USING PROCESS MINING FOR ITIL ASSESSMENT: A CASE STUDY WITH INCIDENT MANAGEMENT

Diogo R. Ferreira[1,2], Miguel Mira da Silva[1,3]
[1]*IST – Technical University of Lisbon, Portugal*
[2]*Organizational Engineering Center, INESC-INOV*
[3]*Information Systems Group, INESC-ID*
Email: {diogo.ferreira, mms}@ist.utl.pt

Abstract
The ITIL framework is the best known set of best practices for managing IT services. In this paper we introduce process mining as a useful technique for assessing whether a business process is implemented according to ITIL guidelines. We evaluated our approach using a real-world case study in an IT vendor developing a complex software platform. The company receives thousands of requests (including bug reports) that can be treated as ITIL incidents. Using process mining, it was possible to extract the behaviour of the existing process and compare it with ITIL Incident Management.

Keywords: ITIL Assessment, Process Mining, Incident Management

Main Author:	Diogo R. Ferreira
Address:	Instituto Superior Tecnico Avenida Prof. Dr. Cavaco Silva 2780-990 Porto Salvo Portugal
Phone:	+351 21 423 35 52
Fax:	+351 21 423 32 68
E-mail:	diogo.ferreira@ist.utl.pt

Presenting Author:	Miguel Mira da Silva
Address:	INESC Rua Alves Redol, 9 1000-029 Lisboa
Phone:	+351 91 967 14 25
E-mail:	mms@ist.utl.pt

Word Count:	ca. 5000 (max.8000)
No. pages:	16 (max.16)

USING PROCESS MINING FOR ITIL ASSESSMENT: A CASE STUDY WITH INCIDENT MANAGEMENT

Abstract
The ITIL framework is the best known set of best practices for managing IT services. In this paper we introduce process mining as a useful technique for assessing whether a business process is implemented according to ITIL guidelines. We evaluated our approach using a real-world case study in an IT vendor developing a complex software platform. The company receives thousands of requests (including bug reports) that can be treated as ITIL incidents. Using process mining, it was possible to extract the behaviour of the existing process and compare it with ITIL Incident Management.

Keywords: ITIL Assessment, Process Mining, Incident Management

1 Introduction

Requirements imposed by new legislation and regulation acts, such as internal control requirements in Sarbanes-Oxley (Zhang, 2007) or risk management requirements in Basel II (Porter, 2003), are increasing the pressure on companies to ensure that some of their key business processes are either being performed according to plan or adhere to common practices set by industry standards. The Information Technology Infrastructure Library (ITIL) (van Bon et al, 2005) sets standard best practices for IT Service Management, ranging from service level and capacity management to incident management, and to configuration, release and change management. However, these best practices are defined as a set of processes that are intentionally non-prescriptive in order to fit into many different kinds of organizations. This flexibility makes ITIL difficult to assess, as some ITIL processes are difficult to translate to operational models (Brenner, 2006).

Checking whether an existing process is a correct and complete implementation of ITIL guidelines will be easier if that process is clearly defined. But in practice, many IT service management processes rely on one or more supporting systems which happen to be used without careful guidance or control, allowing for inconsistent content and incoherent behaviour from different users. Such is the case study presented in this paper, where the main supporting system for incident management allows users to handle incidents and populate the system database freely according to their own needs, practices and workarounds.

To assess processes such as this one, it is necessary to study their run-time behaviour. Whereas in the past capturing business processes would require time-consuming interviews with possibly unreliable results, recent developments in the field of process mining (van der Aalst et al, 2003) provide specialized techniques to automatically discover process behaviour from system logs. These techniques are only limited by the type and amount of data that is possible to collect from the underlying support systems.

In section 2 we introduce ITIL with a special emphasis on the Incident Management process, which will be the focus of the case study. In section 3 we provide a brief overview of process mining techniques. Section 4 introduces the case study, and section 5 presents the main results. Section 6 concludes by reinterpreting the results in terms of ITIL Incident Management.

2 The ITIL framework

ITIL offers a framework for managing most of the operational activities of an IT service provider. ITIL divides these activities into processes and describes how these processes can be optimized based on industry best practices (van Bon et al, 2005). According to the literature, ITIL offers a number of benefits: the provision of IT services becomes more customer-focused, the relationship between IT provider and customer is improved, the services are better described, and the quality, availability, reliability, and cost of the services are better managed.

Unfortunately, implementing ITIL by following the recommendations has proved to be a difficult challenge for many IT organizations (Mendel et al, 2004). Most organizations have reported a long time and significant effort and cost; others claim that services were not improved or cost was not reduced after introducing ITIL; and some departments that introduced ITIL were then left alone by other departments that decided not to introduce ITIL. It is therefore desirable to begin an ITIL implementation by assessing existing processes.

