Process Mining in Case Handling Systems

Christian W. Günther and Wil M.P. van der Aalst

Department of Technology Management, Eindhoven University of Technology
P.O. Box 513, NL-5600 MB, Eindhoven, The Netherlands
{c.w.gunther, w.m.p.v.d.aalst}@tm.tue.nl

Abstract. Process mining has proven to be a valuable tool for tracking down problems or inefficiencies within a variety of business processes, using information from event logs. Compared to traditional process-aware information systems (e.g. workflow management systems), case handling systems allow for much more flexibility by adding all kinds of implicit controls. In this paper we reason about what information can be extracted from a case handling system and how this is accomplished. Further, we investigate how this data can be exploited, in order to achieve greater insight into how business processes are handled within such a flexible environment.

1 Introduction

While checking, monitoring and supervising industrial production processes is quite straightforward, i.e. one only has to physically follow the product through its production stages, this does not hold for informational processes in the same way. Of course the basic procedure is being prescribed by the designed process model, but in case of problems emerging during enactment workers tend to bypass the system and work "behind its back". On the one hand, this makes it hard to spot problems located in the process model itself, as they are effectively evaded by experienced workforce, while on the other hand other problems arising from these very actions can no longer be traced back to the process model and lead to mysterious and unforeseen behavior. These are situations where *process mining* proves to be a valuable tool in unveiling properties of process enactment that would otherwise remain hidden within the system. The general idea of process mining is to examine event logs (i.e. process execution logs also known as transaction logs or audit trails) created by some process-aware information system and analyze them in various aspects. The goal of this analysis is to aid in discovering problems or inefficient behavior and in subsequently tracing these back to deficiencies located in the process model or the organization itself.

Workflow Management (WFM) systems are a nice example of a technology used to construct process-aware information systems. While production workflow technology is becoming increasingly widespread, both concerning "stand-alone" installations of full-fledged WFM systems like Staffware, IBM MQSeries Workflow or COSA and the inclusion of WFM components in e.g. ERP (Enterprise Resource Planning) systems like SAP R/3, PeopleSoft and Oracle, it is

burdened with a set of fundamental problems that cause a fair deal of inefficiencies in practical use and hinder further proliferation of this paradigm. The tightly prescribed process model leads to an inherent lack of flexibility [3, 5, 11, 16, 20, 21, 24, 27, 29, 32] which makes WFM systems a serious roadblock when it comes to changing business processes. As has been argued in [10] the process oriented legacy of workflow management systems, focusing on causal relationships between tasks as its primary driver, leads to four crucial problems. One that is deeply rooted in the WFM paradigm is the partitioning of work in order to make it distributable, while this work is usually performed at a far more fine-grained level by the employees involved. Further, the way WFM systems distribute work is closely intertwined with authorization, i.e. workers are always presented with all activities they are allowed to perform. This can be a problem especially for high-ranking employees whose role implies a large set of sub-roles, such that the really important tasks that only they can perform drown within a mass of other, less demanding tasks. Another downside is the phenomenon of *context tunneling*, i.e. workers are more or less isolated within their currently handled activity and have no overview about the case as a whole. This leads to errors and inefficiencies that could otherwise be evaded easily. However, the problem that is most responsible for the inflexibility of traditional workflow management is its strongly push-oriented nature of routing, prescribing what to do instead of leaving the choice to the users.

To address these problems the *case handling paradigm*, introduced in [2, 10] and implemented in successful tools such as FLOWer (Pallas Athena, [14, 13]), promises remedy. The basic idea is to base progress within the process model on the availability of defined data elements. Further novelties enable users to substantially deviate from the predefined process model, as these no longer describe what is allowed but rather what is *not*, granting a maximal degree of freedom and nearly unlimited navigation within the process. In such an environment where one execution of a process model may look completely different to another one, depending on the case itself and who is working on it, process mining can be expected to yield even more interesting insights into an organization and its processes. Concerning the process perspective it is no longer bound to rediscovering the original process model but can rather be used to see how process-aware workers organize their tasks, discover their otherwise tacit best practices, and thus make them persistent. One further opportunity is to make use of the more data-centered approach of case handling systems by taking into account information associated with data elements in the mining process. In this way a far more detailed understanding can be gained concerning how data is created and transformed during a specific business process.

This paper investigates opportunities, challenges and new methods that may arise from applying process mining techniques to case handling systems, or their enactment logs. The following sections give a short introduction into the case handling paradigm and the technique of process mining. Section 4 introduces different types of logs that can be extracted from a case handling system, after which concrete tools are introduced in section 5. In the subsequent section the

design of a generic converter and envisioned process mining techniques for case handling systems are discussed.

2 Case Handling

While production workflow, emphasizing the routing between atomic activities, is strongly process-oriented, the case handling paradigm focuses mainly on the *case* itself. The case is the primary object to be manufactured, e.g. the outcome of a lawsuit or the response to a customer request. In this case, single activities diminish in importance in favor of the larger context. They are no longer considered atomic steps that have to be performed in an "all or nothing" manner, but rather make up the logical partitions of work between which a transition from one worker to another is possible.

