End to end process discovery in process unaware systems

Sukriti Goel
SETLabs, Infosys Technologies Ltd., India
sukriti_goel@infosys.com

Abstract. Knowledge of end-to-end business processes is critical for an organization to undertake any business process initiative. Business processes in an organization typically span across many applications in different departments. Applications usually mature and develop over a period of time and with varying technologies. The process flow is achieved by integration and data hand offs between applications, systems and programs. This many a times creates a gap between the process as documented and the actual process executing in the systems. While there are multiple process discovery techniques available for process aware information system, the real challenge arises when the process is executed in a legacy scenario and/or across multiple applications and departments. The end to end process discovery approach we propose uses persisted process execution data and discovers processes as they are executed in multiple heterogeneous applications and departments.

Keywords: Process mining, Process extraction, Process discovery

1 Introduction

It is very important for an organization to understand the processes running across organization. Understanding the business processes the way they are being executed is the first step to improve and optimize the process. Any business process related initiative such as business process automation, business process optimization, business process outsourcing would need knowledge of already executing as-is process. The extent of knowledge of the process needed is dependent on the nature of the business process initiative chosen by the organization. A process automation or optimization initiative would require details on average cycle times of execution, process measures and SLA (service level agreement) performance in addition to the process flows while compliance reporting would focus on the controls implemented on the process and the effectiveness of these controls.

Most large organizations have their core business processes implemented across many applications, legacy systems, ERP systems and products. The processing logic and rules are coded in algorithms, batch jobs, database constraints etc. The process flow in such scenarios is achieved by the integration and data hand offs between

applications, systems and programs and hence is not explicit. A single end to end business processes is implemented across more several applications with many of them being process unaware systems [1].

There are multiple methods for documenting the process knowledge of an organization. Process elicitation is one such popularly used method [2]. This method has been used for years and is more dependent on the experience of the process consultant. This method is vulnerable to human interpretations, political influences and inadequate business knowledge.

Another method of extracting the process is to study the application code or to add probes in systems. This method not only time consuming but needs experts in the language in which applications are coded. This becomes a challenge in case the process is executed in a legacy system. An application may have code which is not executed currently but has been there as part of the legacy. While extracting the process from such a system, we will not get information of process as it is executed but the way it was coded. Adding probes can hamper system performance, and is risky as it involves changes, even if minor, in a production environment as well as significant effort in testing the applications. To get the good representation of the process as executed, the probes need be added for long time in the system, up to 3 to 6 months in some cases.

Other methods include multiple automated tools [3-6] to discover the process from the systems. All these systems have common limitation of not being able to discover the manual process but at the same time these tools claim to discover the process as executed in the system. These tools expect presence of event logs or depend on writing code for generating event logs as input to the tool to discover process. Once the event logs are available these automated process discovery tools use the popular process mining algorithms to discover the process.

We propose a Process Extraction method which can discover the multiple end to end processes executing in multiple heterogeneous systems in a non-intrusive manner. In this paper we will list down the challenges in end to end process discovery from process unaware systems, provide an overview of related work and other automated process discovery methods. Later we will discuss the proposed method for discovering the end to end process by tracking the process execution data. We use a case study to show how the process is discovered overcoming challenges of process unaware systems.

2 Process Discovery: Challenges

There are multiple challenges an organization face while trying to discover their end-to-end processes. A typical process is a combination of manual and automated tasks. The process can be structured where most of tasks follow a particular defined order or the process can be unstructured where tasks can be performed in any order. The process can span across multiple departments, roles and users. The automated part of the process can span across multiple heterogeneous systems. Some of these systems can be process aware systems, while some can be transactional systems like ERP, CRM. Some of these systems can be legacy systems or custom applications. The

information required about task execution of a process will be stored in various data formats within the systems, e.g, transactions tables, flat files, audit logs, log files, etc.