2.1 ITIL Assessment

When an IT department decides to implement ITIL, the existing IT activities should be assessed (or evaluated) in order to identify the gaps between the current processes – i.e. the way IT management is currently performed – and the processes as described by ITIL (Litten, 2005). These gaps should then be used to identify requirements for the ITIL implementation project. Gaps may exist in a number of areas, including process, people, technology, services supplied by third parties, or any combination of these. For example, a gap exists if not all incidents are recorded. Another exists if incidents are not classified.

For the purpose of this paper, as ITIL is mainly based on processes, the assessment will concentrate on the design of existing processes. In particular, the existing processes should be compared to processes as proposed by ITIL. If they are different, the ITIL implementation project should focus on closing those gaps.

Unfortunately, capturing an existing process can become unexpectedly difficult. The processes may have been formally defined based on ITIL but, in practice, no one follows the definition; or there may be a tool to support those processes, but IT managers use the tool "in their own way". We have seen these and other problems in real-world scenarios.

Process mining can help discover business processes based on actual run-time data and, as such, it can provide valuable input to ITIL assessment. Once the true behaviour is discovered, then it becomes easier to identify the gaps between the current process and the ITIL guidelines. Ahead we will illustrate this potential in a case study focusing on Incident Management.

2.2 Incident Management

An incident is any event which is not part of the standard operation of a service and which may cause an interruption or reduction in the quality of that service. The purpose of Incident Management is to return to the normal service level as soon as possible by mitigating or eliminating the effects of disturbances in IT services. Incident Management provides immediate benefits, and is sometimes proposed as the

first process to be implemented (Mendel, 2004). The process itself comprises the following main steps:

1. Recording: upon reception, the incident must be recorded.
2. Classification: the incident is characterized in terms of type, impact and urgency, leading to a certain priority class.
3. Matching: a solution may already exist if the incident matches a known problem or error condition.
4. Diagnosis: all available information about the incident is gathered in order to investigate and determine a solution or workaround.
5. Resolution: the solution is applied in order to restore normal service or system operation.
6. Closure: the incident is closed once the service has been restored.

During incident diagnosis, successive levels of service support may be invoked until a solution or workaround is found. This behaviour is known as *escalation* – if the current support level is unable to find a solution, then the incident escalates to the next (higher) support level.

If, despite going through all support levels, the incident cannot be solved – such is the case if there is a defect in an underlying component or infrastructure, for example – then the incident may have to be handled within the scope of other ITIL processes, such as Problem Management or Change Management. In such cases the goal is not merely to restore normal service, but to identify and correct the underlying causes for one or more incidents. The solution for such problems may require bug fixes or system upgrades, for example.

Incident Management may therefore be the entry point for other ITIL processes. That is precisely the scenario in the case study presented ahead. In that scenario, the purpose of the existing process is to handle product issues detected by end users. Some of these can be solved immediately, while others may go to the point of requiring product changes. In both cases, issues follow basically the same process. The difficulty is to find out whether that process complies with Incident Management, and it is in this context that process mining techniques become extremely useful.

3 Process Mining and Conformance

Current transactional information systems – such as ERP (Enterprise Resource Planning), SCM (Supply Chain Management), CRM (Customer Relationship Management) and even BPM (Business Process Management) systems – are able to record large amounts of run-time activity, and there is an enormous potential in using that recorded information to derive meaningful conclusions about the behaviour of business processes. Such analysis may be conducted to check that certain business processes are actually being performed according to plan, to detect anomalous and exceptional behaviour, or even to capture business processes that have never been explicitly designed.

The tools required for such analysis are being studied and developed within the field of process mining (van der Aalst & Weijters, 2004). Currently, these tools are able to extract control-flow models (van der Aalst et al, 2003), data dependencies (Rozinat et al, 2006), and even social network models (van der Aalst et al, 2005). Process mining is an active and promising research field, and already includes a number of different techniques.

3.1 Process mining techniques

In general, all process mining techniques take an event log as the starting point for the discovery of processes behaviour. The event log is a list of records resulting from the execution of some process, where each record contains information about the activity that was executed, the process instance that it belongs to, and the time of execution. With this information different techniques may be applied, namely:

- Directed acyclic graphs (Agrawal et al, 1998) – a technique that is able to generate a dependency graph from an event log.
- Inductive workflow acquisition (Herbst & Karagiannis, 1998) – an approach in which the goal is to find a hidden Markov model (HMM) that best represents the structure of the original process.
- The α-algorithm (van der Aalst et al, 2004) – a technique that is able to re-create a Petri-net model from the ordering relations found in an event log.
- Instance graphs (van Dongen & van der Aalst, 2004) – an approach that aims at portraying a graphical representation of process executions, especially using Event-Driven Process Chains (EPCs).

