BABEŞ-BOLYAI UNIVERSITY
FACULTY OF ECONOMICS AND BUSINESS ADMINISTRATION
BUSINESS INFORMATION SYSTEMS DEPARTMENT

PhD thesis:

MODELING ECONOMIC DECISION MAKING PROCESSES USING INTELLIGENT MINING TOOLS

-summary-

PhD Advisor:
Prof. Nicolae **TOMAI**, PhD

PhD Student:
Cristina-Claudia **DOLEAN**

Cluj-Napoca, 2013

Thesis contents

Contents

Abstract

Process-aware information systems (PAISs, e.g.: Workflow Management Systems, Business Process Management systems (BPMS)) provide event logs which represent the recording actions of users within the information systems. Each event log stores information about the resource executing the task, the timestamp when the task have been started/completed, or data elements recorded within the event (e.g. the name of a new client of a bank). On those logs, process mining techniques can be applied and some control-flow models are discovered. Therefore, the goal of process mining is to extract information about the process from these (event) logs. The most known and simple algorithm which offer a control-flow perspective of the process is the α-algorithm. It gets as input an event log and it provides a process model called Workflow net (WF-net). There are many other algorithms that aim to extract models from event logs, namely the Heuristics Miner, Genetic Miner, Fuzzy Miner, etc. However, the most part of mining algorithms provide a control-flow view of the process, ignoring the data-flow.

As we argued previously, the literature provides a lot of research concerning the control-flow aspects of a process, the data-flow perspective being ignored or at most integrated with the control-flow. The control-flow presents the order of process' activities, but ignores the flow of data. Some data may not be available on time in order to execute a certain task and the workflow stops its execution. This problem was already researched. The process may be improved by analyzing just the data-flow or both models: data-flow model and control-flow model.

The problem that we identified with this previous research is that none of the proposed methods and techniques doesn't have as starting point the analysis of data related events from the log (the main component of process mining). The literature offers a series of data-flow based analysis of a process. With respect to the analysis of the process data-flow, researchers tried to provide a standalone data-flow view or they combined the control-flow with data-flow. In this context, data validation methods are provided and some data-flow errors that can occur in the process' execution are identified. There exists a model that focuses on data movements through a process: the Product Data Model (PDM). The drawback of this approach is that no automatic method is proposed in order to generate the PDM (only a manual one). This thesis tries to fill this gap by providing automated and semi-automated methods to provide a data-centric visualization of a process.

We propose three data-centric mining algorithms in order to extract the data-flow perspective of a process. But the event logs used must comply a certain format. Therefore, we defined an extension and we developed two conversion tools. For the validation we used event logs provided by an ERP system (*Navision*) and by a WfMS (*YAWL*). In each case we compared the results got using the data-centric mining algorithms with some process models focusing on the control-flow perspective.

Keywords: Product Data Model, workflow, control-flow, data-flow, mining algorithms, PAIS, input data elements, output data elements, operations

Introduction

Process mining domain provides a series of techniques to analyze the process models hidden in the "footprints" left by users within information systems. These footprints are called event logs, and the information systems able to generate event logs by integrating applications, resources and processes are called Process-Aware Information Systems (PAISs). There are three types of process mining: discovery, conformance and enhancement. Discovery techniques refer to the extraction of process models from event logs. Beside the discovery part, process mining analyzes the deviations between the intended behavior of a process and the real behavior of a process. Here we speak about conformance checking. The last component of process mining refers to the improvement of the process based on the existing information from event logs.

Nowadays, most of companies are using information systems in order to manage their own business (e.g. ERP systems). ERP systems are designed to manage all data related logistics, human resources, production and sales. Moreover ERP systems are able to share data across different departments. Generally the employees from a company know the internal processes. But after the execution of a particular number of activities, the arising question is what data we need in order to continue our process. Moreover, being in a certain state of a process (after the execution of certain number of activities), the question asked by the employees is "what data is available and can/must be used in the future activities of the process"; or "what data do we still need in order to execute a particular activity". The control-flow perspective doesn't have the ability to provide these answers. Being in a certain state of a process' execution, it may only show the activities which can/must be executed, but it cannot ensure the execution of the activities.

The motivation for this thesis is to offer more insights about the data perspective of process models. Why do we want to analyze these aspects? *Firstly*, the existing process mining techniques focus on the control-flow aspects of a process and the data-flow perspective is ignored or at most integrated with the control-flow. The control-flow presents the order of process' activities, but ignores data movements within the process. *Secondly*, the data-flow perspective may bring improvements to the process (e.g. if the data-flow model shows us an operation with low frequency we may verify if that operation is really executed a few times or it represents a deviation of the intended behavior of the process). *Thirdly*, assuming we are executing a process and we are in a particular state of its execution we can identify, based on the data known until the current state of the process which operations can we perform further. Moreover a data-flow model of the process shows us what additional data we need in order to execute a particular operation. *Fourthly*, none of the proposed methods and techniques don't have as starting point the analysis of data related events from the log (the main component of process mining). There is an approach which offers the data-flow perspective of a process, but no automatic method is proposed to generate it (only a manual one). Moreover, an analysis of data dependencies also exists but neither this does not provide an automatic extraction of the data-flow. Usually, researchers considered the data-flow already created by experts or pulled out by analyzing the semantics.

Figure 1 offers an overview of the thesis' structure. It focuses on the main aspects of each chapter. First three chapters provide an analysis of the research done in the adjacent domains of modeling (Business Process Design, Systems' Design and Process Mining).

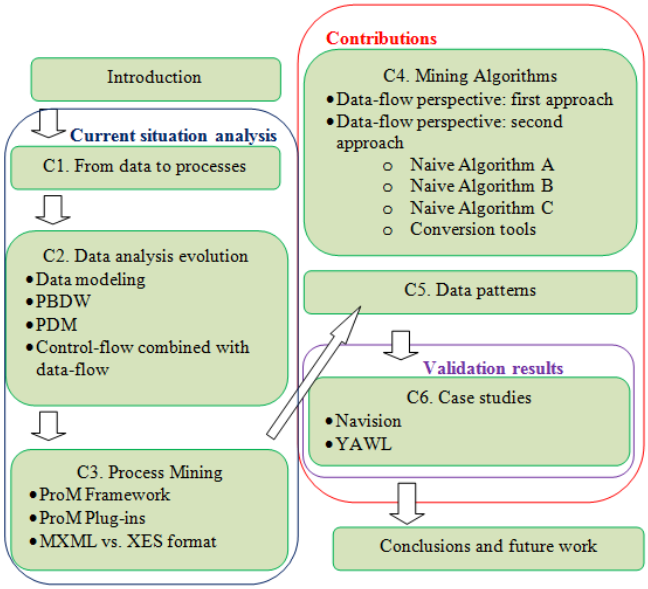

Figure 1 Thesis overview

First chapter depicts how information systems changed from data-oriented to process-oriented. This led to the emergence of Workflow Management Systems (WfMS). Firstly, we review the workflow dimensions. Then, we go through Workflow patterns, in order to emphasize the importance of data in process' execution. In the end, we approach control-flow perspective and data-flow perspective.

Chapter 2 presents the state of the art concerning the existing methods of data analysis. We started by presenting basic data modeling techniques like Entity Relationship Diagram (ERD) [12]. Then, we review some techniques that focus on control-flow perspective like UML Activity Diagram. The literature shows the existence of mixture techniques that bring together data-flow perspective with control-flow perspective (e.g. Process Data Diagram). This chapter also introduces one of the main notions of the thesis: the Product Data Model (PDM), defined in [33].

Chapter 3 makes an introduction to the process mining domain. Process mining exploits event logs of real processes and uses these in order to discover models. Due to the fact that each information system generates event logs in its own manner, some standard formats like MXML (Mining Extensible Markup Language) and XES (eXtensible Event Stream) were defined. Hence, a mining tool that supports several process mining techniques unified under a standardized format (MXML or XES) and offers support in event logs analysis was developed – ProM Framework.

Fourth, fifth and sixth chapters emphasize the contributions of the thesis. Chapter 4 proposes the implemented algorithms and their validation on synthetic event logs, while Chapter 6 validates our assumptions from Chapter 4 by applying the naive algorithms to various event logs. We introduce next a short presentation of these chapters.