Like in traditional WFM there exists a set of precedence relations between single activities making up a process, using well-known patterns [4] for that. However the primary driver for progress is no longer the event of explicitly finishing activities but the availability of values for data objects. While production workflow clearly separates the process from associated data, case handling integrates both far more closely, using produced data not only for routing decisions but also for determining which parts of the process have already been accomplished. With case handling, each task has associated to it data objects for three distinct purposes, while the first association is between a task and all data objects that are accessible while performing it. Furthermore, all data objects that are *mandatory* for a task have to be set (i.e. bound to a value) before the task itself is considered to be accomplished by the system. Finally, every data object can have a random number of tasks to which it is *restricted*, meaning that it can only be altered while performing one of these tasks. The *mandatory* and *restricted* properties are independent from each other, however reason dictates to have the last (in the sense of a causal chain) task featuring a data object as *restricted* also declared as *mandatory* (to make sure it will be provided). User-interactive tasks are connected to a *form*, each providing access to a selection of data objects. Note that one form can be associated to multiple tasks; on the other hand it is also possible to associate a form to the case itself, so that it can be accessed at any point in time.

In order to introduce the case handling principle, we use a small abstract example. Figure 1 shows a very simple *case type*. The concept of a case type is the case handling analogy to a workflow process definition: three tasks A, B and C are making up the process, sequentially chained by causal relationships denoted by connecting arrows. Their *mandatory* relationships to the three data objects x, y and z below are denoted by (curved) lines connecting activities (middle) to data elements (bottom). Similarly, activities are associated with the forms M and N (top). As it can be seen in the illustration, tasks A and B share the same form M, which provides access to data objects x and y. If a properly authorized worker now starts handling task A, the associated form M will open and he will start providing values for the presented data objects. In a traditional WFM

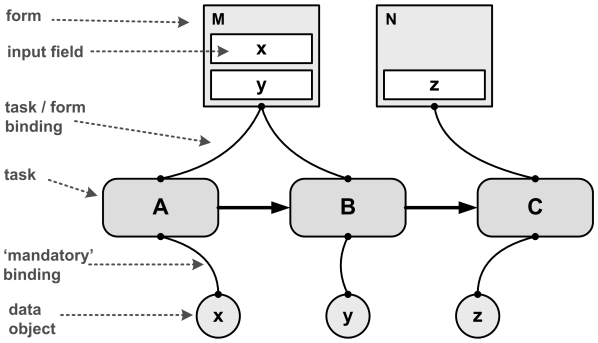

Fig. 1. Simplified example case type

system activity A would not be finished before form M is closed. However, a case handling system regards A as finished as soon as a value for x has been provided (and confirmed appropriately), automatically enabling task B in the background. If the worker would now close form M, another employee could pick up the case where he left it, starting task B, which would provide the same form M with x having a value filled in (that could now be changed again). Another possibility is, however, that the first worker keeps on handling the form, providing also a value for y. This would correspondingly trigger the *auto-completion* of task B (as all associated mandatory data elements, in this case only y, have been provided) and activate task C. Note that if a worker closes a form after filling out only parts of the mandatory data fields of a task, despite the task not being considered finished data already entered is not lost but will be presented to the person continuing work on that task.

Such a closely intertwined relationship between data and process obviously abandons their, often unnatural, separation so rigidly pursued in traditional workflow management. With the status of case data objects being the primary determinant of case status, this concept overcomes a great deal of the problems described in the introduction:

- Work can now be organized by those performing it with a far higher degree of freedom. Activities can either be performed only partly, without losing intermediary results, or multiple related activities can be handled in one go, surpassing the considerably weakened border between single tasks.
- The phenomenon of *context tunneling* can be remedied e.g. by providing overview forms directly associated to the case. Every authorized worker can consult such form at any point in time, ensuring he is aware of the context where necessary.
- Routing is no longer solely determined by the process model. Case types can be designed in such a way that multiple activities become enabled concurrently, providing different ways of achieving one goal. It is up to the user to decide which way to go, with the system "cleaning up behind", i.e. disabling or auto-completing tasks that have not been chosen.

In addition to the *execute* role, specifying the subset of resources allowed to handle a specific task, the case handling principle introduces two further roles crucial for operation. The *skip* role allows workers to bypass a selected task, which could be interpreted as an *exception*. When one thinks of real business processes an exception, like skipping an activity that deals with thoroughly checking the history of a client before granting a mortgage for well-known and trusted clients, is likely to occur quite frequently. The ability to grant the *skip* role to a senior worker renders the necessity for implementing such bypass obsolete, thus greatly simplifying the whole case type. It has to be noted that in order to skip a task all preceding tasks that have not been completed yet have to be skipped (or completed) beforehand. Traditional workflow definitions use loops for repeating parts of the process, e.g. because they have not yielded an expected result. In a case handling system, such construct has been made obsolete as well by the introduction of a *redo* role, enabling its bearer to deliberately roll the case's state back and make a task undone. In doing so, the values provided for data objects during this task are not discarded but merely marked as *unconfirmed*, so that they serve as kind of template when re-executing the affected task. Similar to skipping, before a task can be *redone* all subsequent tasks that have already been completed need to be rolled back as well. Roles in a case handling system are *case specific*, i.e. having assigned the role "manager" for a case type A does not imply that one can play the same role for another case type B. They can be specified in form of a *role graph*, where single role nodes are connected to each other by arcs, symbolizing *is-a* relationships; i.e. being authorized to play a role also implies the authorization to play all roles connected as child nodes.

