Automate Does Not Always Mean Optimize: Case Study at a Logistics Company

Jan Suchy, Milan Suchy, Michal Rosik, and Agnes Valkova

Abstract

(a) **Situation faced**: Dynamic growth of digitized information creates space for the systematic collection of data related to business processes. Extraction of this data is an enormous challenge because of the existence of many systems, which store data in many formats. The logistics company examined here has fully automated its Purchase Order and Invoice Approval processes, driven by a BPM system. Logistics always deals with optimization and cost reduction, and the company asked us whether it was possible to optimize its processes further.

(b) **Action taken**: In our work, we focus on the extraction, pre-processing, and analysis of data that is stored in BPM systems. We presented the methodology with which to extract business-related events from processes of the logistics company, analyzed the BPM system, deployed processes to develop a connector for extracting event data, and used process mining techniques to reconstruct processes from event logs. Advanced analytics techniques make it possible to present collected data in an "as-is" view of processes and to find bottlenecks, loops, delays, and deadlocks.

(c) **Results achieved**: We identified the structure for stored data and the attributes attached to the metadata of the processes. Then we imported newly created process logs into a process mining tool. Next, we introduced a process model and its statistics based on the extracted processes. Finally, we pointed out characteristics and points for improvement in individual human activity. As a result, we identified bottlenecks, loops, suppliers'

J. Suchy • M. Suchy (✉) • M. Rosik • A. Valkova
GRADIENT ECM, Kosicka 56, 82108, Bratislava, Slovakia
e-mail: Jan.Suchy@gradientecm.com; Milan.Suchy@gradientecm.com; Michal.Rosik@gradientecm.com; Agnes.Valkova@gradientecm.com

J. vom Brocke, J. Mendling (eds.), *Business Process Management Cases*, Management for Professionals, DOI 10.1007/978-3-319-58307-5_25

characteristics, and found in the Purchase Order process over-allocated employees to dedicated tasks via the social network.

(d) **Lessons learned**: Today's businesses are process-driven; everything done in a business is a process. A process-driven application is a software that provides automatic execution of business processes and logs the executed activities. Most systems have design-time data that defines the processes and runtime data that includes information on executed activities. One can use connectors to extract the data in the desired process log structure. Process mining techniques allow us to reconstruct the process from logs, analyze it, and find optimization points. Processes can be analyzed from several perspectives: as human to human processes, human to system processes, and system to system processes.

1 Introduction

Most industries today automate their processes via workflow systems (van der Aalst and van Hee 2004). Efforts to capture and automate the desired behavior in industry processes can bring many benefits, but automation may also hide ineffective behaviors and instances. Therefore, when automating processes, companies must monitor them to see what takes place to enhance the processes so they meet changing business needs and eliminate risks.

Capturing processes' current conditions can be complicated and complex. What is needed is a team of people who will monitor all of the resource processes for each activity and record the individual steps that provide solutions to their problem areas. Thus, the team will be able to create a visible and interactive process model with dedicated timeframes for the people involved with each activity. The process model can be represented in a graph-based modeling language like Petri nets (van der Aalst 1998), BPMN (Wohed et al. 2006), or YAWL (van der Aalst and ter Hofstede 2005) so the resulting model holds all of the subjective views of the process analysts who contributed. On the other hand, process mining technology can reconstruct and visualize processes with an objective view in a fraction of the time. Reconstructing a process using process Mining technology requires acquiring all of the data recorded regarding all processes and transforming them into the structures needed for the reconstruction process to occur. Process mining makes it possible to reconstruct processes rapidly and to see the "as-is" reality of a process using a common and objective view. Process mining can reveal what actually goes on in an organization, providing a reality check that reveals the flaws and inefficiencies that must be worked out to enhance the firm's processes and the overall outcome.

Our goal was to show a precise picture of what really goes on in the Purchase Order and Invoice Approval processes. First, we became familiar with the specifications of the processes. Then we defined the structure in which we recorded the data. After familiarizing ourselves with the architecture of the BPM system, we developed a connector that extracts raw data and creates structured event logs. We then imported the event logs into a process-mining tool and introduced the process's basic statistics and characteristics. Finally, we focused on the key human activities, resources, and suppliers involved and gathered the knowledge and optimization points for both processes.