Some of the processes may have been running in the organization for years. Though they have been working well, there is usually little knowledge of the process available in the organization. This is the result of the technology being obsolete, people leaving the organization often and lack of documentation. Even when the process documents are available, they are not be updated as and when the systems and processes were changed.

There are some tools and techniques which can discover the processes as executed in systems. These tools and techniques are dependent on the availability of event logs. While process aware information systems generate the event logs as required by most of these tools and techniques, it is practically difficult to obtain event logs for the end-to-end process across the systems [18]. As discussed by Aalst in a workshop report, in systems that are not process aware, i.e., an explicit process model is missing, it may be more difficult to obtain the right logs. We can create event logs for individual systems and discover the multiple sub-processes but the hand-off of process across systems would not be known. There are multiple entities involved in the process, there is no single identifier flowing across the systems which can identify an individual case. At the same time, the data traces in each system are stored in different manner, adding to the problem of generating event logs. The unstructured part of the process is where the information is stored in a file or sent in an email. These are the typical challenges in creating event logs for end-to-end process.

Organizations are not only looking at discovering information about flow of tasks in a process but are equally interested in finding out run time information of the process. This includes process intelligence information, checking process model conformance of using the process execution data, looking for compliance while process is executing.

All these complexities of the process make the end-to-end discovery a challenge for organizations. In the next section we will examine solutions to some of these challenges proposed in literature.

3 Related Work

Business Provenance technology [9] helps trace end-to-end business operations across heterogeneous systems by collecting, co-relating and analyzing operational data. This approach involves creating a provenance graph for the specific application need based on which the provenance data is generated for extracting the required information about the business operations. The provenance data is generated by accessing application events from either event reporting middleware or by processing the application data and identifying the events. This approach expects the applications to generate the provenance data pre-defined format or the applications need to be instrumented.

Process elicitation is one method which can overcome most of the challenges mentioned in previous section. The drawback of process elicitation is mainly its dependence on business users who narrate the process and consultant who

understands and documents the process. This method needs significant amount of time of consultants and business users thus making it expensive and time consuming. Additionally, this method is vulnerable to human interpretations and political influences [2].

While process elicitation concentrates on interviewing business users, conducting workshops with the process stakeholders to discover the process, there are a few other manual techniques where the task execution data is collected while users are working on the tasks. The task execution data can then be used for manual or automated interpretation of the process based on volume of data collected. The first method of collecting task execution data expects the users to enter the instance id, task name, task start time and task end time for each of the task while working on it [16]. This method overloads the business user to capture extra information and it is not feasible if volume of tasks is high.

Another method of capturing events uses a hybrid method of Process observance and Process logging and recording. In the Process Observance method an observer is responsible for recording events of all kinds except automation events. The automated events are automatically derived from log files. The solution to correlate the manual and automated events of the same case was done on the assumption that each automation event interval for a given subsystem is related to the analysis event interval for that subsystem beginning soonest in time after the automation interval [17]. This method reduces the overhead on the business user to collect data but introduces extra by introducing the observers in the organization. This method is time consuming and is good for analyzing the process for purpose of process improvement initiatives like six-sigma but may turn out to be very expensive to provide process intelligence on a continuous basis.

The idea of Process Mining is not new [10]. Process mining techniques can discover the process as executed in presence of event logs [4,11-15]. The event logs are collected such that (i) each event refers to a task (i.e., a well-defined step in the workflow), (ii) each event refers to a case (i.e., a workflow instance), and (iii) events are totally ordered [12]. Though these process mining techniques have the drawback of not being able to discover the manual part of the process, they are gaining popularity, as in most organizations, the processes are automated.

The event logs are available in process aware applications such as BPM, ERP systems while these event logs need to be created in case of process unaware applications. Though one can write the code to generate the event logs from a single application, it becomes difficult to write code for generating event logs where the process is being executed across multiple applications and deals with multiple entities.