Intertwining authorization with distribution of activities has been one of the major flaws in contemporary WFM systems, because "What one can do" (authorization) does not need to coincide with "What one should do" (work distribution). In a case handling system, the classical *in-tray*, i.e. a list of all activities the user is authorized to perform and that he can choose from, has been replaced by a sophisticated *query mechanism*. This tool can be used to look for a specific case, based on certain features (e.g. case data, or enactment meta-data like the start date of a case instance). Moreover it can be used to create predefined queries tailored to each worker (or, group of workers). A manager is no longer constantly flooded with all possible activities that he can perform, but only those which require a certain role, or e.g. case instances of an order where the combined value exceeds $1000. Obviously the query mechanism can also be used to perfectly imitate a classic in-tray, be it required.

As a reference implementation of the case handling concept, we will use FLOWer [14]. FLOWer of Pallas Athena is one of the most successful software tools to create process-aware information systems on the Dutch market. There it is competing with classical workflow products such as Staffware, FileNet, and COSA. Besides FLOWer there are few other case handling tools. Related products are E.C.H.O. (Electronic Case-Handling for Offices), a predecessor of FLOWer, the Staffware Case Handler [31] and the COSA Activity Manager [30],

both based on the generic solution of BPi [15], Vectus [25, 26], and the open-source system con:cern (http://con-cern.org/).

3 Process Mining

The goal of *process mining* is to extract information about processes from transaction logs [7]. One reason for this is to discover patterns in the way people work and processes unfold, or one may want to check whether things are really executed the way it was prescribed. The results of process mining may be used to align the information system, organization and processes in a better way. Also, a careful analysis of the way things really are, may assist in improving the efficiency, effectivity and quality of business processes. Any redesign effort based on incorrect assumptions of the current process is like an accident waiting to happen. This section introduces the concept of process mining with classical process-aware information systems in mind. In Section 4 we shift to case handling systems.

Traditional process mining techniques assume that it is possible to record events such that (i) each event refers to an *activity* (i.e. a well-defined step in the process), (ii) each event refers to a *case* (i.e. a process instance), (iii) each event can have a *performer* also referred to as *originator* (the person executing or initiating the activity), and (iv) events have a *timestamp* and are totally ordered. Table 1 shows an example of a log involving 19 events, 5 activities, and 6 originators.[1] In addition to the information shown in this table, some event logs contain more information on the case itself, i.e. data elements referring to properties of the case. For example, the case handling system FLOWer logs every modification of some data element.

Event logs such as the one shown in Table 1 are used as the starting point for mining. We distinguish three different perspectives: (1) the process perspective, (2) the organizational perspective and (3) the case perspective. The *process perspective* focuses on the control-flow, i.e. the ordering of activities. The goal of mining this perspective is to find a good characterization of all possible paths, e.g. expressed in terms of a Petri net or Event-driven Process Chain (EPC). The *organizational perspective* focuses on the originator field, i.e. which performers are involved and how are they related. The goal is to either structure the organization by classifying people in terms of roles and organizational units or to show relations between individual performers (i.e. build a social network). The *case perspective* focuses on properties of cases, i.e. the values of the corresponding data elements. For example, if a case represents a replenishment order it is interesting to know the supplier or the number of products ordered.

[1] Note that in Table 1 we abstract from *event types*, i.e. we consider activities to be atomic. In real logs events typically correspond to the start or completion of an activity. This way it is possible to measure the duration of activities and to explicitly detect parallelism. Moreover, there are other event types related to failures, scheduling, delegations, etc. For simplicity we abstract from this in this paper. However, in our process mining tools we take event types into account.

case id	activity id	originator	timestamp
case 1	activity A	John	9-3-2004:15.01
case 2	activity A	John	9-3-2004:15.12
case 3	activity A	Sue	9-3-2004:16.03
case 3	activity B	Carol	9-3-2004:16.07
case 1	activity B	Mike	9-3-2004:18.25
case 1	activity C	John	10-3-2004:9.23
case 2	activity C	Mike	10-3-2004:10.34
case 4	activity A	Sue	10-3-2004:10.35
case 2	activity B	John	10-3-2004:12.34
case 2	activity D	Pete	10-3-2004:12.50
case 5	activity A	Sue	10-3-2004:13.05
case 4	activity C	Carol	11-3-2004:10.12
case 1	activity D	Pete	11-3-2004:10.14
case 3	activity C	Sue	11-3-2004:10.44
case 3	activity D	Pete	11-3-2004:11.03
case 4	activity B	Sue	11-3-2004:11.18
case 5	activity E	Clare	11-3-2004:12.22
case 5	activity D	Clare	11-3-2004:14.34
case 4	activity D	Pete	11-3-2004:15.56

Table 1. An event log

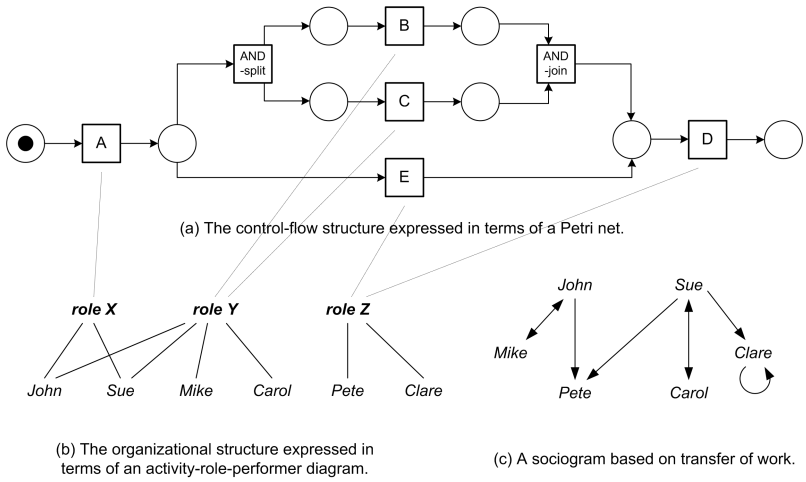

Fig. 2. Some mining results for the process perspective (a) and organizational (b and c) perspective based on the event log shown in Table 1