Our job was to provide to the logistics company tools for continuous business process improvement using three phases of the BPM Lifecycle (Dumas et al. 2013). Using the combination of the connector and the process-mining tool we developed, the company was able to perform the activities related to the phases of process discovery and process analysis, as well as the activities of the process monitoring and controlling phase. Using the connector makes exporting data in predefined intervals possible, and with the help of the process-mining tool, the company can analyze the variations in the process flow and the performance deviations in the set KPIs.

2 Situation Faced

Our logistics company is dealing with large numbers of invoices and purchase orders, covering their transportation business as well as overhead expenses. The number of invoices and orders was rising and the company was about to make several decisions related to hiring additional accountants, eliminating due dates, and loss of invoices. Scanning, automatic recognition, and data extraction and processing of invoices was the first solution the company needed. It was also important to implement a purchase order management system and document management system for storing, searching, and management of the documents. The key element of the solution that would be deployed was workflow automation for the Purchase Order and Invoice Approval processes based on the digitized version of the documents involved.

The benefits of implementing the solution were clear after short period of operation. The new invoice-approval process allowed the company to forego hiring the additional personnel and to process twice the number of invoices with the same personnel. The rate of lost invoices and invoices not approved on time fell to a negligible rate. The new purchase order approval process eliminated the need for double approval in most the cases.

Process automation introduced significant benefits, but it also raised the need for monitoring, controlling behavior, and searching for the additional optimization points. The company wanted to find out how to measure processes and how to measure the resources involved in the process execution. Its main interest was in controlling the fulfillment of enterprise-level KPIs and business rules, analyzing the purchase order process from the viewpoint of suppliers, and measuring discrepancies in the delivery of purchased goods. The increasing volume of rejected invoices was a concern. Data from active process instances were stored in a new system, but for analytical purposes a way to extract the historical data was needed as well.

2.1 Process Definition

To analyze and discover optimizations, we were provided with the Purchase Order process and the Invoice Approval process. Processes are implemented in a process-driven application and driven by a workflow engine. The Process Owner provided us with models and specifications of certain processes through which we became familiar with the individual steps and the process's attributes. The main objective in this part of work was to identify the flow of processes so we could validate the extracted process models.

2.2 Purchase Order Process

The Purchase Order process describes the creation and approval authority when orders are made. The system provides the users the ability to create a new order in the form of editable structured forms, so the user can fill any number of items ordered. Ordered items are defined by a set of attributes. After a new order is created, it is automatically launched into the Purchase Order process. The system then selects, based on the data entered, a tree of authorized users/group of authorities to approve the Purchase Order. Subsequently, the system assigns the approver/approvers tasks by email notifications. The approver may then approve orders or reject them. After the approval process, the system can decide, based on the financially authorized limit, whether the stock level is sufficient to approve the order and, if not, it initiates a reselection process from the existing tree of authoritative users/group of authorities. This process is repeated until all approvals have been acquired. If the order process is deemed approved in the system, the author will be notified of the outcome and the status will change to "Approved." If an order has been rejected, the system will change the status to "Declined," and a notification will be sent out to the appropriate author along with the reasons for refusal. If an order is rejected, the process and the order ends. With approved orders, the system identifies whether it is a cyclic order[1] and, if not, it continues and assigns tasks to those involved in the ordering process, who have been designated by the author of the order during its formation. In this part of the process, the user will have established and forwarded approved orders to its suppliers. The process then continues to confirm receipts of the ordered goods/services. The system then assigns tasks to authorized employees, who are given a chance to confirm a completed delivery order, confirm any partial deliveries, or cancel the order if the customer has cancelled the order. When a partial delivery takes place, the employee can wait for delivery of missing parts of the ordered goods or declare the order as partially delivered. Subsequently, the system marks the order as "Closed," and the process comes to an end.

[1]If the process is identified as a cyclic order, the system automatically declares it to have been delivered, and the process comes to an end. Cyclic orders are characterized by regular repetition.