ProMImport [8] is a framework available which can convert logs in any format into event logs as required by many of the process mining techniques. The framework takes data from Process Aware Information System (PAIS) such as WPS, Staffware, FLOWer, audit trails, ERP systems to generate event logs. Though the framework is extensible to add more input formats, it does not provide suggestions on generating the event logs from process executed in process unaware systems and process executed across multiple heterogeneous systems.

4 Process Extraction (PE) Approach Overview

As multiple robust process mining algorithms are already available [6, 11-16], the proposed method concentrates on the generation of event logs in case the audit trails are not available in the system or when the process spans across multiple systems. Once the event log is generated, the event log is used as input to any of the process mining tools to discover the process. The event log may have enough information to find out the process flow, social networking graphs, business intelligence.

4.1 Detailed Approach

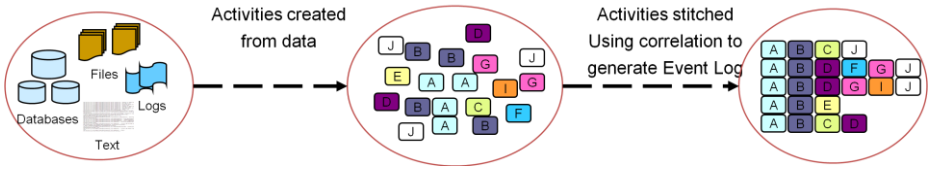

Figure 1: Process Extraction Approach

There are various manners in which an organization stores data for their applications and processes. Some of the processes store information in the form of transaction data in the RDBMS, while others store the audit logs in a file or RDBMS. Some processes store the information in form of log entries, some in the form of flat files as in case of mainframe legacy systems and some information is stored in the form of scanned documents. While the process is executing, the various applications and middleware create logs such as application server logs, these logs contain low granularity details about the execution. As the process executes across multiple systems, there may not be a single way to store process execution data in a simple format.

Process may work on multiple entities thus leading to multiple case identifiers. For example in order management system, the entities involved may be “order”, “item”, “invoice” and “shipment” thus multiple identifiers such as “order id”, “item id”, “invoice id” and “shipment id”.

As the application executes, the minimum data which is stored in applications is the case identifier and the date when the data was stored [7]. This is needed to distinguish the cases from each other. Some applications even store information in the form of audit logs, where each action taken on an entity is recorded with case id and timestamp.

Any information system using transactional systems such as ERP, CRM, B2B, SCM and WFM systems will offer information about the order in which the events of a case are executed [10].

The presence of audit logs makes it easy to generate the event logs only if one entity is involved in the process. Some processes work with two or entities simultaneously, e.g. account creation process may involve creation of a customer as well as creation of an account thus deals with two entities e.g. customer and account. In such a scenario, even if the audit log is available for each entity, it would be difficult to generate an event log which would cover the superset of events occurring while process execution.

The completeness of discovered process is dependent on the creation of data traces during process execution. This drawback can be overcome by using multiple data sources for extracting the same process, e.g. saved transaction data, audit trails wherever available, logs written by applications, middleware logs, etc. In the absence of audit trail and application logs, the process discovered is less complete and at a higher granularity. In one experiment, we realized that for the same process the completeness of process as discovered from audit logs is almost 40% more from the process discovered using transaction data. The result of experiment is shown in Figure 2, 3.

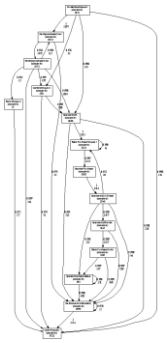

Figure 2: Discovered Process Using Audit logs

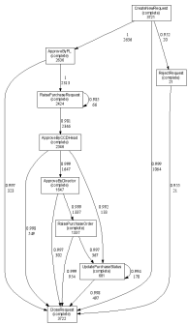

Figure 3: Discovered Process Using Transaction Data

This method is agnostic to technology and data storage mechanism used in each of the systems. The method has following three stages for generating the event logs and thus mining the process.