The process perspective is concerned with the "How?" question, the organizational perspective is concerned with the "Who?" question, and the case perspective is concerned with the "What?" question. To illustrate the first two consider Figure 2. The log shown in Table 1 contains information about five cases (i.e. process instances). The log shows that for four cases (1, 2, 3, and 4) the activities A, B, C, and D have been executed. For the fifth case only three activities are executed: activities A, E, and D. Each case starts with the execution of A and ends with the execution of D. If activity B is executed, then also activity C is executed. However, for some cases activity C is executed before activity B. Based on the information shown in Table 1 and by making some assumptions about the completeness of the log (i.e. assuming that the cases are representative and a sufficiently large subset of possible behaviors is observed), we can deduce the process model shown in Figure 2(a). The model is represented in terms of a Petri net [18]. Note that for this example we assume that two activities are in parallel if they appear in any order. By distinguishing between start events and complete events for activities it is possible to explicitly detect parallelism.

As table 1 shows the performers of activities, we can use this information to discover the organizational structure. For every activity, there is a distinctive set of persons that have performed it. Rather than extending Figure 2(a) with this information, we can also use it to discover organizational structures. Comparing the sets of persons that have executed each activity can yield the underlying *roles* of the process model, in this example we can discover three roles X, Y, and Z. For example, for the execution of A role X is required and John and Sue have this role. The resulting "activity-role-performer diagram" is shown in Figure 2(b). The three "discovered" roles link activities to performers. For five cases these choices may seem arbitrary but for larger data sets such inferences capture the dominant roles in an organization.

The event log can be used to derive a sociogram based on the transfer of work from one individual to another as is shown for the example in Figure 2(c). Here the focus is not on the relation between the process and individuals but on relations among (groups of) individuals. Each node represents one of the six performers and each arc represents that there has been a transfer of work from one individual to another, i.e. whether for the same case an activity executed by A is directly followed by an activity executed by B. For example, both in case 1 and 2 there is a transfer from John to Mike. Figure 2(c) does not show frequencies. However, for analysis purposes these frequencies can be added. The arc from John to Mike would then have weight 2. Typically, we do not use absolute frequencies but weighted frequencies to get relative values between 0 and 1.

The case perspective looks at the case as a whole and tries to establish relations between the various properties of a case (i.e., the "What?" question). Focusing on the case perspective is most interesting when also data elements are logged but these are not listed in Table 1, and therefore this perspective is not addressed in Figure 2. Note that some of the properties may refer to the activities being executed, the performers working on the case, and the values of

various data elements linked to the case. Using clustering algorithms it would for example be possible to show a positive correlation between the size of an order or its handling time and the involvement of specific people.

Orthogonal to the three perspectives (process, organization, and case), the result of a mining effort may refer to *logical* issues and/or *performance* issues. For example, process mining can focus on the logical structure of the process model (e.g. the Petri net shown in Figure 2(a)) or on performance issues such as flow time. For mining the organizational perspectives, the emphasis can be on the roles or the social network (cf. Figure 2(b) and (c)) or on the utilization of performers or execution frequencies.

To address the three perspectives and the logical and performance issues we have developed a set of tools including EMiT, Thumb, and MinSoN sharing a common XML format. Recently, these tools have been merged into the *ProM framework* (see http://www.processmining.org for more details).

4 Extracting Logs From a Case Handling System

In order to analyze business processes using process mining techniques it is essential to retrieve suitable event logs from the system in question. A case handling system logs, like workflow management systems, the occurrence of activities (i.e. executed tasks). However, further *it also provides detailed log information about events referring to any modification of a defined data object* included in the case. In the remainder of this section we discuss the information extractable from case handling system logs.

4.1 Activity Logs

A case handling system keeps much closer track of the events related to activities than many traditional WFMSs, virtually logging every state change in the life-cycle of activities. Whenever an activity becomes enabled, someone starts or stops working on it, or it is finished, a log entry with the appropriate information is created. Of course events like skipping or redoing an activity also get recorded in the logs. This detailed logging allows for a far more comprehensible understanding of what happened during process enactment, the actual workflow can be concisely traced back to user interactions.

4.2 Data Logs

Following its paradigm of more closely intertwining the data and control flow aspect of business processes, case handling systems do not only log events related to activities but also manipulation of case data elements. If these events are considered as activities within a regular process it is possible to mine a process model from such a data log, featuring a fine-grained model of data access events as tasks.

