-19 14:07:44	RegisterInvoice
1	S028429	2013-12-19 16:22:38	ConcretizeProject
1	A0207010	2013-12-19 18:45:56	DenounceContract
1	A0207010	2013-12-19 18:46:35	RegisterAccountAssignment
1	A5919003	2013-12-20 10:14:20	RegisterAccountAssignment
1	S042196	2013-12-20 15:47:03	RegisterInvoice
2	A5988004	2013-12-10 10:05:48	RegisterProvisionalContract
2	A7743002	2013-12-10 11:48:32	RegisterNewCase
2	A5988004	2013-12-10 14:17:02	RegisterAccountAssignment
2	A7743002	2013-12-10 14:19:54	RegisterAccountAssignment
2	A5988004	2013-12-10 15:07:08	RegisterModifyPerson
2	A5988004	2013-12-10 16:16:19	RegisterAmendment
2	S016755	2013-12-11 14:53:46	ConcretizeProject
3	A6112A03	2013-12-04 10:52:31	RegisterAccountAssignment
3	A6112A03	2013-12-04 10:42:50	RegisterAmendment
3	A6926006	2013-12-05 09:23:25	RealiserAmenagementTemporaire
3	A6926006	2013-12-05 10:48:22	RegisterNewCase
3	A6926006	2013-12-05 10:50:12	DenounceContract
3	A6926006	2013-12-05 10:50:41	RegisterAccountAssignment
3	A5945022	2013-12-05 17:54:30	RegisterProvisionalContract
3	A5945022	2013-12-05 17:54:30	RegisterAccountAssignment

Table 1. An event log extract.

We use a fragment of the log that contains 670 traces (case id or process instances) and a total of 24 activities executed by 459 originators.

To extract the model from this event log we use ProM [15], a platform independent open source framework which supports a wide variety of data and process mining techniques in form of plugins. Using the Fuzzy Miner plugin we generate a model describing the process of the log [6]. The model is illustrated in Figure 1. The main activity of the process is to manage different information about the situation of a client. It modifies information on the client's dossier such at the Bank Address, it makes provisional planning, it registers new business plans, invoices and finally manages the provisional and final contract.

4 Methodology

In this section we describe a methodology for extracting the process of an application based on its execution logs.

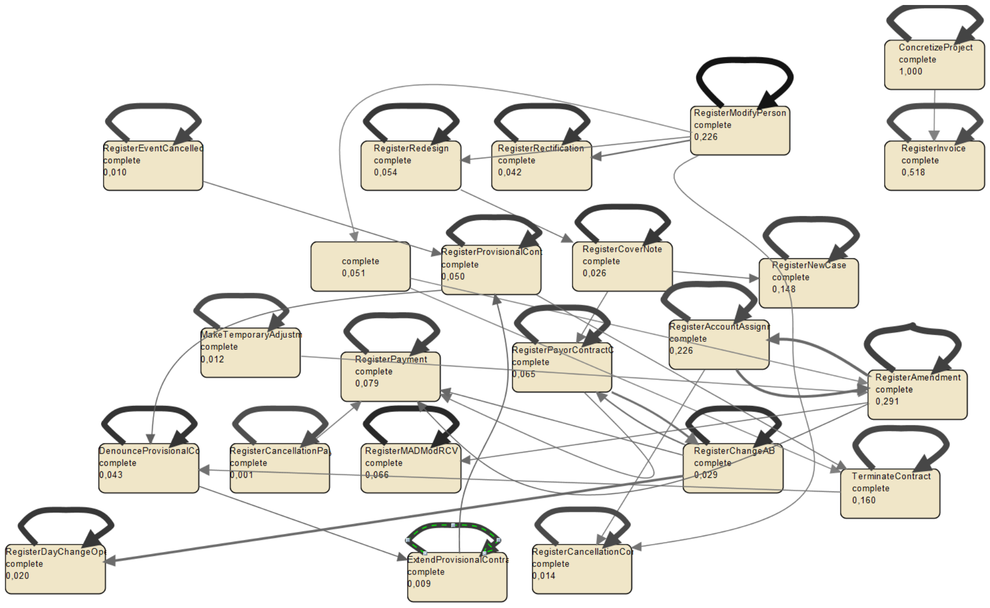

Fig. 1. The process model of the running example as extracted from the Fuzzy Miner tool.