- Hierarchical clustering (Greco et al, 2005) – an algorithm that, given a large set of execution traces of a single process, separates them into clusters and finds the dependency graph separately for each cluster.
- Genetic algorithms (Medeiros et al, 2007) – a technique in which several candidate solutions are evaluated by a fitness function that determines how consistent each solution is with the log.
- Sequence clustering (Ferreira et al, 2007) – a technique that automatically groups sequences into different clusters in order to identify typical behavioural patterns.
- Negative events (Goedertier et al, 2008) – an approach in which the original log is complemented with negative events in order to enable the use of first-order machine learning techniques.

Some of these techniques rely on finding causal relations in the log: task A can be considered to be the cause of task B only if B follows A, but A never follows B. Such techniques are sensitive to noise (van der Aalst et al, 2004). The need to cope with noise has drawn some attention to other techniques such as sequence clustering (Ferreira et al, 2007).

In the case study presented below, we have used the Microsoft Sequence Clustering algorithm (Tang & MacLennan, 2005). Each cluster is represented as a first-order Markov chain, where the current state depends only on the previous state. The probability that an observed sequence belongs to a given cluster is the probability that the observed sequence was produced by the Markov chain associated with that cluster. The algorithm itself is an iterative expectation-maximization procedure (Cadez et al, 2003) that assigns traces to clusters and re-estimates the cluster parameters until they converge. The end result is a set of clusters that represents different behavioural patterns.

3.2 From process mining to process conformance

Besides facilitating the discovery of business processes, the above techniques can also be used to check the conformance of run-time behaviour with respect to a given process model. Conformance can be checked by using metrics (Rozinat & van der

Aalst, 2008) to determine the extent to which the behaviour observed in the event log complies with the control flow specified by the process model. Such analysis is performed by establishing a one-to-one mapping between log entries and activities in the process model, and then checking if the process model would allow each recorded trace to occur.

In our case study, the amount of noise and ad-hoc behaviour, together with the fact that the process model for Incident Management is not rigorously defined, did not allow us to perform such conformance analysis. Nevertheless, by finding the typical behaviour and matching that behaviour to the ITIL guidelines it was possible to draw meaningful conclusions about the conformance of the observed process.

4 Case Study

Our case study is an IT company that offers an advanced platform to facilitate and accelerate the development of custom software applications. The platform is being improved continuously by successive release versions that add new functionality, improve existing features, and correct bugs. Besides extensive manual and automated in-house testing, end users also have an active role in pointing out desired improvements and problems to be solved.

To keep track of all these issues and to handle them appropriately, the company developed a custom solution using its own software platform. The system – called Issue Manager – was developed mainly as a two-tier application having a Web-based interface and a back-end database, where it stores information regarding each issue (such as date, description, submitter, status, priority, risk, severity, etc.) along with all product versions where the issue was detected, as well as the relationships to other recorded issues. Most of these data can be filled with whatever the support team finds appropriate, except for the status field which is allowed to have one of a limited set of possible states.

During handling, the issue goes through a number of different states. Some of these states may actually be skipped for issues that can be solved immediately, while other issues may get to the point of generating a request for change, which will then trigger

a separate development process. Issue handling is, in itself, a process with all the characteristics of Incident Management, including connections to other processes that resemble Problem and Change Management. The goal is to determine how far the behaviour recorded by Issue Manager actually complies with Incident Management.

4.1 Available data

In the database we found that an issue can be in one of 15 possible states: Approved, Assigned, Closed, Discarded, Duplicated, NeedsApproval, NeedsSpecification, NeedsVerification, New, NotApproved, NotResolved, Open, Postponed, Resolved, Waiting. At the start of analysis, the semantics of these states were unknown, although their names provided some indication of their meaning. We also found that the database contained many interrelated tables that aimed at supporting a wide range of functionalities. An analysis of both the database schema and content revealed that there were two tables of interest:

- Table **issue** – contains general information about an issue such as a unique identifier, name, description, product version and date of submission, but also about the priority, severity, present state and who is currently assigned to handle the issue. There were 14,982 issues in the database.
- Table **history** – keeps track of all events where an event is a change of some sort, including change of assignment, change of priority, and change of state; may also contain an explanatory comment about the issue or about the changes made to an issue. There were 143,220 events in the database, roughly ten times as much as the number of issues.

With the data contained in the history table it was possible to build a useful data set for analysis. Basically, each sequence corresponds to the time-ordered list of state changes recorded for a given issue. The fact that the system allowed just about any kind of change to be freely made to an issue (not only state changes but also any other field change) means that the sequences obtained in this way displayed an arbitrary repetition of states when the changes were being made to other fields. For this reason, the sequence length was often longer than it would have been obtained if only the change in the state would be considered. These and other preprocessing steps were done before applying sequence clustering to the data set.