Chapter 4 is the main formal contribution of this thesis, and focuses on our approaches related to the discovering of data-flow perspective. Here, we introduce the implemented

mining algorithms that extract the data-flow perspective of a process. We started with the event logs provided by a decision-aware information system (DAIS), and then we presented three naive algorithms that provide the data-flow perspective of a process. The functionalities of each algorithm are described in this chapter. *Chapter 5* proposes a series of data-patterns (e.g. basic data patterns, complex data patterns and data values data patterns).

Chapter 6 validates the proposed mining algorithms. This is based on two case studies. For the first case study we used the event logs produced by an ERP (*Navision*), while for the second one we used the event logs generated by a PAIS (*YAWL* system). In order to convert the relational database records from Navision or the event logs from YAWL to the desired format we used XESame 1.3 and another two conversion tools. Considering the event logs generated by YAWL, the extraction of PDM elements was possible by analyzing the *start* and *complete* events. Then for each case study we applied first two naive algorithms and we compared the results. Moreover, we compared our models with a control-flow perspective model like the model provided by Alpha Miner.

Finally, last chapter concludes the thesis and proposes the future work.

1. From data to processes

First chapter introduces the nature of information systems from data-centric in the '70s to process-centric latterly. Then we made the transition to the automation of processes; here we reminded workflow management systems (WfMSs). We discuss about two commercial tools provided by Software AG, respectively TIBCO. We also remind one open-source WfMS available on the market; in this direction we depict YAWL. One of the most important aspects treated in this chapter introduces the data-flow perspective of a process.

The interest for processes and business process reengineering appeared in the beginning of 1990s. Here we speak about process-driven approaches. Hammer and Champy [19] define the process like being "*a collection of activities that takes one or more kinds of input and creates an output that is of value to customer*". If the process is automated, by specifying the information and the tasks needed for each activity with respect to a set of rules, it becomes a workflow. This led to the development of Workflow Management Systems (WfMS). *A workflow management system (WfMS) is a generic software tool which allows for the definition, execution, registration and control of workflows* [11]. There are several workflow management systems available on the market (e.g. Staffware, Aris, jBoss, WjMOpen, jBMP, etc.). ARIS Platform provides a series of tools that help to process' analysis (e.g. ARIS Business Designer - for modeling, ARIS Business Architect – for reports, ARIS simulation – for model's simulation, etc.). ARIS Business Server stores a *central database server (e.g. Oracle) used for data administration. TIBCO Staffware provides from over 15 years a BPM complete* solution for organizations. TIBCO acquired Staffware since 2004, therefore TIBCO Staffware Process Suite is the result of this joint. It ensures the integration with the existing IT infrastructure and applications from a company. YAWL (Yet Another Workflow Language) is a workflow system based on Petri nets [25] that supports the execution of workflows. In this respect several workflow patterns [23] are utilized. YAWL is open-source software developed by researchers from University of Technology, Eindhoven, the Netherlands and University of

Technology, Queensland, Australia and it allows the modeling of the tasks within a process. It also offers the possibility to add data variables to the tasks and to define the resources which are executing the tasks. WfM emphasis the control-flow of processes and it is based on Petri Nets, while BPM adds to this the coordination of processes.

Workflow Management Coalition proposes three dimensions of workflows. The control-flow/process dimension offers a view of the entire process by analyzing the activities involved in, while the resource perspective distributes activities to specific resources. The case dimension presents information related to one execution of a certain workflow (business process). Control-flow perspective analyzes the order of the tasks, which task must be executed and when. Tasks are executed by resources with certain roles. The case dimension is formed by the process instances (a series of tasks).

In the last decade a major importance has been given to the control-flow perspective of the processes and analysis of the data-flow perspective was almost ignored. But data is needed in order to execute the tasks from a process (model).

Figure 2 partially depicts the activities from a hotel: from client registration to client's check-out. We will focus on the first two activities of the process. If we look to control-flow of the process we observe the order of tasks' execution (e.g. *Register client*, *Room allocation*, etc.). Each activity needs to consume data (input data elements) in order to be executed and in turn it produces data (output data elements). In our example the execution of *Register client* activity is possible if all textbox is fulfilled by the hotel's receptionist (*first name, last name, title, street, city, postal code, country, phone* and *e-mail*). These insights are not emphasized by the control-flow perspective.

Further this data is submitted to next activity (*Room allocation*). Even the last activity of the process (*Register check-out*) need the data about the client (the data elements from the first activity). The second activity also need data in order to be executed. After all data is available (*room number, room type, number of nights, number of adults, number of children* and *notes*).

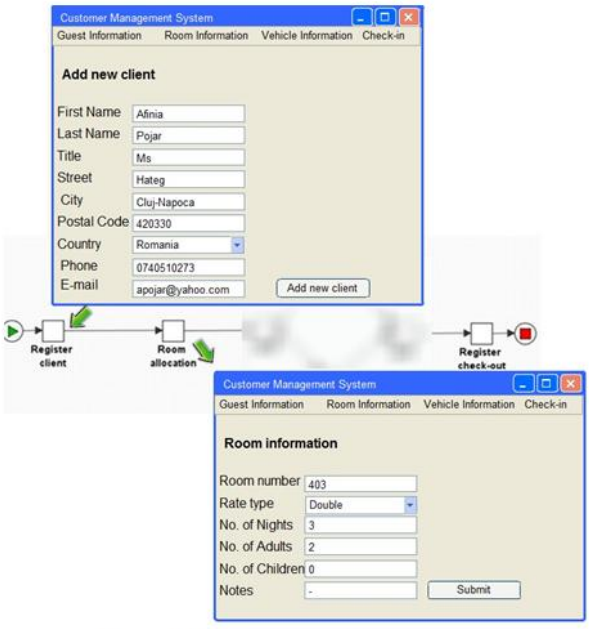

Figure 2 Data behind activities

Thus data moves through a process, from one activity one another. Data elements produced by an activity may be used by all activities from the process. Here if we refer to processes with lots of activities that consume and produce several data elements there exists a complex data movement through a process.

Next we will introduce information related to workflow patterns. There are several types of workflow patterns (e.g. control-flow data patterns, data-flow patterns, resource patterns, exceptions handling patterns), we will focus data-flow patterns. Data perspective classifies data into four categories:

a) *data visibility* (expresses the visibility of data elements in a process),

b) *data interaction* (depicts the way how data elements are communicated to process components),

c) *data transfer* (takes into consideration how data moves through the elements of the process), respectively

d) *data-based routing* (refers to the way how data elements have impact to the control-flow perspective).

Conclusions

This chapter presents the changing brought on information-systems during time. If at the beginning of 1970s the information-systems' focus was related to data (e.g. relational databases), the 1990s opens the demand to process-centric approaches (e.g. workflow management systems).

Business processes depict the way how a company organizes the activities in order to produce valuable products or services. The organizational performance may be improved using business process reengineering. The automation of business processes makes an organization more efficient. So, workflow management systems help to the simultaneous execution of activities by decreasing the entire duration of a business process.

WfMSs propose to treat case-driven business processes in order to analyze different perspectives: control-flow perspective, data perspective, respectively resource perspective. Last decade was flooded by analysis of control-flow perspective. Aspects related to resource perspective were also handled. But, a minor importance was given to data-flow perspective. In this chapter we introduced general aspects related to data-flow perspective and detailed insights will be provided in the next chapter.

2. Data Analysis evolution

An important part of the data analysis evolution study is published in [26]. It presents a survey of previous research on modeling the data-flow perspective of business processes. We reached the conclusion that none of the analyzed methods, techniques or models do not focus on the data changes during a process' execution.

Figure 3 illustrates the interest shown on data analysis during time. Data analysis started with the study of Data Structured Diagrams in 1969. Not long after this, the concept of Entity Relationship Diagram (ERD) [12] was defined. Then the process-aware systems made their

presence felt and new approaches focused on data have been developed. In order to analyze the data-flow of a process, researchers tried to provide a standalone data-flow view or they combined the control-flow with data-flow. In this context, data validation methods are provided and some data-flow errors that can occur in the data-flow are identified.