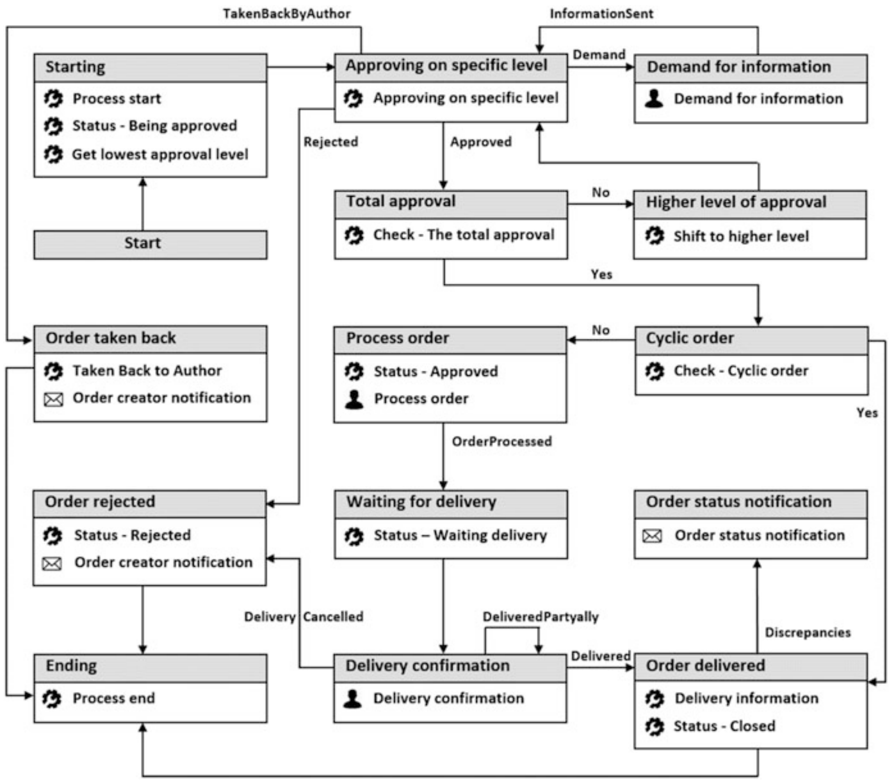

Fig. 1 Process diagram of the Purchase Order process

Figure 1 shows the process diagram for the Purchase Order process, which creates four human tasks and sixteen system tasks.

2.3 Invoice Approval Process

The Invoice Approval process is automatically prompted when an invoice is scanned/digitized. First, the system automatically assigns invoices to the correct order based on the order number. This solution was achieved by means of the combination of BPM and document-centric software. If the order is not found, the system will generate a task for manual entry of the order and assign it to a group known as "Accountants," whose goal is to find the order and pair it with its invoice. After assigning the order, the system automatically compares the total amounts, and if they do not match, the system checks the correctness of the cost center and continues to process the invoice in the approval process.

The Invoice Approval process has the same procedures as the Purchase Order process. The system generates tasks to confirm the accounted invoice. In this part of the process, the employee checks all other information pertaining to the invoice and, if the employee finds no discrepancies in the invoice, the system generates tasks to complete the accounting part/accounts payable process of the invoice, whereby the process will come to an end. However, if the employee finds irregularities, the process will continue, and the system will generate tasks to resolve them. At this point, the employee may reject the invoice and the process ends, or obtain missing data to enable the system to return the invoice to a re-approval state. Figure 2 shows the process diagram of the Invoice Approval process, which creates seven human tasks and thirteen system tasks.