Figure 1 shows that sequence length varies widely, from issues with a single recorded event to issues with over 50 events. In fact, the longest sequence had 75 events, most of which were just a repetition of the "Waiting" status. Figure 1 also shows that most issues had sequence lengths between one and 15.

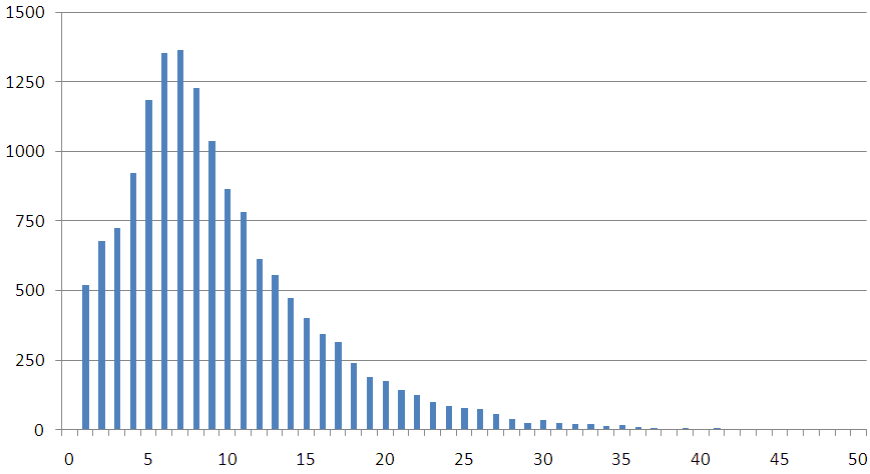

Figure 1. Number of issues vs. sequence length found in the history table

4.2 Preprocessing

The following preprocessing steps were applied to the data set:

1. *Drop events with low support* – Figure 2 shows the number of occurrences of each state in the history table. The states in the bottom of Figure 2 have low support since they occur only very rarely. Therefore, all events labelled as "NeedsSpecification" and "NotApproved" were discarded.
2. *Drop consecutively repeated events* – since many consecutive events were created by changes to fields other than state, they could be considered as a single event for our purposes. Around 63% of all events were eliminated in this step. The average sequence length also decreased dramatically, and there was an increase in the number of sequences with length between one and five.
3. *Drop sequences with either insufficient or excessive length* – Figure 1 shows that many sequences are actually non-sequences as they comprise a single event, so these sequences were removed. About 1,000 sequences were eliminated in this step.
4. *Drop sequences with repeated events* – a sequence that contains a (non-consecutive) repetition in a state represents a case where the handling of an

issue had to recede to a previous state. Sequences with such repetitions display a mixture of behaviour, which makes them difficult to assign correctly to a single cluster. About 2,500 sequences were eliminated in this step.

5. *Drop unique, one-of-a-kind sequences* – sequences that are unrepeatable are not interesting for the purpose of identifying typical behaviour. About 300 unique sequences were removed from the data set.

After these steps, 11,085 sequences remained, with a total of 35,778 events.

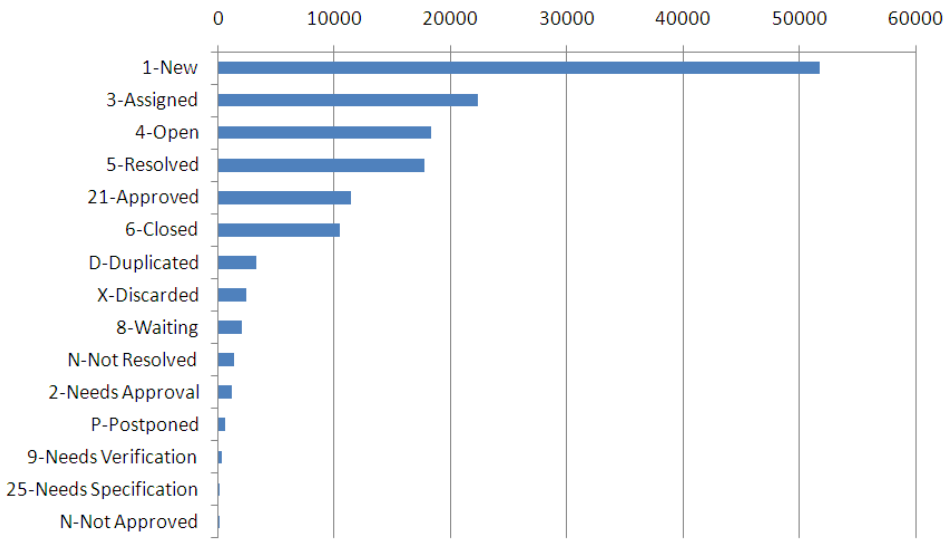

Figure 2. Number of occurrences of each status value in the history table

5 Results

A key parameter when applying the sequence clustering algorithm – or, for that matter, any kind of clustering algorithm – is the number of clusters to be used. For the Microsoft Sequence Clustering algorithm, this number of clusters can be specified either manually (i.e. by the user) or found automatically; in this case, the algorithm will make use of heuristics to find the ideal number of clusters for the given data.