4.1.1 Configuration

The objective of the setup is to find the data traces left by the process during execution. To achieve this it is necessary to identify the systems and entities involved in the process. The typical data traces left by process execution is inserting a row in a DB table, updating a row in DB table, creating a new file, adding a new log entry in a file, sending a mail, receiving a mail, and so on. Each of the updated and created data trace is a candidate event.

Process execution data from all systems is studied to identify candidate events. The identified events are mapped to the business activities. Typical events which are mapped are process start and end, activity start, activity end based on the process data. Each activity is also assigned a correlation id.

4.1.2 Generating event logs

Once the setup is ready, the events are read from all the identified data sources attached to all the systems where the process is executed. Each event is mapped to a business activity as per the defined rules. This creates a cloud of unrelated activities. These activities need to be correlated to create event logs. In case there is a single entity involved or transaction id is available, the activities are correlated using the entity id or transaction id. The challenge is when there are multiple entities involved in the process and transaction is not available. In this case, the activities are first correlated using individual entity ids leading to smaller sub processes. Then these sub processes are correlated to each other to create event log of end to end process using the relation between entities. To illustrate this let's take an example of a process which involves 2 entities order and invoice. For each order, an invoice is generated and this relation is maintained in the transaction data. In this example, transaction data stores that for order id "ord1" the related invoice id is "inv100". Once the activity clound is available, all the activities with identifier "odr1" are correlated and in the same manner all the activities related to "inv100" are correlated to generate two separate event logs for sub-processes. Given the relation that "inv100" is related to "ord1", the event log related to "ord1" and "inv100" are correlated and sorted in the timestamp to generate the event log for the end-to-end process. This leads to the generation of event logs as required by most process mining algorithms.

4.1.3 Process Mining Algorithm

There are a large number of mining algorithms available in existing process mining tools. The event logs generated using the proposed approach can be used as input to most of the mining algorithms thus discovering the end to end process.

5 Case study: Order to Cash Process

We would consider an illustrative Order to Cash process to explain the challenges in process discovery and how some of those can be solved with the proposed approach. The typical Order to Cash process touches multiple departments, companies and back-end applications. The process may be split in smaller sub processes such as quotation to order, order-to-shipping, shipping to delivery, order to invoice and invoice to payment. Each of these sub processes may be handled in separate applications. The entities involved in order to cash process are a) customers, b) order, c) invoice, d) delivery e) accounts and ledgers, f) payment, etc.

Let's look at an Order to Cash process of a typical organization in detail. The process begins with a sales inquiry captured in a sales document in the sales support

system. The inquiry may result in a sale leading to order creation in the order processing system. At the delivery processing system the delivery is created for each order, picked, and goods issue is posted. In the billing system, an invoice is created and released to the financial system and also sent to the customer. Incoming payments are documented in the payment processing system and then posted in the financial system.

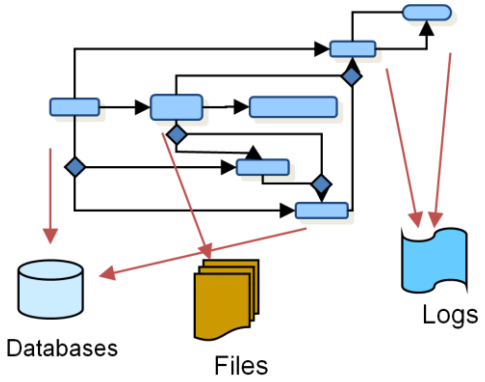

Figure 4: Order to Cash Process

The typical organization has multiple systems where the process is executing. Each of these systems is maintained by different departments and has been created at different times thus uses different technologies. As the need for storing the data is also different for each system, the data traces created while executing the process are different in nature (Figure 4). While the Sales support system captures the information in form of a text file, the financial system and payment system store audit logs. The invoice system stores the transaction data with timestamp of all the actions happening in the system in transaction tables. The order processing system captures the data in transaction tables about creation of the order and then the final status of the order.