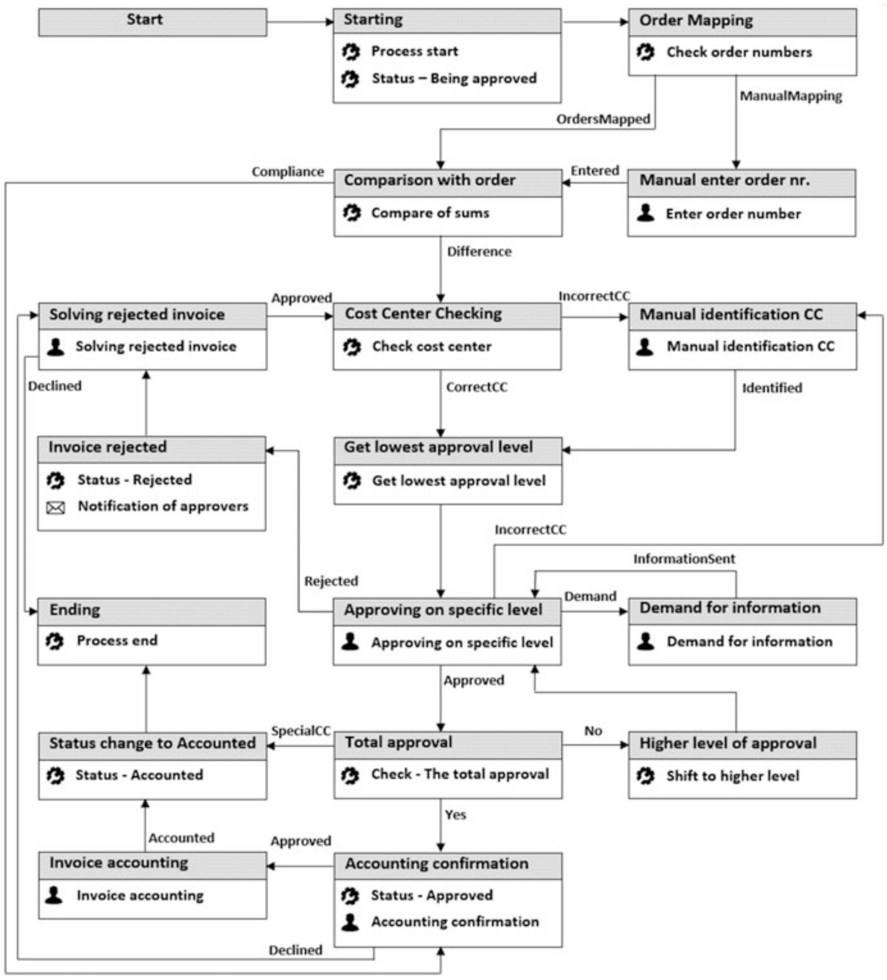

Fig. 2 Process diagram of the Invoice Approval process

3 Action Taken

In order for us to analyze the two processes and what takes place within them, we had to extract data from the company's databases and structure it. Data extraction is a challenge because data may be located in the database as well as in other formats (e.g., message logs, flat files, transaction logs, document management systems, ERP systems). Our main objective is to analyze the data obtained from a process-oriented perspective. In this section, we discuss information that should be present in such event logs.

Table 1 shows a fragment of an event log in which the information typically needed to analyze are presented. The main assumption is that the event log contains data related to a single process. Each event of the case is related to a single process instance, frequently marked as "case." Table 1 shows that Event ID 45678–45681 is related to Case 1. Another important factor is that every event must be related to an activity. As Table 1 shows, events refer to activities like Process Order, Being Approved, and Lowest level. In order for process analysis to take place, one must define the minimal requirements for the log: Case ID and Activity. If the log does not contain a timestamp, the correct chronological sequence must be secured at the first stage of events. Table 1 also indicates other information for each event, so we can see all events that have a timestamp. Without correctly ordered events we

Table 1 Fragment of the event log

Case ID	Event ID	Activity	Start timestamp	End timestamp	Event type	Resource
1	45678	Order delivered-	26.11.2014 12:51	26.11.2014 12:51	1	System
1	45679	Closed	26.11.2014 12:51	26.11.2014 12:51	1	System
1	45670	Waiting	23.10.2014 13:58	23.10.2014 13:58	1	System
1	45680	Approved	23.10.2014 13:25	23.10.2014 13:25	1	System
1	45681	Process order	23.10.2014 13:25	23.10.2014 13:58	2	USER2358
2	45682	Process start	23.10.2014 10:36	23.10.2014 10:36	1	System
2	45683	Being approved	23.10.2014 10:36	23.10.2014 10:36	1	System
2	45684	Lowest level	23.10.2014 10:36	23.10.2014 10:36	1	System
2	45685	Process end	25.11.2014 7:18	25.11.2014 7:18	1	System
2	45686	Approving	23.10.2014 10:36	23.10.2014 11:13	2	USER2358
2	45687	Approving	23.10.2014 11:13	23.10.2014 13:18	2	USER0357

would not be able to detect casual dependencies in process models. The number of timestamps recorded per event can be analyzed from a performance perspective, such as in terms of the duration between implemented events, where the activity itself has a duration value of zero; the duration of performed events, where the durations between events is zero; and the individual duration of that process instance, referred to as a throughput time. Other than the duration of events, we can further examine events between implementations, or waiting time. Table 1 also includes the resources attribute, which distinguish the personnel dedicated to specific activities. Attributes can be examined on two levels: an event-level attribute and a case-level attribute. A case-level attribute holds information regarding concrete process instances in which attributes' values are noted for all events corresponding to its case. Event-level attributes hold information that pertains to events within a case, so the values of these attributes are within a case within an event that may vary.