Data logs may also be exploited by performing statistical analysis techniques on them, thereby exposing typical access patterns or general characteristics of data usage within the handling of a certain case type. Specific access patterns can also be used as indicators for inadequately designed processes (e.g. if rolling back a certain task is not an exception but rather the rule). Moreover, it is possible to use data as a starting point for process design. As suggested by the Product Based Workflow Design (PBWD) approach [28, 1], one can structure information in a hierarchical structure similar to the Bill of Material used in manufacturing. It is worthwhile to emphasize that, besides case handling systems, many systems log updates on data elements. Consider for example a Product Data Management (PDM) system like Windchill (http://www.ptc.com/), logging the status of every design artifact (e.g. a technical drawing). ERP systems like SAP R/3 also use the concept of document flows to record events at the business level, i.e. not the process but some document is the key concept when logging events.

4.3 Unified Logs

To unleash the full potential of process mining in case handling systems it is necessary to also take into account its characteristics of handling control flow and data in a unified manner, namely by analyzing those in a respectively combined fashion. Since activities can also be perceived as abstractions of bulk data processing and modification rather than fine-grained data updates, the data perspective and the process perspective reside on different levels of abstraction. In combining logs from both perspectives, care has to be taken in order to not distort the contained information. The order of events within the unified log can be derived from the events' timestamps. Whenever two events from different layers are in conflict, the semantic implications should be taken into account to resolve it (e.g. starting work on an activity is naturally preceding the consequent setting of a data object). It is also important to mark each event, so that it can later be identified to either stem from the activity or data log. Only then can interrelations between both perspectives be extracted in a meaningful way.

5 FLOWer and ProM

After introducing the concepts of process mining and case handling, and exploring their relationships, we consider two concrete software systems: FLOWer as an example of a case handling product and ProM as an example of a process mining tool.

Case handling as a paradigm of business process management is relatively novel, thus it is not surprising that no large number of actual case handling system implementations can be found in practice. For our research we are at the moment focusing on the commercial case handling system FLOWer, which is introduced in the next section.

Process mining is closely linked to tools advertised through buzzwords such as Business Activity Monitoring (BAM) and Business Process Intelligence (BPI).

However, most of the commercial tools focus on simple performance indicators such as flow times and frequencies. A notable exception is ARIS PPM [22] of IDS Scheer. Compared to commercial tools, the functionality of ProM is far superior when it comes to process mining. Therefore, we focus our attention on bridging the gap between FLOWer and ProM. Note that ProM can export its results to ARIS PPM for alternative (i.e. more traditional) ways of analysis.

5.1 FLOWer

FLOWer is a commercial case handling system manufactured by the Dutch company Pallas Athena. It implements the case handling paradigm very close to its theoretical foundations, rendering it a highly relevant object of research in this field.

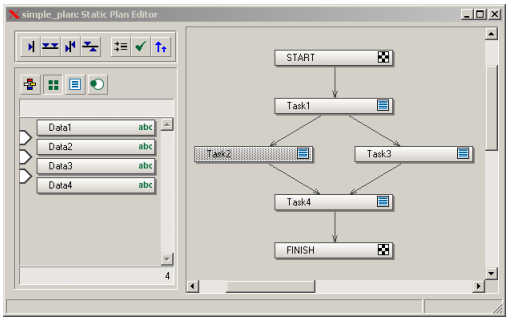

Fig. 3. Screenshot of the FLOWer Process Designer

The FLOWer process designer, shown in Figure 3, features a vast set of control flow constructs that can largely be mapped on the well-known control flow patterns [4]. For the sake of performing process mining, FLOWer requires the process designer to explicitly activate the recording of management information for both tasks and data elements at design time.

5.2 ProM

In order to foster development in the process mining domain and to achieve a maximal degree of interoperability between previously existing and newly developed methods and programs a common XML based format[2] for storing process logs has been established [7]. To further proceed on this path to combine efforts the ProM framework has been developed, providing a common base for the development of techniques related to the field of process mining.

[2] See www.processmining.org.

The key property of the ProM framework is its highly modular architecture, incorporating mining algorithms and methods in the form of plugins. The framework itself provides a common graphical user interface, high-level data structures and methods for working with process logs and models, and also mechanisms for visualizing mined process models using an array of formats.

After selecting a log file and the plugin which is to be used for its analysis, the plugin-specific options for mining can be configured. The result can then be visualized in different ways, including graph-based visualizations like Petri nets or EPCs. Depending on the result of the plugin the outcome of mining can be analyzed further, e.g. by checking certain properties like soundness of a mined Petri net.

5.3 FLOWer Log Extraction

The first step on the agenda of performing process mining in case handling systems is to acquire event logs from such a system. FLOWer stores event logs not in traditional text file format, but it rather creates for each event an entry in one or more database tables used for storing management information. While the log information stored in there is very comprehensive, containing a vast array of meta-data, at the same time it is quite hard to access and extract in a useful manner. For a process mining project within the UWV (a large Dutch company using FLOWer), Ana Karla Alves de Medeiros developed an initial version of a converter from FLOWer to our XML format. Based on this prototype, a new, more versatile, and easily extensible import filter has been developed. This filter directly accesses the log database and is at the moment able to extract log information for both task and data events for each case type stored there. At this point in time it is not yet possible to create a unified log containing both data and task events, however, this feature is in development and should be available in the near future.