The process model of the application is obtained by analyzing the event logs and applying process mining algorithms. For event logs coming from real-life applications, like the insurance company application we are working on, the discovered model is large and difficult to explore and analyze. The end user, in order to understand the process of the application, needs to read the process model which is very complex, containing millions of activities. And still, the information he obtains cannot help him explore the application and detect problems or propose modifications.

Giving an abstract view of the extracted model and assigning a meaning to each element of this abstract view is a big challenge.

Techniques for solving the abstraction problem have already been proposed. Most of these approaches are based on structural properties such as identifying patterns [3], translating the log into abstract sequence of events [4], [2] or techniques that aim to provide different levels of details that correspond to different execution scenarios [5]. In contrast to these works, our approach aims to give an abstract view of the process model based on its semantic meaning. That is, we explore the event logs based on the number of actions executed by users based on which we create groups of semantically related events in the model.

To overcome this challenge, we propose a methodology that incorporates clustering and semantic analysis. We cluster users (originators) based on their behavior, aiming to give a meaning to each of these clusters.

To achieve our goal we follow the methodology described below:

1. Extract the process model using the Fuzzy Miner algorithm [6].
2. Analyze the event logs and extract one profile for each user such that, the profile of a user contains the frequency of each executed activity. Cluster of users with common behavior.
3. Introduce domain based knowledge, analyzing semantically the activities identified in the event log. Identify activities of high importance and connect semantically the different activities using ontologies.
4. Identify in the model the produced clusters and associate them with the semantic annotations.

Figure 2 captures the different steps of our methodology.

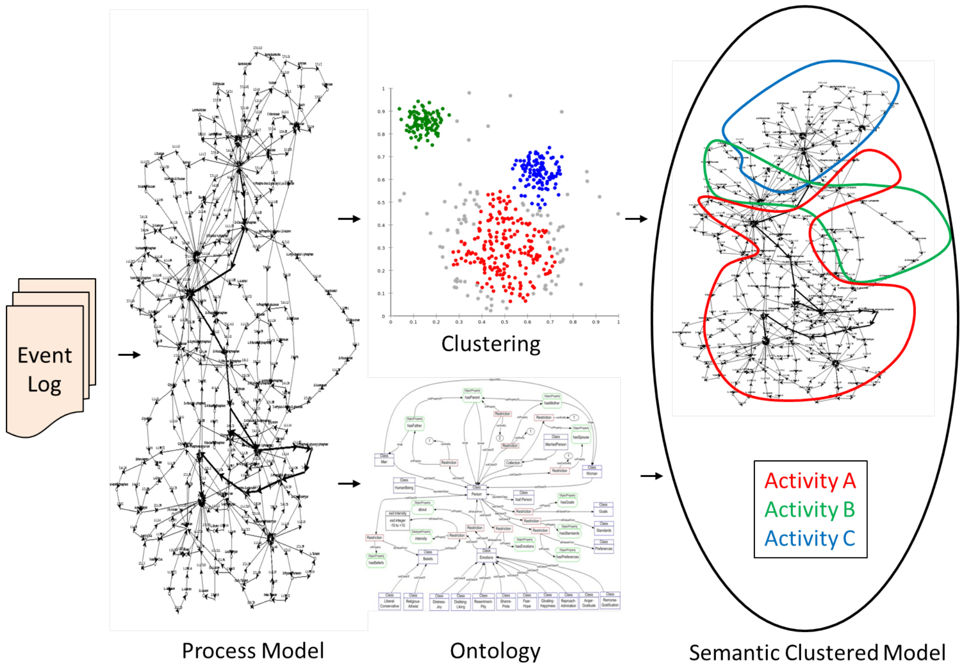

Fig. 2. The proposed methodology which couples clustering and knowledge from a semantic ontology.