To be able to reason in regard to logs and to specify the requirements for event logs, we formalized several notions (van der Aalst 2011).

Definition 1 (Event, Attribute) Let E be the *event universe*, that is, the set of all possible event identifiers. Events may be characterized by *various attributes*, such as an event that has a timestamp, corresponds to an activity, is executed by a particular person, or has associated costs. Let AN be a set of attribute names. For any event $e \in E$ and name $n \in AN$, $\#_n(e)$ is the value of attribute n for event e. If event e does not have an attribute named n, then $\#_n(e) = \perp$ (null value).

Definition 2 (Case, Case Attribute, Trace, Event log) Let C be the *case universe*, that is, the set of all possible case identifiers. Cases, like events, have attributes. For any case $c \in C$ and name $n \in AN$, $\#_{\mathrm{n}}(c)$ is the value of attribute n for case c ($\#_n(c) = \perp$ if case c has no attribute named n). Each case has a special mandatory attribute *trace*: $\#_{trace}(c) \in E^*$. $\underline{c} = \#_{trace}(c)$ is a shorthand for the trace of a case. We assume $\#_{\mathrm{trace}}(c) \neq \langle\rangle$, that is, traces in a log contain at least one event.

A *trace* is a finite sequence of events $\sigma \in E^*$ such that each event appears only once; that is, for $1 \leq i < j \leq |\sigma|$: $\sigma(i) \neq \sigma(j)$.

An *event log* is a set of cases $L \subseteq C$ such that each event appears at most once in the entire log; that is, for any $c_1, c_2 \in L$ such that $c_1 \neq c_2$: $\partial_{\mathrm{set}}(\underline{c}_1) \cap \partial_{\mathrm{set}}(\underline{c}_2) = \varnothing$.

If an event log contains timestamps, then the ordering in a trace should respect these timestamps; that is, for any $c \in L$, i and j such that $1 \leq i < j \leq |\underline{c}|$: $\#_{\mathrm{time}}(\underline{c}(i)) \leq \#_{\mathrm{time}}(c(j))$. Events and cases are represented using *unique* identifiers. An identifier $e \in E$ refers to an event, and an identifier $c \in C$ refers to a case. This mechanism allows us to point to a specific event or a specific case, as there may be many events with identical attributes; for example, the start events of activity a may have been recorded for other cases, and there may even be multiples of such events within a case. Similarly, there may be several cases that followed the same path in the process. These identifiers are just a technicality that helps us to point to particular events and cases, so they do not need to be present in the original data source but may be generated when we extract the data from the various data sources.

Extracted data from the data sources must be saved in a suitable format. The standard format for storing and exchanging event logs is Mining eXtensible Markup Language (MXML). Using MXML, one can store event logs like the one shown in Table 1 using an XML-based syntax.

Another format is eXtensible Event Stream (XES) (Günther 2009), which is the successor to MXML. The XES format has been made less restrictive and extendible based on many practical experiences with MXML.

The most widely used format used is comma separated values (.CSV). This format is less restrictive than XES, but it only enables one to store data, not to create one's own extensions.

We met many challenges when extracting the event logs (van der Aalst 2011). One of the challenges is known as *correlation* events, which are events that need to be related to each other. Dealing with legacy and a variety of interconnected systems requires additional effort to correlate events; see (Ferreira and Gillblad 2009) for an example of an approach with which to correlate events with no a priori information. Events must be ordered per case, which does not require timestamps in principle, but when merging data from different sources, one must typically depend on *timestamps* to sort events in order of occurrence. In extracted data, we can come across existing cases, which are still running, so event logs typically just provide a *snapshot* of a longer running process. Another challenge is the *scoping* of the event log. Information systems may have thousands of tables, so one must know which tables incorporate the relevant data and know how to locate the required data and scope it. The *granularity* of logged events is also an issue in the system, as some systems produce low-level events. There are several approaches to preprocessing these types of events; for instance, low-level patterns that appear frequently can be abstracted and merged into a new event that represents the performed activity (Bose and van der Aalst 2009).