As FLOWer allows to structure case types, regarding their process perspective, using sub-plans (i.e. sub-processes) both task and data events can be located at multiple "levels" per case type. Thus, it has been made configurable whether events on all levels of a case type shall be included within one single log file, or whether the events shall be grouped in multiple files on a per-level basis. An auxiliary level report provides information about the level structure of case types in form of a plain text file. As the ProM framework allows for loading XML log files also from compressed ZIP archives, all files created in one import session are saved within one such archive, underlining their logical coupling.

Further options include the prioritization and exclusion of certain case types according to their ID. Regarding the effort that is needed to extract FLOWer logs, due to a massive amount of database queries and demanding post-processing necessary, this proves to be a useful feature in saving processing time and concentrating on certain case types in the focus of attention.

5.4 Mining FLOWer logs with ProM

A case handling system can be perceived as superset of a traditional workflow management system, regarding its possibilities, and thus the behavior potentially generated by it. While log traces produced by a WFM system will, presuming no errors or cheats during enactment, be bound to follow the possibilities prescribed by the process model, this is absolutely not true for case handling processes. Here the case type does not prescribe actions but rather limits the set of potential tasks to a reasonable subset, so that the user always has far more opportunities to deviate from the standard path suggested. Further, the capability of FLOWer to skip and redo tasks makes it possible to navigate through the process back and forth, thus allowing for a previously unknown degree of freedom. Loops do no longer have to be predefined, and as such be easy to trace back to the model, they can be performed at any point within the process given the fact that appropriate roles do apply.

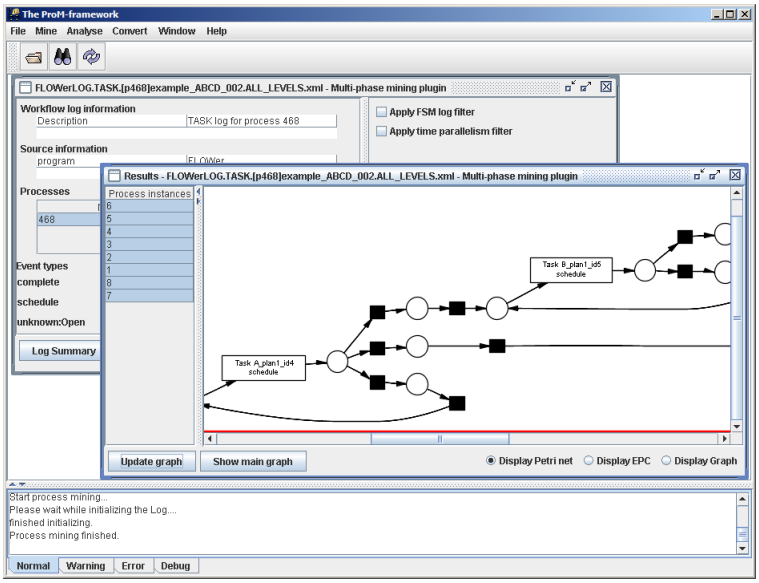

Fig. 4. Screenshot of the ProM framework analyzing a FLOWer task log

Figure 4 shows a screenshot of ProM, with the result of applying the multi-phase mining algorithm [19] to a FLOWer task log of a very simple example case type of four tasks. In the test executions of this case type excessive use has been made of the flexibility offered by FLOWer, and the mining result reflects this by e.g. including a vast number of task-skipping and returning arcs. It is obvious that process mining techniques originally developed for mining logs created by a WFM system can also be used for analyzing case handling logs. Given a suitable process and a conservative fashion of enactment this may be helpful in providing

insight to case type and organization and prove sufficient. However, when users make full use of the flexibility and freedom offered by a case handling system the results of mining are prone to making interpretation hard. Fortunately, some mining plug-ins such as the Social Network Analyzer (SNA) [6] are more robust in this respect.

6 Discussion

The work that has been performed so far, regarding the application of process mining to case handling systems, is still in its infancy. As described above, an import filter has been created that allows for the extraction of comprehensive log information from the case handling system FLOWer. The following section points out directions in which development of import facilities for ProM continues, introducing the new generic import filter framework. In the subsequent section a selection of approaches dealing with the combined, or "holistic", analysis of both task and data event logs are presented, which we plan to pursue.

6.1 Log Import Filter Framework

So far there existed several converter programs for retrieving the necessary information from the respective system and delivering a process log in the common XML format for ProM. There are converters for Staffware and MQSeries logs, and also semi-automatic tools supporting the extraction of SAP document flows. From a software engineering point of view the separate converters form a rather undesirable situation, as all of these import filters share a great set of common functionality. For example, the fact that all of them have their own methods for writing the XML format is a serious problem with respect to code maintenance.

This drawback has been tackled by the development of a generic framework for log import filters, providing common tools and means that are used repeatedly. For example, not only writing the logs has been encapsulated by the framework, also configuring import filters and keeping this configuration persistent is handled transparently by the framework. Import filter implementations can fully concentrate on the plain logic for log extraction, with the framework providing a common, user-friendly GUI. They can be loaded as plugins at runtime, which makes the architecture especially flexible and extensible.

One feature worth mentioning is the *log anonymizer* that is built into the framework and can be used for any import filter in a transparent manner. Using this tool, all properties of a log, e.g. originator names or task identifiers, can be exchanged by meaningless ID strings. Of course this replacement is performed in a way that preserves the identity of anonymized items (i.e. all occurrences of one identifier are replaced by the same anonymizer string).