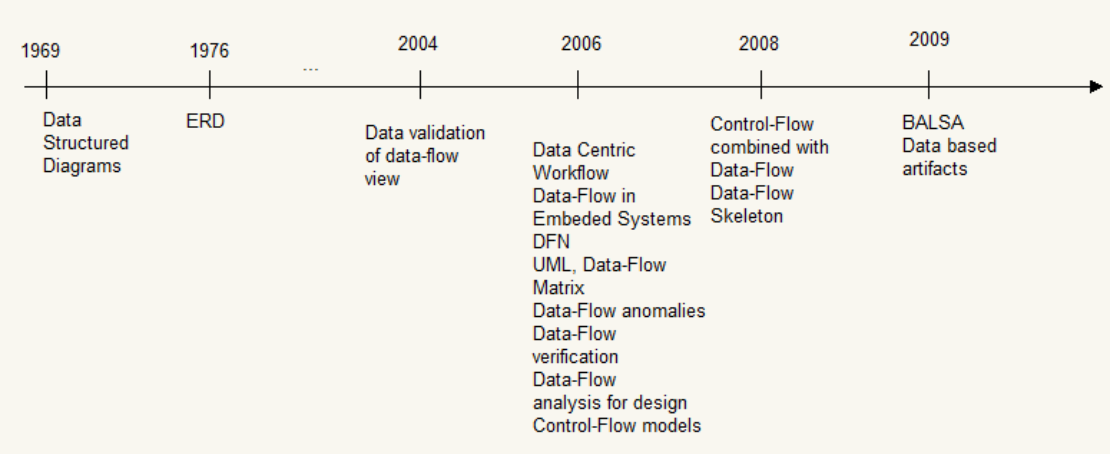

Figure 3 Data analysis evolution

Data modeling refers to the analysis of data-oriented structures (e.g. relational databases). First of all, in relational databases design, the model widely known and used for depicting the data elements and their interaction is the Entity Relationship (ER) diagram [12]. Such a model depicts the *entities*, their *attributes* and the *relationships* between entities. ERDs depict a static data model; meanwhile we want to dynamically model the data-flow perspective of a (business) process. Therefore, it is obvious that creating an ER diagram from a multiple-trace log and using it in connection to process models is unfeasible.

Next we made connections how an UML Activity Diagram may represent the data perspective. But this one offers insights about the order of activities from a process. Therefore, using those diagrams in the workflow context is unfeasible.

A technique which combines UML Activity Diagram with a data-based approach is depicted in [31]. First, the UML Activity Diagram is extracted, and then the data elements are separated in two categories based on the activity they belong: input data elements if they are read within the current activity and output data elements if they are written within the current activity. Data elements help to the development of the data-flow matrix. Then this matrix is integrated into the control-flow perspective (represented by the UML Activity Diagram) and Process Data Diagram results. The issue of this approach is that no automated method is proposed in order to provide the data-flow perspective. Moreover, no pure data model provided because the data-flow perspective is combined with the control-flow perspective. An approach closer to the Process Data Diagram depicted below is the Data-Flow Skeleton Filled with Activities (DFSFA) [15]. The workflow process is derived from the data-flow skeleton and then is filled with activities. First, a data-flow dependency tree is generated based on the data dependency. Next step is to generate the data-flow skeleton and then fills it with activities. The main difference to our approach is that we take into consideration event logs generated by information system and automatically produce the data dependency by classifying the data elements into input and/or output data items, while

in [15] and in [30], the data dependency is pulled out by analyzing the semantics. In other words, the input and output data are known for the process designer from the beginning.

One of the most known UML diagram proposed at the beginning of '70s is the Data-flow Diagram (DFD). This diagram shows the processes generated by an information system by combining processes (activities) with external entities (resources) and data (data objects and data stores). The main issue is that a DFD only depicts the activities which are directly related to the data processing; therefore the activities that do not involve any data modification are not depicted in a DFD. Moreover, no pure data model is provided because a DFD combines resourced with activities and data.

Metagraphs [9], [10] represent another modeling technique which combines data elements with the tasks of workflows. Therefore it does not present a pure data-centric approach, but the real shortcoming appears when we are dealing with complex metagraph: they are difficult to be read, respectively to be analyzed.

Next we will present an approach which is closer to our approach: product-based workflow design. First we depict the Bill-of-Materials as it is defined in [22]. This term is generally used in manufacturing engineering and it has a tree-like structure. In [3] the definition of Bill-Of-Material has been extended with options and choices. This was the starting point in order to define the Product Data Model (PDM) [33]. A PDM provides a data-centric visualization of a workflow (process – see Figure 4). The main issue of this approach is that no automatic method is proposed for PDM extraction. Because manually activity composition requires a lot of time even for specialized persons, an activity composition in the context of Product Based Workflow Design is introduced in [1]. Moreover, the algorithm that allows the automatic generation of the activities for a given PDM is implemented in ProM Framework. Having the XML representation of the PDM, the XSD file may be imported in ProM Framework. The XML definition consists of the set of data elements, respectively the set of operations. The algorithm returns the set of activities based on the set of input elements, the set of output elements, respectively the set of operations. Moreover it returns another visualization of the initial PDM taking into consideration the most important data elements of the initial PDM. The common point to our approach is given by the employment of ProM Framework in order to provide the data-centric visualization; while the main difference is that our approach proposes for each activity from the process (an activity from a process represents an event from the resulted event log) a single operation into to the corresponding PDM.

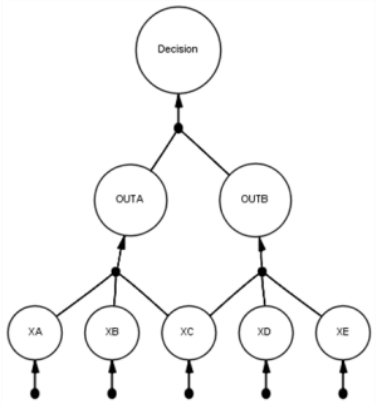

Figure 4 Product Data Model Structure

Next we will introduce some methods and techniques used to control-flow and data-flow verification. Even if a pure data model is not proposed, some data-flow verification techniques are defined [28]: redundant data, lost data, missing data, mismatched data, inconsistent data, misdirected data and insufficient data.

There are several detection methods and techniques designed to discover control-flow errors in workflow designs [5], [6], [29] and implemented in tools (Woflan [7], [34], [36]), while the data-flow verification assumes a detailed analysis of data dependencies [32]. Deadlocks, livelocks, soundness, interminable looping, and synchronization belong to control-flow verification area. Moreover, a series of anti-patters concerning data-flow are introduced in [32]: missing data, strongly redundant data, weakly redundant data, strongly lost data, weakly lost data, inconsistent data, never destroyed data, twice destroyed and not deleted on time.

The literature also provides a discovery algorithm of dataflow errors mentioned above [37]. In order to find data-flow errors, the GTforDF (Graph Traversal for DataFlow) algorithm provide an approach that traverses the workflow to be analyzed in order to find all case instances. For every task of a case are defined the input and output sets.

Then we depicted the dataflow models (graphs which depict in a graphical way programs). These are closer to our first approach of data-flow model's extraction in terms of visualization (dataflow model uses circles for operands' representation instead for data elements). It uses some arithmetical operations (a program) as input in order to provide the visualization instead using an event log. But event logs do not refer only to arithmetical operations; they consist of activities (and data modifies at activity level). Any modification of data may be seen as an operation.

Conclusions

This chapter presents some data analysis methods and techniques or models focused on data-flow view of a process. We started with basic types of data modeling (e.g. ERD) because they underlie the design of relational databases. We also reminded some approaches that emphasis the control-flow perspective (e.g. UML Activity Diagram) in order to make the transition to approaches which combines data-flow perspective with control-flow perspective (e.g. Process Data Diagram).

Then we depicted the Product Data Model as it was defined in [33] and how it maps to our approach. An approach that uses PDMs is depicted in [1]: automatic data processing steps are grouped into activities. The XSD file to be imported in ProM consists of the set of activities and the set of data elements [33]. The drawback of this approach is that it considers the PDM XML file already created and based on it the aggregation is build. Our goal is to automatically create the PDM (first the XML file in XES format, then its visualization).