3.1 Event Log Extraction

We designed and implemented a connector that extracts event logs from a process-driven application that ensures a proper process-monitoring functionality. All data is present in relational databases. We used design-time parts of the database to define processes, activities, events, cases, and case-level attribute data. Activity, events, and their metadata distinguish the monitored processes, and we extracted data from runtime parts in the database.

Design-Time Data The design-time data contains all process definitions. For the individual processes defined, information is available regarding the date of implementation, actual running versions, and the historical record of previously implemented versions. In addition, all process have defined their activity sessions/lines and their metadata. For individual activity, corresponding events are defined as one of two types: human or system. Information about the users who performed the human activities in the process was also available. The important information was that which involved the setup process concerning the amount of detail that will be logged, where the most necessary values represent complete logging. Throughout the process of logging all activities, attribute values pertaining to process instances regarding individual process activities were important for analysis.

Runtime Data The runtime data contains all the data for the running instances of processes and their components. Each released process instance contains unique identifiers. Unique identifiers and the foreign key for a process instance in which it took place are recorded for activities carried out in the process. Individual events are recorded for activities that contain unique identifiers that themselves contain a foreign key to an associated activity instance (which is specific to when an event instance occurs). For individual events, timestamps are recorded and the human activity type holds the employee that performed the activity. Metadata related to process, activity, and event instances are linked via specific foreign keys.

Figure 3 demonstrates a scheme for the processes extracted in an event log. Using the information about processes gained from the design-time areas of the database, we extracted data from the runtime parts of the database. Table 2 contains a list of attributes needed to reconstruct individual processes. The attributes "Resource" and "Event Type" are expandable attributes for reconstructed social networks and in respect to the server or human activity.

Data extracted from processes and supplementary information regarding suppliers were supplier name, supplier city, and supplier state as case attributes.

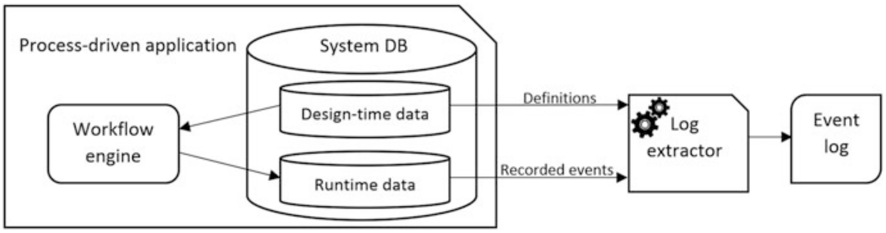

Fig. 3 Scheme of event log extraction

Table 2 Base extracted attributes

Attribute name	Orders	Invoices	Description
Case ID	✓	✓	Process instance identifier
Activity	✓	✓	Activity + event name
Start timestamp	✓	✓	Event start time
End timestamp	✓	✓	Event end time
Event Type	✓	✓	1 system, 2 human
Resource	✓	✓	Resource name

For the Invoice Approval process, user comments data was extracted to identify the most frequent reason for refusal of invoices. An additional case attribute was the case status, which helped to reveal the differences between completed, running, error, and deleted process instances. Analyzed data is provided only in regard to actual versions of processes.

Logs were stored in .CSV files. If we were to rate the quality of logs based on maturity levels (IEEE Task Force on Process Mining 2011), their rating would be five stars because logs are derived from a BPM system. Events are recorded in an automatic, systematic, reliable, and safe manner. Given such recording, the reconstruction of the processes did not require pre-processing of the data. The connector we developed enabled us to export the process instances according to the design-time information, which were completed and executed in the latest versions of both processes.

3.2 Process-Mining Techniques

We used a process-mining technique for reconstructed processes, as this technique can extract information from event logs. The goal of process mining is to discover, monitor, and improve processes. Process mining includes discovery process models, conformance checking (comparing model and log), social networking, organizational structure mining, case prediction, and history-based recommendations.