Judging by the kind of sequences found in the input data set, a number of about 12 clusters seemed to be a good initial guess. After setting the parameters and running the algorithm on the data set, 14 clusters were created, but some of them displayed very similar behaviour. Running it again with different settings, the algorithm produced nine clusters, but still similar behaviour was observed in different clusters.

Figure 3 shows the most common sequences (top 5) in each of the nine clusters found; cluster 9 shows less sequences because it has only two types of sequences. The top sequences in clusters 3, 4, 6 and 8 are clearly related, and other sequences within different clusters were also found to be similar. The results suggested than the number of clusters should be decreased further.

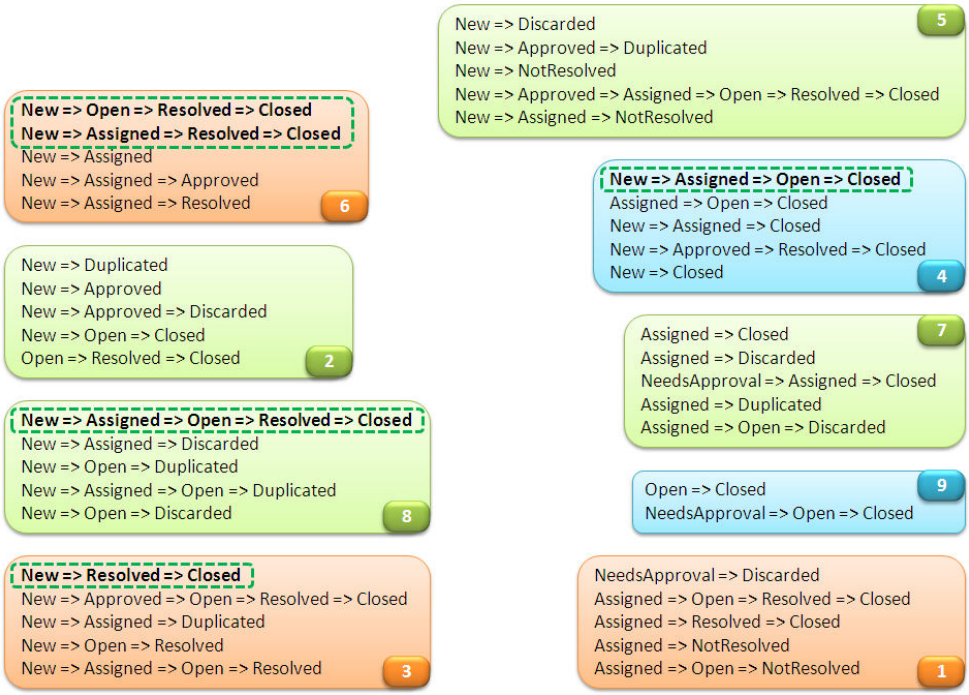

Figure 3. Top sequences for a model with 9 clusters

By setting the number of clusters to automatic, a surprising result emerged. Even though there was a lot of heterogeneous behaviour in the input data set, the algorithm produced just two clusters as shown in Figure 4. The behaviour in each cluster was rearranged to show events and transitions with higher support on top. Given that these are roughly the same events and transitions in both clusters, there is actually not much variation in the input set. In fact, the most frequent behaviour of cluster 1 in Figure 4 is similar to the behaviour of clusters 6 and 8 in Figure 3, while the most frequent behaviour of cluster 2 in Figure 4 resembles the behaviour of cluster 7 in Figure 3.