Dataflow model was defined in order to provide a graphical visualization of a program. The visualization is similar to PDMs' visualization (it uses circles for elements depicted in the program, e.g. for the operands). Our approach uses an event log as input. But event logs do not refer only to arithmetical operations; they consist of activities (and data modifies at activity level). Any modification of data may be seen as an operation.

Some data-flow anti-patterns are formalized in [32]. They depict errors in the flow of data and provide methods for their discovery (e.g. missing data pattern).

We reached the conclusion that none of the analyzed methods, techniques or models do not emphasize the data movement (or changes) through a process execution. Moreover, none of the approaches presented do not use as starting point an event log in order to analyze the data perspective.

3. Process mining

This chapter makes the introduction to the process mining domain (see Figure 5) by presenting the main notions of process mining. Process mining combines process modeling and analysis techniques with data mining and machine learning [4]. It consists of a set of techniques and methods which provide information about the analyzed process.

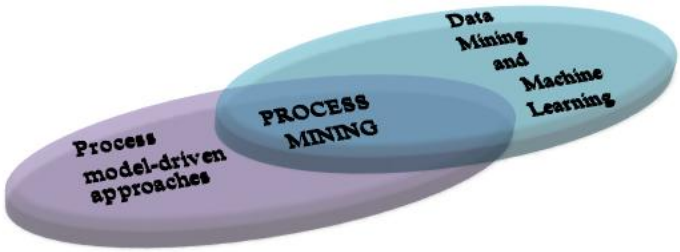

Figure 5 Process mining domain

Process mining techniques extract knowledge, models from event logs. Beside this, using process mining techniques, the business process can be monitored or improved. Each event refers to an activity of the process. It stores information about the name of the activity, the resource which completed the activity, the time when it took place or any other information related to the activity. The totality of events of a process execution forms a case (process instance). A case refers to a single execution of the process. More traces form the event log (see
Figure 6). The event log depicted below consists of a number of *t* traces. Each trace is composed of a certain number of events (it may differ from one trace to another; e.g. trace 1 has *m* events, while trace *t* consists of *r* events). The number of attributes of each event may differ from one event to another, e.g. event *r* from trace *t* has *s* attributes.

Process mining represents a large domain and it entails several approaches: process discovery, conformance checking and enhancement. Process discovery refers to discovery algorithms that extract knowledge through mined models. It involves the existence of an event log and based on the behavior observed in the log, a process model is built. Mined models may have different approaches:

a) *control-flow perspective*: focusing on the order of activities (e.g. Alpha miner[4], [8], [20], Fuzzy Miner [17], Heuristics Miner [38], Genetic Miner [21]);
b) *organizational perspective*: focusing on the relations between the resources (e.g. Social network miner [2]).

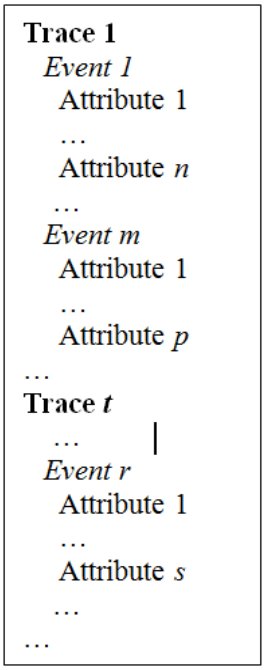

Figure 6 Event log components

Conformance checking approach [27] aims the discovery of deviations between the intended behavior of a process and the real behavior of a process. There are two ways in order to do that::

a) the original process model is compared to the mined model or

b) the analysis of the way how the initial model or the mined model are able to reproduce the input event log.

In order to apply process mining techniques to event logs some tools are needed. There are different tools for extracting knowledge from process execution logs. The issue is that they are using different input formats and the models resulted are represented in several ways. In order to solve this drawback, a research group from Eindhoven University of Technology developed a generic framework called ProM. ProM Framework supports several process mining techniques in the form of plug-ins. Moreover, ProM is able to import event logs compliant with the MXML [13], [14], [16] or XES [18] formats.

Each process-aware information system has its own data structure. For that, the event logs produces by different process-aware information systems have different formats. That is the reason why a standardized way to visualize these event logs is needed. The first standard format proposed was MXML (Mining Extensible Markup Language [13],[14],[16]). It represents a XML-based format for storing event logs and it has a standard notation for storing timestamps, resources and transaction types. It can add data elements to events and cases. The successor of MXML is the eXtensible Event Stream (XES). The aim of XES is to offer portability for general data mining, text mining, and statistical analysis [18]. The plug-ins developed in ProM 6 framework are grouped in packages and any package can be installed and updated independently of the framework.

Compared with MXML format, XES extension considers all attributes equals. Each attribute may have a different type (e.g.: string, date, integer, float or boolean). Paper [18] depicts each of these attribute types. In order to attach semantics to data, a series of

extensions have being defined. Every extension has a specific number of attributes. They may be defined at log, trace or event level. Moreover, an attribute may contain in its turn another attributes (meta-attributes). Here we speak about nested attributes. Each extension has three mandatory attributes: "*name*", "*prefix*" and "*uri*". The "*name*" attribute refer to the name of the extension. The "*prefix*" attribute makes the connection between the extensions declared at log level and the extension used at trace level, respectively at event level. The „*uri*" prefix holds the path to the definition of the extension. The current standard extensions [18], [35] are: the concept extension, the lifecycle extension, the organizational extension, the time extension and the semantic extension.

Conclusions

This chapter introduces the notion of process mining, an approach that combines elements from process modeling, data mining and machine learning. The starting point of process mining is the event log and the aim of process mining is to extract knowledge from event logs.

The event logs provided by PAISs or WfMSs use different input formats. That is the reason why ProM Framework was created: to provide a unified tool which includes a series of process mining algorithms. At the beginning it was developed only for the extraction of process models from event logs but latterly it offers process verification, conformance checking, social network analysis and much more.

Usually the plug-ins implemented in ProM Framework represents mining algorithms that analyze an event log (e.g. they offer a visualization of the process model depicted in the event log).

In order to provide a unified environment for process mining two process mining standards are defined: MXML and XES. We focused on XES standard and its extensions (e.g. concept extension) in order to provide the data-flow visualization. Thus, the event log used as starting point for the data-flow perspective use XES format.

We conclude this chapter by analyzing some plug-ins implemented in ProM Framework. None of them does not offer a visualization of data movement through a process, most of them focusing on control-flow perspective (e.g. Alpha Miner, Fuzzy Miner). Even if there are algorithms that take into account some data presented in the process, they do not offer a data-flow perspective of the process.

4. Mining Algorithms

This chapter presents the main contribution of the thesis by providing information about the implemented data-centric algorithms. Figure 7 emphasizes the steps followed in order to create the data-flow model. Because the event logs provided by information systems do not comply with the Data-Flow Event Log format we created two conversion tools: *Data-Flow Event Log format Convertor* and *Convert to I/O log*. First we defined the XML format for the Data-Flow event logs (XES format). Therefore, we defined an extension which characterizes Data-Flow event logs (*InputOutput* extension). Having the desired XES format we analyze

the event logs generated by decision-aware information systems and YAWL system by applying the implemented mining algorithms. The event logs generated by the decision-aware information system do not respect the XES format, but they contain the needed data elements. They are conformant to the old format proposed for event logs, namely the MXML format.