4.1 Clustering

Clustering is an approach for grouping data into subsets. The objective of a clustering algorithm is to group together similar objects while forming distinct clusters where objects in one cluster are dissimilar to those in the other clusters. Clustering falls in the unsepervised learning case as no prior information is available on the class membership of each object [1]. Formally, given a set of objects $V = \{v_1, \ldots, v_N\}$ expressed in a vectorial form, one seeks to assign them in K classes optimizing an objective function $\mathcal{F}$ that measures the quality of the clustering:

$$\arg\min_{P} \mathcal{P}$$

where P refers to any partitioning of the data.

One of the most popular clustering algorithm is k-means which groups the data in order to minimise the inter-class distance:

$$\arg\min_{P} \sum_{k=1}^{K} \sum_{v \in P_k} ||v - \mu_k||_2^2$$

where $\mu_k = \frac{1}{|P_k|} \sum_{v \in P_k} v$ is the mean of the vectors belonging to class k and $|| \cdot ||_2^2$ is the squared Euclidean distance which is calculated as follows for two vectors v and v':

$$\sum_{i=1}^{D} (v_i - v'_i)^2$$

where D refers to the dimensionality of the vectors.

In our case we seek to cluster users based on their behavior. To do so, we need to represent the data to a vector space model starting from the event log which consists of tuples of the form $< u, a, t >$ where:

- u is the user (or originator of an action)
- a is the executed action
- t is the timestamp of the action

To create the vectors we identify all different actions a_k, $k \in \mathbb{N}$ of the log. Then for each user u_j, $j \in \mathbb{N}$ we create a vector v_j, such that: $v_j =< n_{a_1}, ..., n_{a_k} >$, for $k \in \mathbb{N}$ and n_{a_i} is the number of actions executed by the user u_j.

4.2 Ontologies

In this section we describe how we build the ontology of our application. The goal is to incorporate knowledge extracted from ontologies into the clusters produced from the analysis of the event logs.

An ontology can express semantically the system concepts and provide a formal representation of the relations between them [12]. In most of the cases, ontologies are created manually by engineers and domain experts.

In our case, we aim to facilitate the task of experts by building an ontology semi-automatically. Analyzing the model extracted from the log analysis, we define rules that faciliate the extraction of the basic concepts of the ontology.

We start by giving some formal definitions about the ontology and the process model.

An ontology is a tuple $\{C, R, H\}$ where:

- C is the set of concepts
- R is the set of relations defined over concepts
- H is a directed acyclic graph over concepts defined by the subsumption relation $\leq_r$ between concepts, such that for concepts $C_1, C_2 \in C, C_2 \leq C_1$, the concept C_1 subsumes the concept C_2 with the property r.

The process model we obtain analyzing the even logs is a transition system S such that $S = (\Sigma, \Lambda, \rightarrow)$ where:

- Σ is a non-empty set of states
- Λ is the set of labels, one for each state of the system
- $\rightarrow$ is a transition relation $\rightarrow \subseteq \Sigma \times \Sigma$

The ontology we build consists of a tuple as described above.

We define C the set of concepts of our ontology such that $C = c_0 \cup C_1 \cup C_2 \cup C_r$ where:

- c_0 is defined as the root concept of the ontology. It correspond to the label $\lambda_0 \in \Lambda$ of the state s such that $\nexists s' \in S, s' \rightarrow s$ where $s \in S\}$. The root is defined by the *Business Process* concept.
- C_1 is the set of concepts that corresponds to 'independent' edges of the process model, i.e. edges that have no incoming or outgoing transitions. The set of concepts C_1 is defined as $C_1 = \{c_i \in C\}$, $i \in \mathbb{N}$ such that c_i correspond to the labels $\lambda_i \in \Lambda$ of the set of states $\{s_i \in S\}$ for which $\nexists s', s'' \in S | s_i \rightarrow s', s'' \rightarrow s_i\}$. Each concept c_i of C_1 is named after the label λ_i of the edge s_i.
- C_2 is the set of concepts that corresponds to 'popular' edges of the process model, i.e. edges broadly interconnected with other edges through incoming and outgoing transitions. The set of concepts C_2 is defined as $C_2 = \{c_j \in C\}, j \in \mathbb{N}$ such that c_j correspond to the labels $\lambda_j \in \Lambda$ of the set of states $\{s_j \in S\}$ where for $s', s'' \in S$ and for the set of transitions $\{\{t_{in}\}, \{t_{out}\}$ $|t_{out} : s_j \rightarrow s', t_{in} : s'' \rightarrow s_j\}$ it holds $|t_{in}| + |t_{out}| > e$ for a given threshold e. Each concept c_j of C_2 is named after the label λ_j of the edge s_j.
- C_r, for $r \in \mathbb{N}$, is the set of concepts that corresponds to all remaining edges. We perform lexical analysis on the labels of the edges such that for a label λ_r we split it in two parts, λ_{r_A} and λ_{r_B}. The first part λ_{r_A} corresponds to the first word of the label. For the set of concepts C_r we use the labels λ_{r_B}.