The core incentive has been to make writing further import filters as straightforward as possible. Another goal is to provide third parties with a powerful and yet easy-to-use tool for collecting actual logs in practice, in order to facilitate the testing of mining algorithms on more practical examples. The availability

of anonymization is expected to significantly lower the reluctancy of users to exchange logs.

6.2 Holistic Approaches for Log Analysis

The fact that a combined case handling log includes data- and task-related events creates a somewhat novel situation. There are no longer just different perspectives of the same data but such unified case handling logs effectively contain the same process on two different, yet closely related, layers of abstraction. Apart from analyzing data and activity logs separately, which makes perfectly sense in certain settings, it appears more promising to look at both data and activity events in a holistic fashion, taking into account the relations between them which are inherently driving a case handling system.

For performing a case type verification the starting point would be to first mine the control flow perspective and derive an appropriate process model, incorporating phenomena like skipping and redoing tasks in an implicit manner. The next step is to assign data modification events to the realm of tasks. As in a case handling system data can basically only be manipulated between the events of starting and finishing a task, and as these events are explicitly recorded in the activity logs, this assignment of data manipulation events to tasks is quite straightforward. However, data modification events that have resulted from modifying case forms (that can be accessed at any time) need to be interpreted appropriately to avoid distortion. The result of this complete procedure can be thought of as a (simplified) process model, suggesting for each task within the process a set of data elements as mandatory, following the pattern of data manipulation events recorded in the logs. Such an enhanced process model could then be compared to the original case type definition to see whether what the case type designer intended is really the way cases are handled within an organization.

In order to optimize the coupling and separation of work into activities, a holistic mining approach can also investigate e.g. the idle time between successive tasks. If there is an overwhelming majority of cases where two successive tasks A and B are finished with practically no delay this might be a hint that they should better be clustered into one single task. Correspondingly, in case one task is always interrupted and resumed later on it seems to be a good idea to think about splitting it into two separate tasks.

The above approach is tailored towards business process improvement, and as such a rather informal process representation, e.g. following the way FLOWer depicts its case type plans, seems most appropriate. However, given the fact that we are dealing with process models that are closely intertwined with data manipulation information, the modelling technique of Colored Petri Nets (CPN) [23] seems to be an obvious choice. It allows for a concise graphical representation of data flow by assigning potentially structured data types ("colors") to places and tokens. As described above it should be possible to derive from case handling logs the information, which data objects are typically modified during each task. Starting from this, each place in the CPN process model could be assigned

a specific color which is assembled from all data elements written by the preceding and read by the subsequent task, i.e. transition. The derivation of a CPN process model from unified activity and data logs is anticipated to be a complex endeavor. Nevertheless, given its formal semantics, the existence of excellent tool support for their verification and analysis, and the good fit with the case handling paradigm, the CPN language is considered to be the foundation for our future efforts on process mining in case handling systems.

7 Related Work

Flexibility has been a "hot research topic" in workflow literature since the late nineties [3, 5, 11, 16, 20, 21, 24, 27, 29, 32]. Flexibility triggers all kinds of interesting research questions, e.g. if one changes a process what to do with the running cases? [5]. However, the idea of case handling, as it has been defined in [2, 10], is to avoid this. These ideas have been successfully implemented in FLOWer (Pallas Athena, [14, 13]) and have proven to be valid in many practical applications. FLOWer has been applied in payment institutions, banks, insurance companies, government bodies, telecommunications, housing corporations, educational institutions, health care, police, courts of law, and IT companies.

The idea of applying process mining in the context of workflow management was first introduced in [12]. Cook and Wolf have investigated similar issues in the context of software engineering processes using different approaches [17]. It is impossible to point out the many process mining algorithms proposed in literature. However, we would like to mention the α algorithm [9] which served as a starting point for the ProM framework. For more information on process mining we refer to a special issue of Computers in Industry on process mining [8] and a survey paper [7].

As far as we know this is the first paper focusing on the link between process mining and case handling, which explains its explorative nature.

8 Acknowledgements

This research is supported by the Technology Foundation STW, applied science division of NWO and the technology programme of the Dutch Ministry of Economic Affairs.

The authors would like to thank Ana Karla Alves de Medeiros for implementing the first converter from FLOWer to ProM. We would also like to thank her and the rest of the "process mining team" (among others Ton Weijters, Boudewijn van Dongen, Minseok Song, Laura Maruster, Eric Verbeek, Monique Jansen-Vullers, Hajo Reijers, Michael Rosemann, Anne Rozinat and Peter van den Brand) for their on-going work on process mining techniques. Parts of this paper have been based on earlier papers with these researchers.