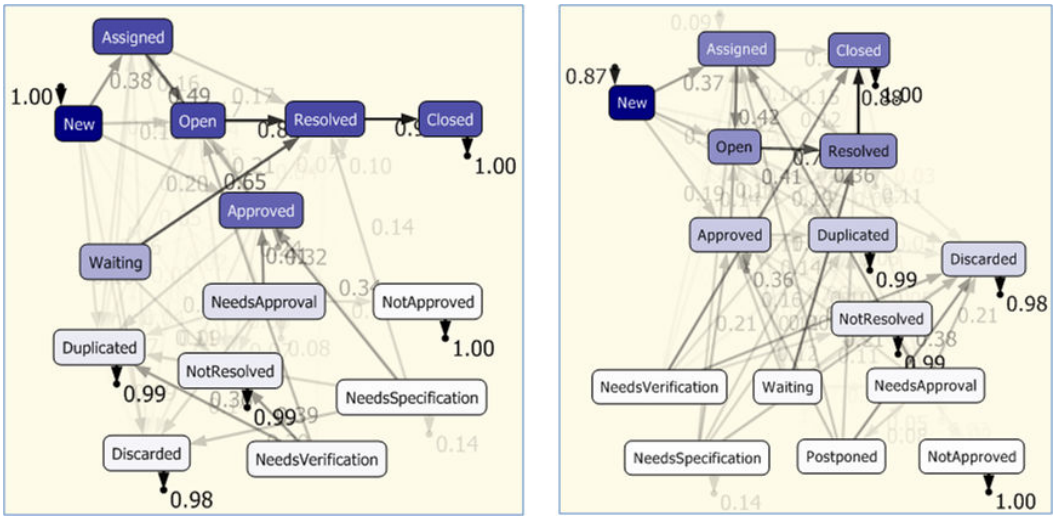

Figure 4. Sequential behaviour obtained for a model with 2 clusters

The fact that the algorithm ended up separating the input sequences in just two clusters (one cluster would not be clustering) is an indication that it is difficult to divide the input behaviour in several clearly distinguishable groups. And yet, the data set does contain very different sequences, as can be seen by simple manual inspection. These results suggest that the observed behaviour, despite being quite heterogeneous, is evenly distributed in such a way that it is difficult to identify clearly distinct patterns.

5.1 Global behavioural model

If behaviour cannot be separated into different clusters, then a single global model derived from the input data should suffice to identify typical behaviour. Figure 5 depicts such model. Rather than transition probabilities, state and transition counts are shown, providing an indication of the most frequent states as well as the most occurring transitions in absolute terms. To improve readability, node shading and line weight were made proportional to the state and transition counts, respectively. It is easy to see, for example, that "New" is the most recurring state, and that in most sequences the following state is "Assigned".

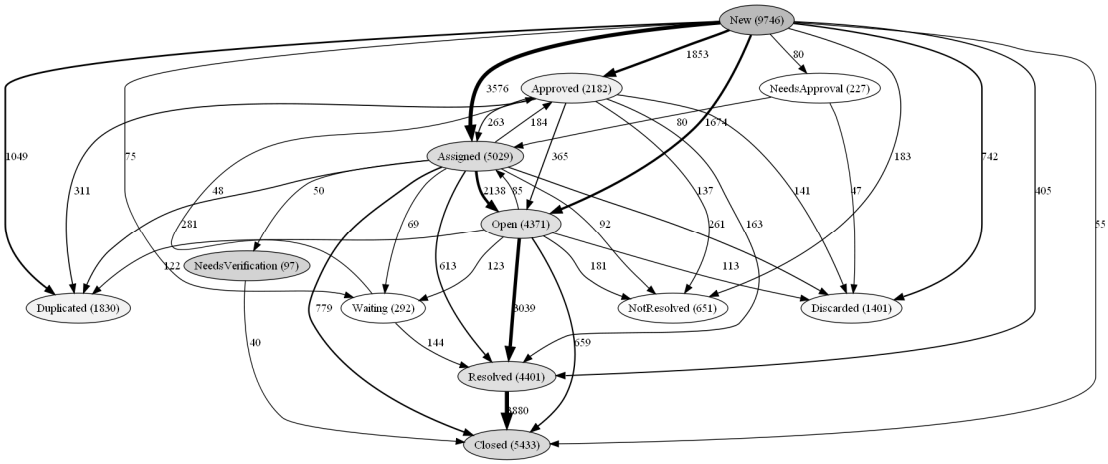

Figure 5. Global model for the preprocessed data set with absolute status and transition counts.

However, some care must be taken when drawing conclusions about the most common sequences, as subsequent transitions may refer to different issues. Figure 6 shows the actual most common sequences for the entire preprocessed data set.