Business Challenges: As the process executes across various departments and systems, it is not clear how the hand-offs are happening in the process. Individual departments and system owners know how the related sub-process is executed but no one knows the end-to-end process model. The organization is looking for implementing a SAP system and hence wants to know the process as it is executed currently. The aggregation of the documentation of individual systems is not able to give the complete picture of the process. Even the discussion among the system owners and business users is providing partial information about the process.

Approach: We can use the proposed approach to generate the event logs for the end-to-end process. PE approach uses the data traces left by process execution in different systems. The first step in the method is to identify the entities involved in the process and systems which process those entities. In current case the entities and systems of the process can be identified by brief discussion with business users and IT

staff. Next step is to identify the various data sources for each system. This task is done with help of the IT staff.

The approach further involves identifying the event in each change in data in the data source resulting from execution of the process. For example, when a new sales opportunity is created, a new file is inserted in the document management system. In the same way, when a new payment is received a new audit entry is made in payment systems as well as the new entry is made in the audit logs of the financial system. For invoice processing and order processing systems, a row is inserted in the invoice table when a new invoice is created and a new row is inserted in order table when a new order is created. Further, when the order processing is completed, the corresponding row in the order table is updated with timestamp of order completion.

The PE approach further suggests identification of the correlation identifier for each event, e.g. sales opportunity id is identifier for any sales opportunity entered in the sales support system while order id is the identifier for order in order processing system. This approach assumes that there is some relation between sales opportunity id and order id and it is stored as entity relation in the systems for establishing traceability. The relationship between each entity is also identified.

Further the method and algorithms reads all the events which happen in the various data sources by appropriately querying the data sources. The algorithm maps these events to the business activity, e.g. event of creation of new file in sales support system maps to completion of 'create sales opportunity' activity. In the same way, event of inserting a new record in order table in order processing system maps to completion of 'create order' activity.

The algorithm correlates these activities executed in various systems by identifying activities related to same case, which further sorts the activities of a case using timestamp to generate the event logs as required by most process mining algorithms.

Further the ProM tool can be used to apply process mining algorithms to the event logs to discover the process as executed in the systems.

We can discover 70-90% of the process in terms of completeness as some parts of the process maybe manual which is not recorded in the system. Some parts of the process could not be discovered as not some of the process execution data is lost either because the data is overwritten or in some case the data is written temporarily and deleted immediately after the execution is complete. This can happen in case of order table where status column gets overwritten and final status is always 'order completed', and the intermediate status e.g. 'invoice generated', 'payment received' are not captured as we used the historical data to discover the process and the orders which we looked at were already completed.

6 Conclusions and Further Work

Automation-assisted process discovery holds a lot of promise today. It offers the capability of finding out the process as executed in the systems, not as perceived by the users. It uses the historical data to discover the process hence does not interfere with the execution of process. The process discovery is done from persistent process

execution data available in various data sources. While all business transaction data which is of any relevance is usually logged for further processing, a limitation of the current approach is that if a business activity does not leave a trace in persisted data, it cannot be discovered: manual activities and the calculations done in system memory fall within this category.

In addition, the availability of process information required for Business Process Intelligence (BPI) and process optimization depends on the quality of data traces. Usually the data traces are not stored at the start of an activity, so it may be difficult to generate information like average time to complete an activity. The mining algorithms do not mine the process flow rules along with the discovered process.

The area of subsequent research is to improve the quality of event logs so that it can generate enough process information as required by process optimization and BPR initiative. Another area is to explore is on mining of the business flow rules along with discovering process using the data traces as available in the systems.