References

1. W.M.P. van der Aalst. On the automatic generation of workflow processes based on product structures. *Computers in Industry*, 39:97–111, 1999.
2. W.M.P. van der Aalst and P.J.S. Berens. Beyond Workflow Management: Product-Driven Case Handling. In S. Ellis, T. Rodden, and I. Zigurs, editors, *International ACM SIGGROUP Conference on Supporting Group Work (GROUP 2001)*, pages 42–51. ACM Press, New York, 2001.
3. W.M.P. van der Aalst, J. Desel, and A. Oberweis, editors. *Business Process Management: Models, Techniques, and Empirical Studies*, volume 1806 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 2000.
4. W.M.P. van der Aalst, A.H.M. ter Hofstede, B. Kiepuszewski, and A.P. Barros. Workflow Patterns. *Distributed and Parallel Databases*, 14(1):5–51, 2003.
5. W.M.P. van der Aalst and S. Jablonski. Dealing with Workflow Change: Identification of Issues and Solutions. *International Journal of Computer Systems, Science, and Engineering*, 15(5):267–276, 2000.
6. W.M.P. van der Aalst and M. Song. Mining Social Networks: Uncovering Interaction Patterns in Business Processes. In J. Desel, B. Pernici, and M. Weske, editors, *International Conference on Business Process Management (BPM 2004)*, volume 3080 of *Lecture Notes in Computer Science*, pages 244–260. Springer-Verlag, Berlin, 2004.
7. W.M.P. van der Aalst, B.F. van Dongen, J. Herbst, L. Maruster, G. Schimm, and A.J.M.M. Weijters. Workflow Mining: A Survey of Issues and Approaches. *Data and Knowledge Engineering*, 47(2):237–267, 2003.
8. W.M.P. van der Aalst and A.J.M.M. Weijters, editors. *Process Mining*, Special Issue of Computers in Industry, Volume 53, Number 3. Elsevier Science Publishers, Amsterdam, 2004.
9. 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.
10. W.M.P. van der Aalst, M. Weske, and D. Grünbauer. Case Handling: A New Paradigm for Business Process Support. *Data and Knowledge Engineering*, 53(2):129–162, 2005.
11. A. Agostini and G. De Michelis. Improving Flexibility of Workflow Management Systems. In W.M.P. van der Aalst, J. Desel, and A. Oberweis, editors, *Business Process Management: Models, Techniques, and Empirical Studies*, volume 1806 of *Lecture Notes in Computer Science*, pages 218–234. Springer-Verlag, Berlin, 2000.
12. 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.
13. Pallas Athena. *Case Handling with FLOWer: Beyond workflow*. Pallas Athena BV, Apeldoorn, The Netherlands, 2002.
14. Pallas Athena. *Flower User Manual*. Pallas Athena BV, Apeldoorn, The Netherlands, 2002.
15. BPi. *Activity Manager: Standard Program - Standard Forms (Version 1.2)*. Workflow Management Solutions, Oosterbeek, The Netherlands, 2002.
16. F. Casati, S. Ceri, B. Pernici, and G. Pozzi. Workflow Evolution. In *Proceedings of ER '96*, pages 438–455, Cottubus, Germany, Oct 1996.
17. 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.

18. J. Desel, W. Reisig, and G. Rozenberg, editors. *Lectures on Concurrency and Petri Nets*, volume 3098 of *Lecture Notes in Computer Science*. Springer-Verlag, Berlin, 2004.
19. 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.
20. C.A. Ellis and K. Keddara. A Workflow Change Is a Workflow. In W.M.P. van der Aalst, J. Desel, and A. Oberweis, editors, *Business Process Management: Models, Techniques, and Empirical Studies*, volume 1806 of *Lecture Notes in Computer Science*, pages 201–217. Springer-Verlag, Berlin, 2000.
21. T. Herrmann, M. Hoffmann, K.U. Loser, and K. Moysich. Semistructured models are surprisingly useful for user-centered design. In G. De Michelis, A. Giboin, L. Karsenty, and R. Dieng, editors, *Designing Cooperative Systems (Coop 2000)*, pages 159–174. IOS Press, Amsterdam, 2000.
22. IDS Scheer. ARIS Process Performance Manager (ARIS PPM): Measure, Analyze and Optimize Your Business Process Performance (whitepaper). IDS Scheer, Saarbruecken, Gemany, http://www.ids-scheer.com, 2002.
23. K. Jensen. *Coloured Petri Nets. Basic Concepts, Analysis Methods and Practical Use*. EATCS monographs on Theoretical Computer Science. Springer-Verlag, Berlin, 1992.
24. M. Klein, C. Dellarocas, and A. Bernstein, editors. *Adaptive Workflow Systems*, volume 9 of *Special issue of the journal of Computer Supported Cooperative Work*, 2000.
25. London Bridge Group. *Vectus Application Developer's Guide*. London Bridge Group, Wellesbourne, Warwick, UK, 2001.
26. London Bridge Group. *Vectus Technical Architecture*. London Bridge Group, Wellesbourne, Warwick, UK, 2001.
27. M. Reichert and P. Dadam. ADEPTflex: Supporting Dynamic Changes of Workflow without Loosing Control. *Journal of Intelligent Information Systems*, 10(2):93–129, 1998.
28. H.A. Reijers, S. Limam, and W.M.P. van der Aalst. Product-based Workflow Design. *Journal of Management Information systems*, 20(1):229–262, 2003.
29. S. Rinderle, M. Reichert, and P. Dadam. Correctness Criteria For Dynamic Changes in Workflow Systems: A Survey. *Data and Knowledge Engineering*, 50(1):9–34, 2004.
30. Software-Ley. *COSA Activity Manager*. Software-Ley GmbH, Pullheim, Germany, 2002.
31. Staffware. *Staffware Case Handler – White Paper*. Staffware PLC, Berkshire, UK, 2000.
32. M. Weske. Formal Foundation and Conceptual Design of Dynamic Adaptations in a Workflow Management System. In R. Sprague, editor, *Proceedings of the Thirty-Fourth Annual Hawaii International Conference on System Science (HICSS-34)*. IEEE Computer Society Press, Los Alamitos, California, 2001.