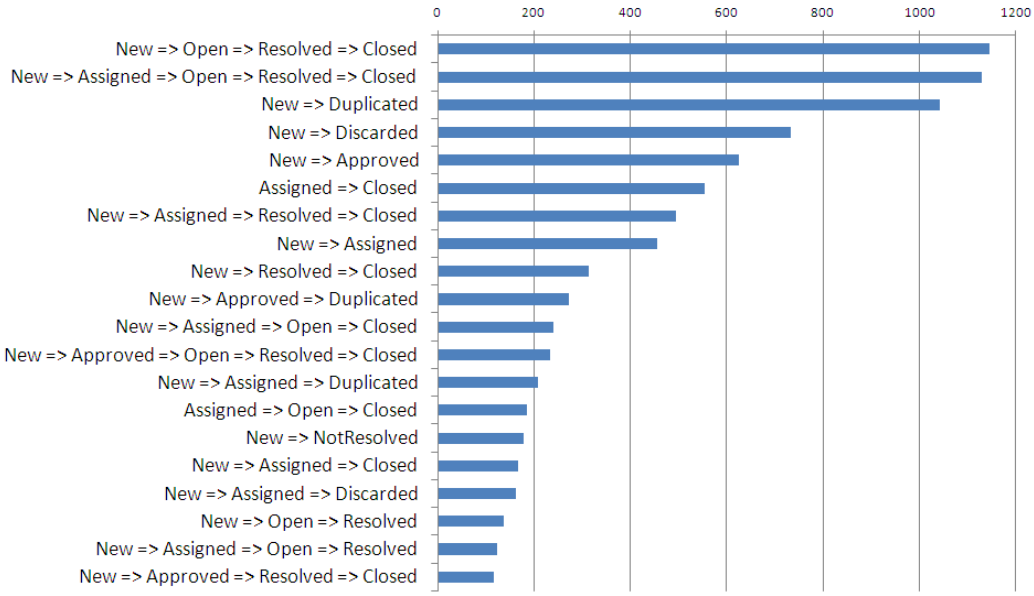

Figure 6. Top sequences in the preprocessed data set

6 Analysis

When the company was presented with a detailed account of the results above, they found that the conclusions agreed with what they expected. Basically, whatever the channel an issue comes from, it will be recorded in the system as “New”. Then someone will look at it and check whether it is a duplicate issue, whether it is relevant, what priority level it should be assigned, whether there is enough

information for the issue to be handled, whether there are other issues that could be related to this one, etc.

In some cases, the issue may end up being “Discarded” or being labelled as “Duplicated”. In most cases, it will follow the regular handling process. The issue may be “Assigned” either to a specific person, or collectively to the support team. The state will be changed to “Open” when someone is actively working on that issue. At this point, it will generally be a matter of time until the issue becomes “Resolved”. A few issues may end up in a “NotResolved” state but this result is, in general, not to be expected. Issues are automatically “Closed” when a new product release that includes its resolution is made available.

This description clearly resembles the ITIL Incident Management process described earlier: recording, classification, matching, diagnosis, resolution and closure are all present. In this case, classification and matching are being done in a single step between “New” and “Assigned”; diagnosis takes place when the issue is “Open”; resolution and closure are signalled by appropriate states as well. The difficulty is that the database contains much more behaviour than this description is able to account for. This is due to a number of reasons, including:

- Some states are no longer being used. For example, in the past it was common to make new issues go through an approval process, and some of that behaviour is still present in the database, as can be seen in Figure 5 in the transition from "New" to "Approved". Today, that approval is implicit when the issue changes from "New" to "Assigned".
- The support team members usually skip steps when the solution to the issue is obvious. For example, the team member who opens the issue may immediately recognise the problem and solve it, jumping directly to the "Open" state.
- The state transitions may appear to be reversed as the result of arbitrary loops. For example, an issue may be assigned to, and opened by, a team member, just to find that it should have been assigned to someone else; in this case, a transition from "Open" to "Assigned" will be recorded. The same behaviour can be observed in ITIL when there is escalation to a higher support level.

- The classification of an issue as a duplicate, or the decision to discard it, may come later in the process when more data is available about the issue.

These special but frequent cases explain most of the behaviour shown in Figure 5. The overall behaviour is definitely close to the Incident Management process. The analysis could now proceed, for example, by checking criteria such as those defined in (Brenner et al, 2002) or by measuring KPIs (Bartolini et al, 2006).

7 Conclusion

ITIL assessment is a laborious task that requires business processes to be monitored and reinterpreted in terms of ITIL guidelines. Process mining techniques can greatly simplify such analysis by being able to extract behaviour from large volumes of run-time data. In the case study, these techniques provided compelling reason to believe that the issue handling process indeed resembled the Incident Management process.

More importantly, the case study shows that process mining is a valuable tool for assessing this ITIL process and, as a result, potentially other ITIL processes as well. Such potential depends of course on the extent to which the existing process is supported by information systems that can provide useful data for analysis. In this study the analysis was conducted mainly on a behavioural perspective, but other perspectives – such as functional, informational, organizational, etc. (Schmidt, 2006) – could be considered as well, which provide a broad ground for future work.