7 References

1. Leymann F., Reisig W., Thatte S.R., and van der Aalst W.M.P., The Role of Business Processes in Service Oriented Architectures, number 6291, Dagstuhl Seminar Proceedings. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, (July 2006)
2. Verner L.; The Challenge of Process Discovery; BP Trends; May 2004
3. B. van Dongen, A.K. Alves 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 G. Ciardo and P. Darondeau, editors, *Application and Theory of Petri Nets 2005*, volume 3536 of *Lecture Notes in Computer Science*, pages 444-454. Springer-Verlag, Berlin, 2005.
4. Aalst, W., van der ;Process Discovery Technology: a Comment on Paul Harmon's June 10 advisor, BPTrends - July 01, 2008Harmon P., Fujitsu's Process Discovery Technology; BPTrends Email Advisor, Volume 6, Number 11, June 10, 2008
5. Case Handling with FLOWer: Beyond workflow. Pallas Athena BV, Apeldoorn, The Netherlands, 2002.
6. Aalst, W. van der, Weijters, A., Maruster, L.: Workflow Mining: Discovering Process Models from Event Logs. IEEE Transactions on Knowledge and Data Engineering 16(9),1128–1142 (2004)
7. Woodfill, J. and Stonebraker, M.: An Implementation of Hypothetical Relations. In Proceedings of the 9th international Conference on Very Large Data Bases (October 31 - November 02, 1983). M. Schkolnick and C. Thanos, Eds. Very Large Data Bases. Morgan Kaufmann Publishers, San Francisco, CA, 157-166. (1983)
8. C. Guenther and W.M.P. van der Aalst. A Generic Import Framework for Process Event Logs. In J. Eder and S. Dustdar, editors, Business Process Management Workshops, Workshop on Business Process Intelligence (BPI 2006), volume 4103 of Lecture Notes in Computer *Science*, pages 81-92. Springer-Verlag, Berlin, 2006.
9. Curbera, F., Doganata, Y., Martens, A., Mukhi, N. K., Slominski, A.: Business Provenance – A Technology to Increase Traceability of End-to-End Operations. R. Meersman and Z. Tari (Eds.): OTM 2008, Part I, LNCS 5331, pp. 100–119, 2008.
10. 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.W.M.P. van der Aalst, A.J.M.M. Weijters,

and L. Maruster. Workflow Mining: Discovering Process Models from Event Logs. IEEE Transactions on Knowledge and Data Engineering, 16(9):1128–1142, 2004.
11. J.E. Cook and A.L. Wolf. Discovering Models of Software Processes from Event-Based Data. ACM Transactions on Software Engineering and Methodology,7(3):215–249, 1998.
12. Alves, A.K., Using Genetic Algorithms to Mine ProcessModels: Representation, Operators and Results. 2003.
13. B.F. van Dongen and W.M.P. van der Aalst. Multi-Phase Process Mining: Building Instance Graphs. In P. Atzeni,W. Chu, H. Lu, S. Zhou, and T.W. Ling, editors, International Conference on Conceptual Modeling (ER 2004), volume 3288 of Lecture Notes in Computer Science, pages 362–376. Springer-Verlag, Berlin, 2004.
14. R. Agrawal, D. Gunopulos, and F. Leymann. Mining Process Models from Workflow Logs. In Sixth International Conference on Extending Database Technology, pages 469–483, 1998.
15. Stijn Goedertier, David Martens, Jan Vanthienen, Bart Baesens; Robust Process Discovery with Artificial Negative Events The Journal of Machine Learning Research , MIT Press, Volume 10, Dec 2009
16. Victor R. Basili and David M. Weiss. A methodology for collecting valid software engineering data. IEEE Transactions on Software Engineering, SE-10(6):728-738, November 1984
17. Alexander L. Wolf and David S. Rosenblum "A Study in Software Process Capture and Analysis". 2nd International Conference on the Software Process, Berlin, Germany, February 1993.
18. W.M.P. van der Aalst. Process Mining and Monitoring Processes and Services: Workshop Report. In F. Leymann, W. Reisig, S.R. Thatte, and W.M.P. van der Aalst, editors, *The Role of Business Processes in Service Oriented Architectures*, number 6291 in Dagstuhl Seminar Proceedings. Internationales Begegnungs- und Forschungszentrum fuer Informatik (IBFI), Schloss Dagstuhl, Germany, July 2006.