We define R the set of relations of our ontology such that $R = r_0 \cup r_i$ where:

- r_0 is defined as the property 'is based on'. The root concept c_0 is connected using this concept with the concepts of the sets C_1 and C_2 such that for $c \in C_1, C_2$, $c \leq_{r_0} c_0$.
- The set of properties r_i corresponds to λ_{i_A}. Concepts of the set C_2 are connected to concepts of the set C_r using properties r_i such that for $c_j \in C_2$ and $c_r \in C_r$, for $j, r \in \mathbb{N}$it holds that $c_r \leq_{\lambda_{i_A}} c_j$. Which concepts of C_2 will be connected with concepts of C_i is a manual and random process.

For the example we are studying, we extract the ontology depicted in Figure 3. To construct this ontology, we analyze the process model shown in Figure 1. We identify seven 'independent' and 'popular' edges shown at the second level from the top of the ontology graph. These edges are transformed to concepts and connected to the root 'BusinessProcess' with the property 'is based on'. The concepts of the third level are created based on the lexical analysis of the edges of process model. For example, the edge 'EnregisterAvenant' leads to the 'EnregistrerChargementPayeurContrat'. The edge 'EnregistrerChargementPayeurContrat' is transformed to the concept 'ChargementPayeurContrat' and connected to the concept 'EnregistrerAvenant' with the property 'enregistrer'.

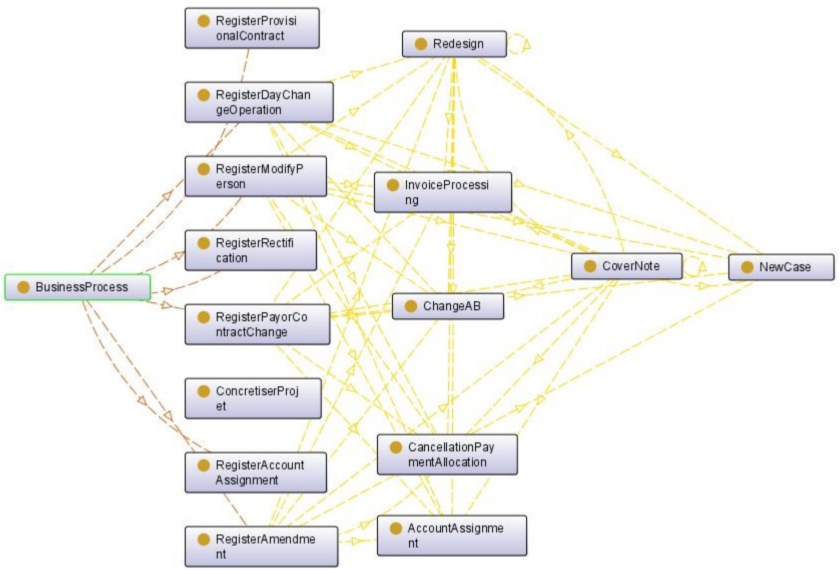

Fig. 3. The ontology

5 Results and Discussion

In this section we describe the results of our methodology as applied to the MMA case. The event log we used contains 670 process instances and a total of 24 activities and 459 users.

The vector of each instance is standarized in the range $[0, 1]$ feature-wised and it is normalized to unit norm. The k-means algorithm was applied using different values of the number of classes $k = \{2, 3, 4\}$. The minimum inter-class distance was achieved for $k = 4$ which was also evaluated with the mean Sihlouette Coefficient [9].