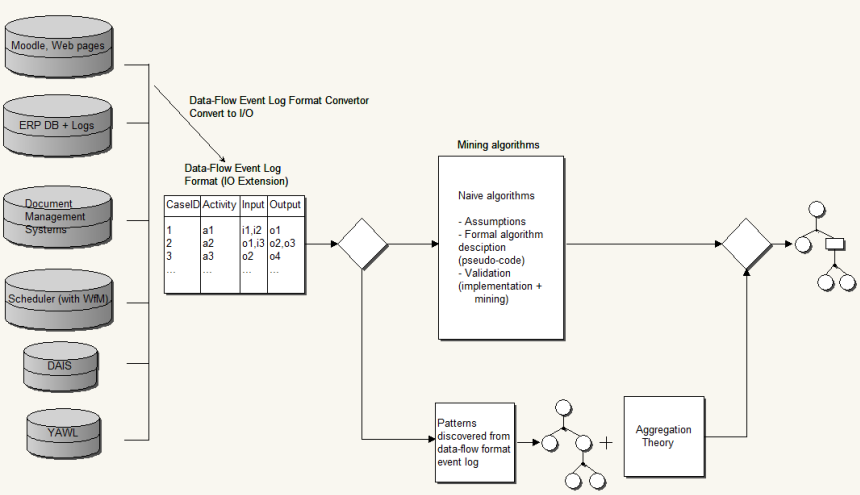

Figure 7 General approach

We propose two approaches in order to get the data-flow perspective of a process:

a) the first approach provides a PDM for each trace from the event log and

b) the second approach provides a PDM for the entire event log.

In the first case an aggregation of all (or as much traces the user wants) traces is possible. More details can be found in [24]. First we begin with the development of an application which offers the data-flow perspective of a process having as input a single trace from an event log. Then we will depict the naive algorithms integrated in ProM Framework.

First approach uses event logs generated by a decision-aware information system. For each execution of the process depicted by the DAISs a PDM is provided. This approach takes as input a trace from a XML file that contains all the recorded operations made by the user in a web-based software. The software represents an application for a credit loan. It may be used by the manager of a company in order to make a decision concerning a credit loan. The software offers information about the activity of the company in a whole year. The decision maker can make a complete funding decision regarding the loan to be contracted or a decision related to the loan: the period of credit loan, the type of credit loan and so on.

The event logs generated by the user interaction with the software are stored in a database consisting of five tables and they comply with the MXML format depicted in the previous chapter. Appling the mining algorithm through the resulted event logs, a PDM is automatically generated.

This chapter offers insights about the pseudo-code of the PDM-XML file generation. This file must be imported in ProM Framework (5.2 or newer) in order to get the graphical visualization. As we argued earlier, for each case id a PDM is built.

In order to validate our first approach several running examples and four use cases are used. The PDM-XML file structure is created based on the following elements: BD (basic

data), ID (introduced data), DD (derived data), and O (output). Next step is to define the operations based on BD, ID and DD. Once these elements are extracted, the PDM-XML file may be generated and imported into ProM Framework in order to get the data-centric visualization.

Second part of the fourth chapter of the thesis offers insights about three naive algorithms implemented in order to provide the data-flow perspective: Naive Algorithm A, Naive Algorithm B and Naive Algorithm C. Each algorithm produces a different graphical visualization. Naive algorithm A focuses on the set of existing activities as they appear in the event log. The second algorithm compares the current activity in the log with the previous activities taking into consideration the input data elements, respectively the output data elements. Finally, Naive algorithm C includes the probability that an operation may generate multiple output data elements.

As we argued earlier, the mining data-centric algorithms need event logs respecting the Data-Flow Event Log format as input. Therefore, we defined an extension which characterizes Data-Flow event logs (InputOutput extension). This extension defines nested elements in order to associate input, respectively output data elements for each event. Moreover, this definition associates two data elements for each event (*input* and *output*). For every *input* element two data elements are defined: name and value. In the same way, a particular *output* element contains two data elements. As the names are suggesting the *name* element refers to the name of input/output data element(s), while the *value* element provides the value of the input/output data element(s). Considering the fact that an event may contain more than one input or output data element we choose to delimitate the data elements by a hash tag („#"). This extension only defines the data elements needed for each activity, while the name of the operation may be found in concept extension.

CaseID	Activity	Input Data Elements	Output Data Elements
1	registration	fName, lName, title, street, City, pCode, country, phone, eMail	newClient
1	roomInformation	newClient, roomNo, rateType, noNights, noAdults, noChildren and notes	roomInfo
2	registration	fName, lName, title, street, City, pCode, country, phone, eMail	newClient
2	roomInformation	newClient, roomNo, rateType, noNights, noAdults, noChildren	roomInfo

Table 1 Cases for Naive Algorithm A and B

Next we will present the data-centric mining algorithms. We developed three data-centric mining algorithms. For each algorithm we provide a formal definition, a pseudo-code and a series of particular assumptions (e.g. multiple input data elements, single output data element, multiple output data elements, frequency etc.).

Naive algorithm A identifies the operations uniquely based on their names. If there exists an activity which have the same name, but different input data elements or/and different output data elements, the operation which is represented in the model is the first one found in the event log.

For Naive Algorithm B, an operation is defined on the strength of the component elements: input data elements set, respectively output data elements set. First we are looking for input

and output data elements in the event log and then we assign the set of input/output data elements to operations. Therefore, the operations having different input (output) data elements will be shown in the PDM visualization. The last algorithm (Naive algorithm C) focuses on the multiple output data elements (its functionalities are the same with the functionalities of Naive Algorithm B, the difference being that the first reminded algorithm accepts multiple outputs).

Next we depict the differences between first two algorithms. The input log is presented in Table 1 while the data visualization in Figure 8.

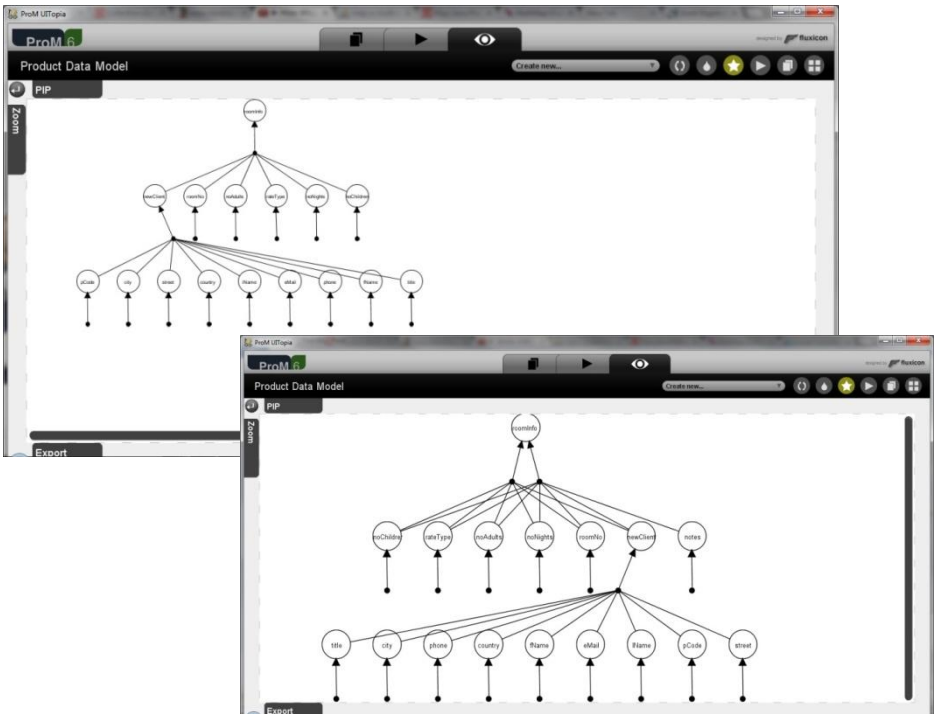

Figure 8 ProM Framework – Naive Algorithm A applied to hotel registration process logs from Table 1

Next we will focus on the convertors developed: Data-flow Event Log Format Convertor – DFC and Convert to I/O. First applies to event logs produced by ERP systems and it is not fully automated while the second one refers to event logs with start and complete events (e.g. event logs produced by YAWL system).

First step of DFC is to determine the process to be analyzed. In this phase, business specialists (e.g. business analysts, managers, head of departments, etc) derive process steps based on internal procedures and based on their experience. Then we set trackers on the tables which are providing the necessary data in order to execute the process. Once we have the event log generated by the system we can apply the Data-flow Event Log Format algorithm. First we have to define the clusters for our process. These are extracted from the process activities (e.g. for Order to Cash process we can have the following clusters: create order, generate invoice and cash invoice). Basically each activity represents a cluster. Once we have the clusters created, we analyze each of them. For each cluster all data is collected (including their names and values). Hence, these are the input data elements according to

InputOutput extension. As for the output data element we create an artificial data element which takes as name the initials of the belonging cluster. It also will keep the related value.