References

van der Aalst, W., van Dongen, B., Herbst, J., Maruster, L., Schimm, G., Weijters, A.J.M.M. (2003) Workflow Mining: A Survey of Issues and Approaches. *Data and Knowledge Engineering*. 47(2). 237-267.

van der Aalst, W., Reijers, H., Song, M. (2005) Discovering Social Networks from Event Logs. *Computer Supported Cooperative Work*. 14(6). 549-593.

van der Aalst, W., Weijters, A. (2004) Process Mining: A Research Agenda. *Computers in Industry*. 53(3). 231-244.

van der Aalst, W., Weijters, T., Maruster, L. (2004) Workflow mining: discovering process models from event logs. *Transactions on Knowledge and Data Engineering*. 16(9). 1128-1142.

Agrawal, R., Gunopulos, D., Leymann, F. (1998) Mining Process Models from Workflow Logs. *6th Intl. Conf. on Extending Database Technology: Advances in Database Technology*. LNCS 1377. 469-483. Springer.

Bartolini, C., Salle, M., Trastour, D. (2006) IT Service Management driven by Business Objectives: An Application to Incident Management. *10th IEEE/IFIP Network Operations and Management Symposium*. 45-55.

van Bon, J., Pieper, M., van der Veen, A. (2005) *Foundations of IT Service Management Based on ITIL*. Van Haren Publishing.

Brenner, M. (2006) Classifying ITIL Processes: A Taxonomy under Tool Support Aspects. *First IEEE/IFIP Intl. Workshop on Business-Driven IT Management 2006* (BDIM'06). 19-28.

Brenner, M., Radisic, I., Schollmeyer, M. (2002) A Criteria Catalog Based Methodology for Analyzing Service Management Processes. *13th IFIP/IEEE Intl. Workshop on Distributed Systems: Operations and Management* (DSOM 2002). LNCS 2506. Springer.

Cadez, I., Heckerman, D., Meek, C., Smyth, P., White, S. (2003) Model-Based Clustering and Visualization of Navigation Patterns on a Web Site. *Data Mining and Knowledge Discovery*. 7(4). 399-424.

van Dongen, B., van der Aalst, W. (2004) Multi-Phase Process Mining: Building Instance Graphs. *Intl. Conf. on Conceptual Modeling*. LNCS 3288. 362-376. Springer.

Ferreira, D., Zacarias, M., Malheiros, M., Ferreira, P. (2007) Approaching Process Mining with Sequence Clustering: Experiments and Findings. *5th Intl. Conf. on Business Process Management* (BPM 2007). LNCS 4714. 360-374. Springer.

Girolami, M., Kabán, A. (2005) Sequential Activity Profiling: Latent Dirichlet Allocation of Markov Chains. *Data Mining and Knowledge Discovery*. 10(3). 175-196.

Goedertier, S., Martens, D., Baesens, B., Haesen, R., Vanthienen, J. (2008) Process Mining as First-Order Classification Learning on Logs with Negative Events. *3rd Workshop on Business Processes Intelligence* (BPI'07). LNCS 4928. Springer.

Greco, G., Guzzo, A., Pontieri, L. (2005) Mining Hierarchies of Models: From Abstract Views to Concrete Specifications. *3rd Intl. Conf. on Business Process Management*, BPM 2005. 32-47.

Herbst, J., Karagiannis, D. (1998) Integrating Machine Learning and Workflow Management to Support Acquisition and Adaptation of Workflow Models. *9th Intl. Workshop on Database and Expert Systems Applications*. 745-752.

Litten, K. (2005) Five Steps to Implementing ITIL. International Network Services. BT INS.

Medeiros, A., Weijters, A., van der Aalst, W. (2007) Genetic Process Mining: An Experimental Evaluation. *Data Mining and Knowledge Discovery*. 14(2). 245-304.

Mendel, T., Garbani, J.-P., Ostergaard, B., van Veen, N. (2004) Implementing ITIL: How To Get Started. Forrester Research

Porter, D. (2003) BASEL II: Heralding the Rise of Operational Risk. *Computer Fraud & Security*. 2003(7). 9-12.

Rozinat, A., van der Aalst, W. (2008) Conformance checking of processes based on monitoring real behavior. *Information Systems*. 33(1). 64-95.

Rozinat, A., Mans, R., van der Aalst, W. (2006) Mining CPN Models: Discovering Process Models with Data from Event Logs. *Seventh Workshop on the Practical Use of Coloured Petri Nets and CPN Tools* (CPN 2006). Aarhus, Denmark.

Schmidt, R. (2006) Flexibility in Service Processes. *CAISE'06 Workshop on Business Process Modelling, Development, and Support* (BPMDS'06). Luxemburg. June 5-9.

Tang, Z., MacLennan, J. (2005) *Data Mining with SQL Server 2005*. Wiley.

Zhang, I. (2007) Economic consequences of the Sarbanes–Oxley Act of 2002. *Journal of Accounting and Economics*. 44(1-2). 74-115.