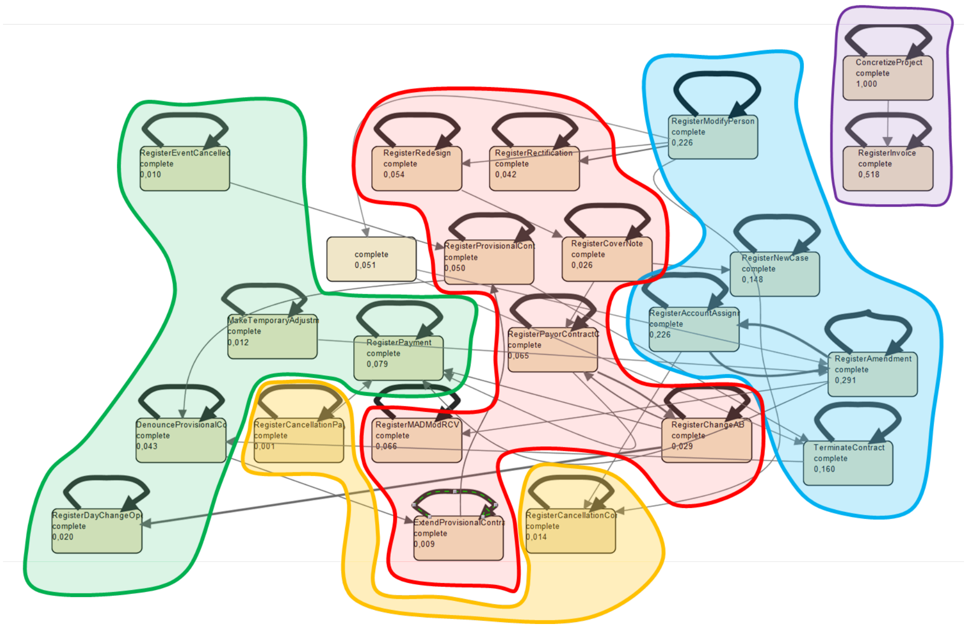

Fig. 4. The four clusters of the process model

Figure 4 shows the four different clusters that we obtain for the process model of Figure 1. The next step consists of enriching these clusters with semantic information, such that each cluster is identified by those actions that describe its process. Using the ontology of Figure3 we extract the events that are illustrated in Figure 5. Each of these events characterize semantically the clusters and give sufficient information for understanding their meaning.

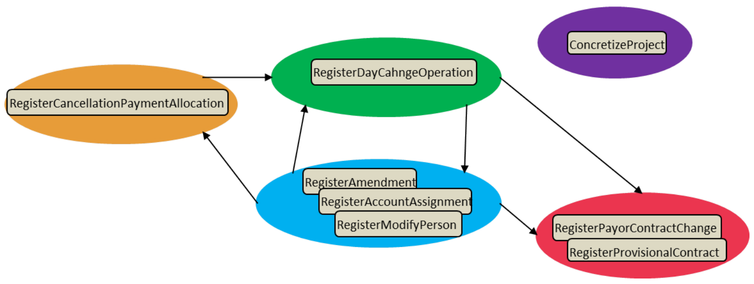
ConcretizeProject
RegisterDayCahngeOperation
RegisterCancellationPaymentAllocation
RegisterAmendment
RegisterAccountAssignment
RegisterModifyPerson
RegisterPayorContractChange
RegisterProvisionalContract

Fig. 5. Semantic Cluster

6 Conclusions

In this work we introduced a methodology for clustering the event logs and identifying the most important activities of each cluster. The goal is to present an understandable and easy to manipulate model to the end user. We used process mining algorithms to extract the process model from the event logs, clustering algorithms to divide the logs to subgroups based on common behavior of users and finally built ontologies to identify the most important activities and the relations between them.

The ontology we built can be used to identify the most important activities of each cluster. The second level of the constructed ontology contains the most significant activities of the process. These activities can be identified in the clusters and give the semantics of each of them. More details for each cluster can be obtained from the lowest levels of the ontology.