The second convertor (Convert to I/O) transforms start and complete event logs into event logs complying with Data-Flow Event Log Format. The conversion plug-in is integrated in the ProM Framework („Convert to I/O log" plug-in). It takes an event log having start and complete events (in XES format) as input and it returns the corresponding event log in data-flow event log format (with input and output data elements for each event). Event logs with *start* and *complete* events make the separation of data elements easier. Furthermore artificial data elements are no more needed (like in the previous conversion approach - DFC).

Conclusions

This chapter introduces the main contribution of the thesis. In this line two approaches are defined: a) first approach provides a PDM for each trace from the event log and b) second approach provides a PDM for the entire event log. The second approach takes as input an event log using the input/output format. On this direction, the input/output format was defined using XES standard. This is the format standard allowed by ProM Framework (beside MXML format). The mining framework lacked plug-ins that provides a data-flow visualization, the most of implemented plug-ins focusing on control-flow perspective (e.g. α-algorithm, Fuzzy Miner, Heuristics miner, etc.). For each algorithm which provides the data-flow perspective a plug-in was implemented in ProM Framework and each plug-in offers a different visualization of the process.

In order to comply with Data-Flow Event Log format (and *IO* extension), two convertors are developed: Data-flow Event Log Format Convertor – DFC and Convert to I/O. First applies to event logs produced by ERP systems and it is not fully automated while the second one refers to event logs with start and complete events (e.g. event logs produced by YAWL system).

5. Data patterns

In order to validate the data patterns from Product Data Models we used the event logs from [4]. The logs were created in order to focus on the control-flow perspective. Therefore, only data regarding the name of the activity, the resource executing the activity, the timestamp when activity was performed were depicted in the event log. For routing conditions there are also used data. We improved these event logs by filling additional data for each event in order to highlight the data perspective. Therefore, we used the *InputOutput* extension defined. Each event is characterized by input and output data elements. Even if we introduced new data elements for each event from the log the control-flow remains the same.

We introduced five data-patterns identified at event level (e.g. Data Join Attributes – DJAP, Same output(s), different input(s) – SODI, Data Updates - DU), but these data-patterns may also be found at trace level. SODI and SIDO patterns provide the alternative for Exclusive Choice (XOR-Split) pattern from control-flow.

The figure below shows the data-patterns that can be identified from a data-centric visualization. It is obviously that data-patterns cannot be identified in the control-flow perspective. In this case XOR-Split pattern from control-flow perspective is represented by SIDO pattern from data-flow perspective.

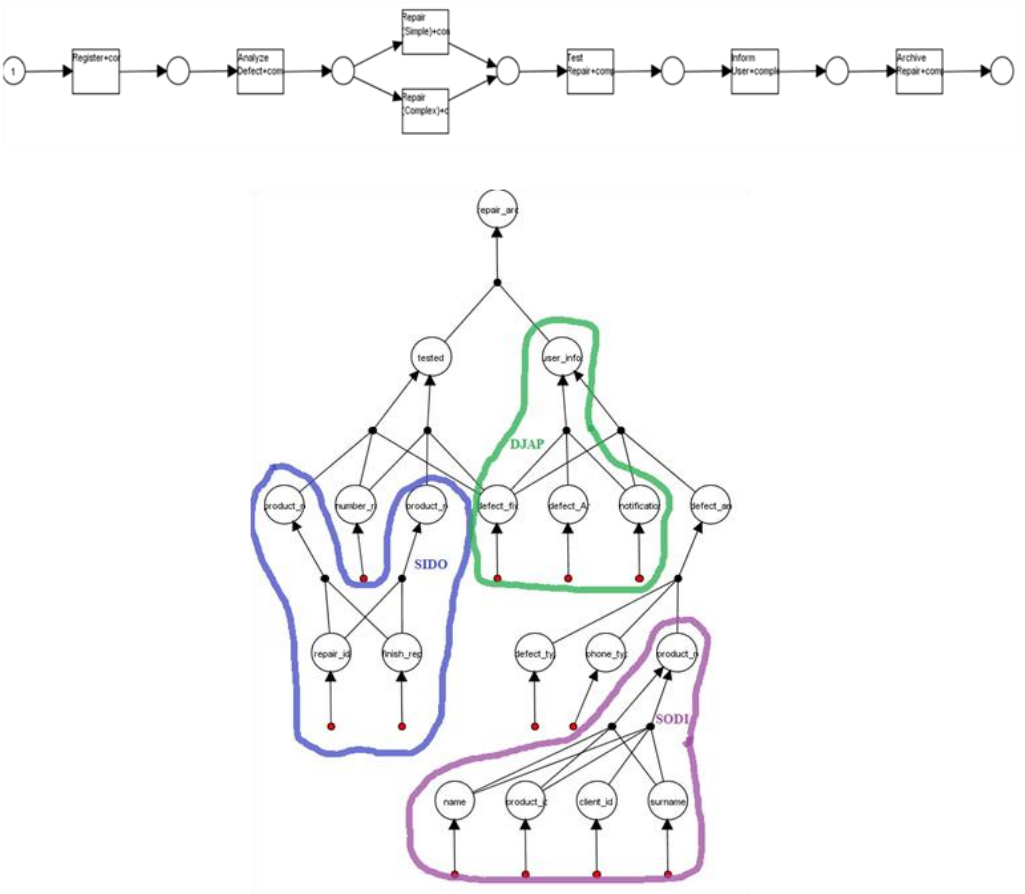

Figure 9 Control-flow vs. Data flow (Data patterns)

Conclusions

In this chapter a series of data-patterns is depicted (e.g. a) basic data patterns (BP): data join attribute pattern (DJAP), b) complex data patterns: same input(s), different output(s) pattern (SIDO), Same output(s), different input(s) (SODI), and c) data values data patterns: data updates (DU), conditional data value (CDV)). Moreover, a formalization of these data patterns is provided.

Data patterns also offer data-based representation for control-flow patterns like Exclusive Choice (XOR-Split) pattern. We propose two data-based representations for Exclusive Choice through *Same output(s), different input(s)* (SODI) and *Same input(s), different output(s)* (SIDO) patterns. The most suitable to XOR-Split pattern is the first one (SODI) because when we have two activities whose produced data element are the same, also the operations (activities) are the same; while if the output data element are different, also the activities are not the same.

6. Case studies

The sixth chapter validates the implemented algorithms using two case studies (see Figure 10). For each case study different conversion tools were used in order to prove the generality of the conversion tools. The first case study uses XESame and DFC converter in order to provide the event logs because the data source is given by Excel (.xlsx) files; while for the second case-study we implemented a new conversion tool for event logs using *start* and *complete* events.

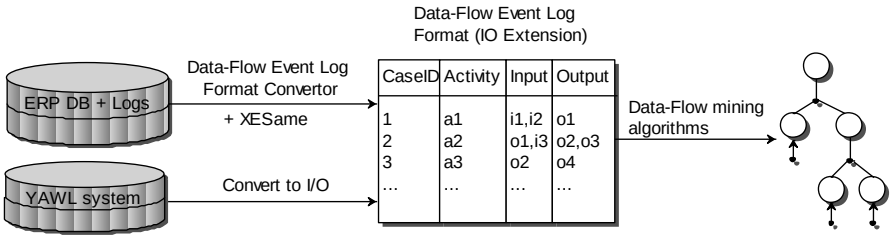

Figure 10 Case studies approaches

These conversion tools transform the event logs in the desired format (Data-Flow Event Log format). Therefore we can apply the naive (data-centric) algorithms. For each case study we presented the visualization provided by a plug-in which emphasizes the control-flow of a process (e.g. Alpha Miner). Then we applied the naive algorithms and we depicted the advantages provided by the data-flow perspective.

The event logs used for the first case study are extracted from a data export from Navision, an ERP system used by several companies from Romania. The data for this case study is provided by Farmec[1].

This chapter provides a section that presents in a detailed manner the way how Navision event logs are converted to -Flow Event Log format. The resulted event log may be imported into ProM Framework and a series of analysis may be realized. After we imported the resulted event log into ProM we observed that the event log has 251 cases and 523 events. First we applied some algorithms that focus on the control-flow perspective. These models do not show which data is consumed or produced by a particular activity even if this data is stored at activity level.

Naive algorithm A focuses on the operations (activities), while Naive Algorithm B (see **Figure** 11) concentrates on the input and output data elements of operations. Even if in the event log there are orders which are not specifying the client whom an order belongs this operation does not appear in the model. This shortcoming is solved by Naive Algorithm B since it focuses on the data elements consumed or produced by an activity. Second algorithm shows that there are two orders for which the buyer is not known. Therefore 251 orders have all the information available. Moreover the model shows that for each order an invoice was generated, but only 21 invoices were collected.

[1] www.farmec.ro

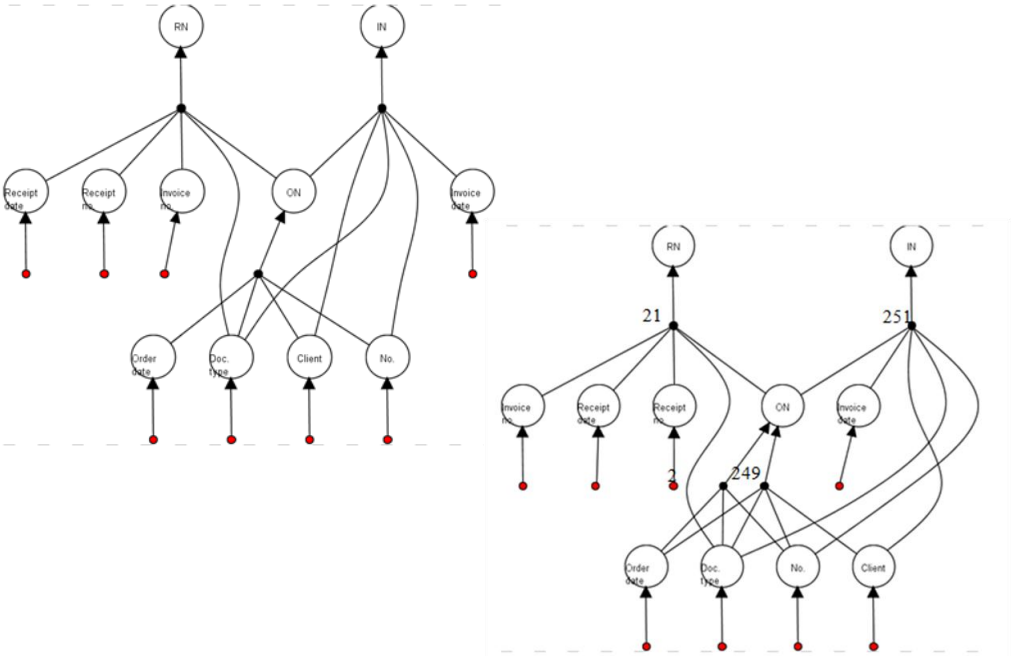

Figure 11 Naive Algorithm A vs. Naive Algorithm B applied on Navision event log

The second case study uses YAWL event logs. First, we created a YAWL specification depicting the approval process for going in an international or national mobility (see **Figure** 11). Then we generated event logs through process' simulation. For a better understanding we used two traces generated after the workflow execution. On the first execution of the workflow specification all the conditions are accomplished by the users (see the red tasks from Figure 12). On the other hand, on the second execution (see the green tasks from Figure 12), the mobility is not preview in the Plan. Therefore, a modification of the Plan is necessary and then it must be approved. The rest of the workflow has the same execution like the first one.

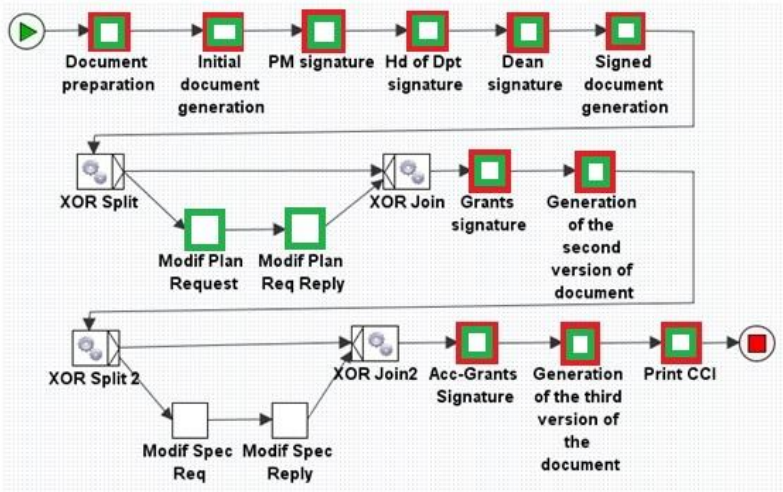

Figure 12 Control-flow for getting the approval to go in an international or national mobility process

We used "Convert to I/O" plug-in on the event logs resulted after the execution of YAWL specification. We mapped each event from the event log to an operation from the

PDM. We set the attributes from start event to the set of input data elements, while the attributes from complete events to the set of output data elements.

There are some exceptions when we refer to the first task of the workflow. The first task of the workflow refers to the preparation of the mobility document. Being the first task, this is a particular case because it only involves input data elements in the PDM visualization (*departure_time, arrival_time, city, holder* and *amount*). In order to make the PDM executable, at implementation level an exception is thrown. If the current data element is certified as input data element, before it is added to the PDM model verification is performed: namely if the data element is produced (it is output data element in a previous operation) until the current state.

The input/output converter does not take into consideration the XOR (OR) pattern from control-flow perspective. Yet the XOR pattern may be easily identified in the PDM visualization (see Figure 13).

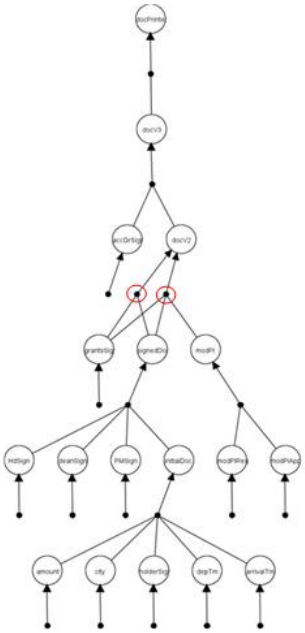

Figure 13 XOR pattern in PDM

Conclusions

In order to validate the proposed algorithm two case studies are introduced. For the first case study we used the event logs produced by an ERP (Navision), while for the second one we used the event logs generated by YAWL system. In order to change the event logs from Navision in the desired format we used XESame 1.3. and DFC, while for the event logs generated by YAWL system, the development of a converter was necessary. The data-flow visualization of a process is possible if information about each operation presented in the event log can be extracted (name of operation, input data element(s) and output data element(s)). Considering the event logs generated by YAWL the extraction of PDM elements was possible by analyzing the start and complete events.

Both case studies show the additional information brought by the data-flow perspective because it provides more information about the process than control-flow

perspective does. Moreover PDMs are able to represent XOR and AND patterns specific to control-flow perspective.

Conclusions and future work

Last chapter concludes the thesis and presents future work. Generally, process mining methods and techniques provide a control-flow perspective of the process, ignoring the data-flow perspective. We became aware that recently there is an increased interest shown for the data perspective (e.g. many employees from real companies are asking questions like "if I am in a certain state of a process, what data I know so far and what data will I need to find out in order to complete the process?"). This motivated us to pursue a mining approach on event logs aimed at extracting a data-flow model. We introduced the problem context in the first chapters of the thesis, then we presented our approach and, finally, we validated our claims using various event logs.

The main contributions of this thesis are:

- Defining a standard format for the event logs with data (IO extension using XES standard);
- The development and implementation of several mining algorithms which mine a model showing the data-flow perspective;
- The implementation of two conversion (DFC and Convert to I/O)
- Using different sources of data in order to validate our approach
- Providing a XOR-Split pattern alternative in the context of Data-flow perspective

The data-centric algorithms focus on the names of data elements from a process and each data element has the same value during a process execution. But the value of each data element is also important. Thus, we want to include in our research the analysis of the values of data elements. So far, for a process execution (trace) a data element has the same value, but in reality its value may be changed during a process execution or it can be deleted (in this case the data element will not arise in the resulted model). A value change during a process execution may cause cycles, which are not allowed according to the standard PDM definition. As future work we intend to include data values in order to provide a more accurate data-centric representation.