This methodology can be used to deal with incompleteness. The generated logs provide information about only a fraction of the process. By clustering users, we can complete the behavior of one user with actions identified at the behavior of other users of the same cluster. Similarly, we can identify erroneous behaviors, analyzing the actions of users of the same clusters and observing unusual sequence of actions. Another possibility is to predict the behavior of users based on the action followed by other users of its cluster.

Future work includes defining other parameters for clustering the log, different than the user behavior as well as incorporating multi-view clustering algorithms. For each cluster of logs, we can generate the corresponding process model and associate models between them by merging common elements. Improving the built ontology and expanding it with existing ontologies can improve the semantic interpretation of the clusters. Finally, we plan to apply the proposed methodology to larger experimental sets and validate the results by experts of the domain.

Acknowledgement

The authors would like to thank Dr. Josep Carmona for his valuable comments and suggestions for improving this work and the anonymous reviewer for his useful comments.

References

1. C. M. Bishop. *Pattern Recognition and Machine Learning (Information Science and Statistics).* Springer-Verlag New York, Inc., Secaucus, NJ, USA, 2006.
2. R. Bose, E. Verbeek, and W. van der Aalst. Discovering hierarchical process models using prom. In *Proceedings of the 3rd International Conference on Business Process Management.*

3. R. P. J. C. Bose and W. M. P. van der Aalst. Abstractions in process mining: A taxonomy of patterns. In *Business Process Management, 7th International Conference, BPM 2009, Ulm, Germany, September 8-10, 2009. Proceedings*, pages 159–175, 2009.
4. C. W. Gnther and A. Rozinat. Activity mining by global trace segmentation.
5. G. Greco, A. Guzzo, and L. Pontieri. Mining hierarchies of models: From abstract views to concrete specifications. In *Proceedings of the 3rd International Conference on Business Process Management*, BPM'05, pages 32–47, Berlin, Heidelberg, 2005. Springer-Verlag.
6. C. W. Günther and W. M. P. Van Der Aalst. Fuzzy mining: Adaptive process simplification based on multi-perspective metrics. In *Proceedings of the 5th International Conference on Business Process Management*, BPM'07, pages 328–343, Berlin, Heidelberg, 2007. Springer-Verlag.
7. C. Petri. *Kommunikation mit Automaten.* Schriften des Rheinisch-Westfälischen Institutes für Instrumentelle Mathematik an der Universität Bonn. Rhein.-Westfäl. Inst. f. Instrumentelle Mathematik an der Univ. Bonn, 1962.
8. Object Management Group. Business process model and notation (bpmn) version 2.0. Technical report, Object Management Group (OMG), jan 2011.
9. P. Rousseeuw. Silhouettes: A graphical aid to the interpretation and validation of cluster analysis. *J. Comput. Appl. Math.*, 20(1):53–65, Nov. 1987.
10. A.-W. Scheer and F. Habermann. Enterprise resource planning: Making erp a success. *Commun. ACM*, 43(4):57–61, Apr. 2000.
11. M. Song, C. Günther, and W. van der Aalst. Trace clustering in process mining. In D. Ardagna, M. Mecella, and J. Yang, editors, *Business Process Management Workshops*, volume 17 of *Lecture Notes in Business Information Processing*, pages 109–120. Springer Berlin Heidelberg, 2009.
12. S. Staab and R. Studer. *Handbook on Ontologies.* Springer Publishing Company, Incorporated, 2nd edition, 2009.
13. W. van der Aalst. Process mining: Making knowledge discovery process centric. *SIGKDD Explor. Newsl.*, 13(2):45–49, May 2012.
14. W. M. P. van der Aalst. *Process Mining: Discovery, Conformance and Enhancement of Business Processes.* Springer Publishing Company, Incorporated, 1st edition, 2011.
15. B. F. van Dongen, A. K. A. de Medeiros, H. M. W. Verbeek, A. J. M. M. Weijters, and W. M. P. van der Aalst. The prom framework: A new era in process mining tool support. In *Proceedings of the 26th International Conference on Applications and Theory of Petri Nets*, ICATPN'05, pages 444–454, Berlin, Heidelberg, 2005. Springer-Verlag.