We conclude that we reached the proposed objectives by providing automated and semi-automated methods to get the data-flow perspective.

Selected References

[1] van der Aa H., Reijers H.A., and Vanderfeesten I., Composing Workflow Activities on the Basis of Data-flow Structures. In: Proceedings of the 11th International Conference on Business Process Management (BPM 2013), Lecture Notes in Computer Science, vol. 8094, pp. 275-282, Springer Verlag, Berlin, 2013

[2] van der Aalst W.M.P., "On the Automatic Generation of Workflow Processes based on Product Structures", Computers in Industry, 39:97-111, 1999

[3] van der Aalst W.M.P., "Woflan: a Petrinet- based workflow analyzer", Syst. Anal. Model. Simul., 35, 3, 345-357, 1999

[4] van der Aalst W.M. P., "Workflow verification: Finding control-flow errors using petri-net based techniques", In W.M. P. van der Aalst, J. Desel, and A. Oberweis, editors, Business Process Management: Models, Techniques, and Empirical Studies, pages 161-183. Springer-Verlag, Berlin, 2000

[5] van der Aalst W.M.P. and van Hee K.M., "Workfow Management: Models, Methods", and Systems. MIT press, Cambridge, MA, 2004

[6] van der Aalst W.M.P., Weijters A.J.M.M., and Maruster L., "Workflow Mining: Discovering Process Models from Event Logs", IEEE Transactions on Knowledge and Data Engineering, 16(9):1128-1142, 2004

[7] van der Aalst W.M. P., Reijers H. A. , and Song M., Discovering social networks from event logs. Computer Supported CooperativeWork, 14(6):549-593, 2005

[8] van der Aalst W.M.P., Process Mining: Discovery, Conformance and Enhancement of Business Processes, Springer Verlag, 2011

[9] Basu A., Blanning R.W., "Metagraphs and Their Applications", Integrated Series in Information Systems, Senes, Springer, 2007

[10] Basu A., Blanning R.W., "Metagraphs in workflow support systems", Decision Support Systems 25, pages 199- 208, 1999

[11] BPM and Workflow Handbook, Future Strategies Inc., Layna Fischer (ed.), 2007,

[12] Chen P.P.-S. , "The entity-relationship model toward a unified view of data", ACM Transaction Database System 1, 1 pp. 9-36, 1976

[13] van Dongen B. F. and van der Aalst W. M. P., "A meta model for process mining data", In Proceedings of the CAiSE, 2005, vol. 5, pages 309-320

[14] van Dongen B., Alves de Medeiros A.K., Verbeek H.M.W., Weijters A.J.M.M., and van der Aalst W.M.P., "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

[15] Du N., Liang Y., Zhao L. , "Data-flow skeleton filled with activities driven workflow design", in Won Kim & Hyung-Jin Choi, ed., 'ICUIMC' , ACM, pages 570-574, 2008

[16] Gunther C. and van der Aalst W.M.P., "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

[17] Gunther C.W. and van der Aalst W.M.P., "Fuzzy Mining: Adaptive Process Simplification Based on Multi-perspective Metrics" , In G. Alonso, P. Dadam, and M. Rosemann, editors, International Conference on Business Process Management (BPM 2007), volume 4714 of Lecture Notes in Computer Science, pages 328-343, Springer-Verlag, Berlin, 2007

[18] Gunther C.W., "XES Standard Definition", Fluxicon Process Laboratories, November 2009

[19] Hammer M., Champy J., "Reengineering the Corporation: A manifesto for Business Revolutuon", New York, 1993
[20] de Medeiros A.K.A., van Dongen B.F., van der Aalst W.M.P., Weijters A.J.M.M., "Process Mining: Extending the ?-algorithm to Mine Short Loops", BETA Working Paper Series, WP 113, Eindhoven University of Technology, Eindhoven, 2004
[21] de Medeiros, A. K. A., Genetic Process Mining. PhD thesis, Technische Universiteit Eindhoven, Eindhoven, The Netherlands, 2006
[22] Orlicky J.A., "Structuring the bill of materials for mrp", Production and Inventory Management, pages 19-42, 1972
[23] Petri C.A. , "Kommunikation mit Automaten", PhD thesis, Institut fur instrumentelle Mathematik, 1962
[24] Petruşel R., Aggregating individual models of decision-making processes. InProceedings of the 24th international conference on Advanced Information Systems Engineering (CAiSE'12), Jolita Ralyté, Xavier Franch, Sjaak Brinkkemper, and Stanislaw Wrycza (Eds.). Springer-Verlag, Berlin, Heidelberg, 47-63, 2012
[25] Reisig W., "Petri Nets: An Introduction", volume 4 of Monographs in Theoretical Computer Science: An EATCS Series. Springer-Verlag, Berlin, 1985
[26] Riehle D. and Züllighoven H., "Understanding and Using Patterns in Software Development", Theory and Practice of Object Systems, 2(1):3-13, 1996.
[27] Rozinat A., and van der AalstW. M. P., Conformance checking of processes based on monitoring real behavior. Information Systems, 33(1):64-95, 2008
[28] Sadiq S.W., Orlowska M.E., Sadiq W., Foulger C., "Data-flow and Validation in Workflow Modelling", In Fifteenth Australasian Database Conference (ADC), Dunedin, New Zealand, volume 27 of CRPIT, pages 207-214, Australian Computer Society, 2004
[29] Sadiq W., Orlowska M.E., "Analyzing Process Models using Graph Reduction Techniques", Information Systems, 25(2):117-134, 2000
[30] Sun S. X., Zhao J.L., "Activity Relations: A Dataflow Approach to Workflow Design", proceedings of International Conference on Information Systems 2006, Milwaukee, Wisconsin, USA, Paper 44, 2006
[31] Sun S.X., Zhao J.L., Nunamaker J.F., Liu Sheng O.R., "Formulating the Data-flow Perspective for Business Process Management", Information Systems Research, 17(4), pages 374-391, 2006
[32] Trcka N., van der Aalst W.M.P. , and Sidorova N., "Data-Flow Anti-Patterns: Discovering Data-Flow Errors in Workflows", In P. van Eck, J. Gordijn, , and R. Wieringa, editors, Advanced Information Systems Engineering, Proceedings of the 21st International Conference on Advanced Information Systems Engineering (CAiSE'09), volume 5565 of Lecture Notes in Computer Science, pages 425-439. Springer-Verlag, Berlin, 2009
[33] Vanderfeesten I., "Product-Based Design and Support of Workflow Processes", Eindhoven University of Technology, Eindhoven, 2009
[34] Verbeek E., van der Aalst W. M. P., "Woflan 2.0: a Petri-net-based workflow diagnosis tool", In Proceedings of the 21st international conference on Application and theory of petri nets (ICATPN'00), Mogens Nielsen and Dan Simpson (Eds.), Springer-Verlag, Berlin, Heidelberg, pages 475-484, 2000
[35] Verbeek H., Buijs J. C. A. M., Dongen B. F., and van der Aalst W. M. P., "XES, XESame, and ProM 6", Information Systems Evolution, pages. 60-75, 2011
[36] Verbeek H.M.W., Basten T., van der Aalst W.M.P., "Diagnosing Workflow Processes using Woflan", Computing Science Report 99/02, Eindhoven University of Technology, Eindhoven, The Netherlands, 1999

[37] Wang J., Kumar A., "A framework for document-driven workflow systems", In Proceedings of the 3rd International Conference on Business Process Management. Lecture Notes in Computer Science, vol. 3649, Springer Verlag, pages 285-301, 2005
[38] Weijters A.J.M.M. and van der Aalst W.M.P., "Rediscovering Workflow Models from Event-Based Data using Little Thumb", Integrated Computer-Aided Engineering, 10(2):151-162, 2003