A number of tools for process mining is available for commercial and academic use. Commercial tools for process mining offer simple visualizations for end users and are significantly faster than other tools are in processing Big Data. Academic tools offer more algorithms, which may be difficult for less skilled users to apply. However, academic tools may have a wider range of use, and in the process of reconstruction they can expand support for concurrency (van der Aalst 2016). Because of the possibility of a large volume of data, we chose to use the commercial tool *Minit*[2] for the process analysis, as we are familiar with its functionality and it offers the most modern process-discovery algorithm, which is similar to fuzzy mining (Günther and van der Aalst 2007). With *Minit* we can import datasets for a wide range of possibilities in analyzing process models in a logistic company and their statistical characteristics. It was importance in our analysis to analyze the social network in order to bring out the fine details of each zoned resource in all processes.

After we imported the event log into *Minit*, identified process maps were created. Figure 4 shows a Purchase Order process map, and Fig. 5 shows an Invoice Approving process map. Both of the process maps[3] portrayed contain the case count metric.

[2]www.minitlabs.com

[3]Both process maps were redrawn for better readability after they were printed on paper.

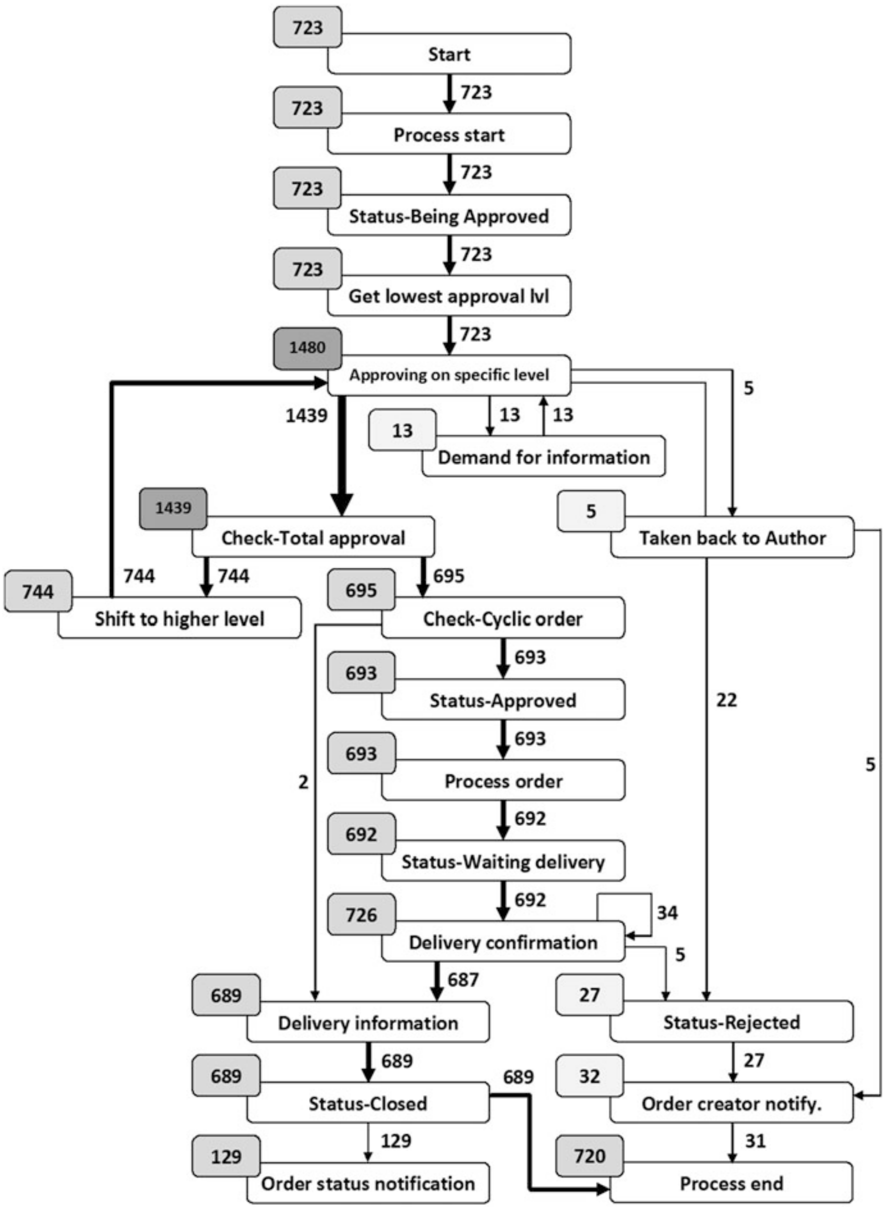

Fig. 4 Purchase Order process map

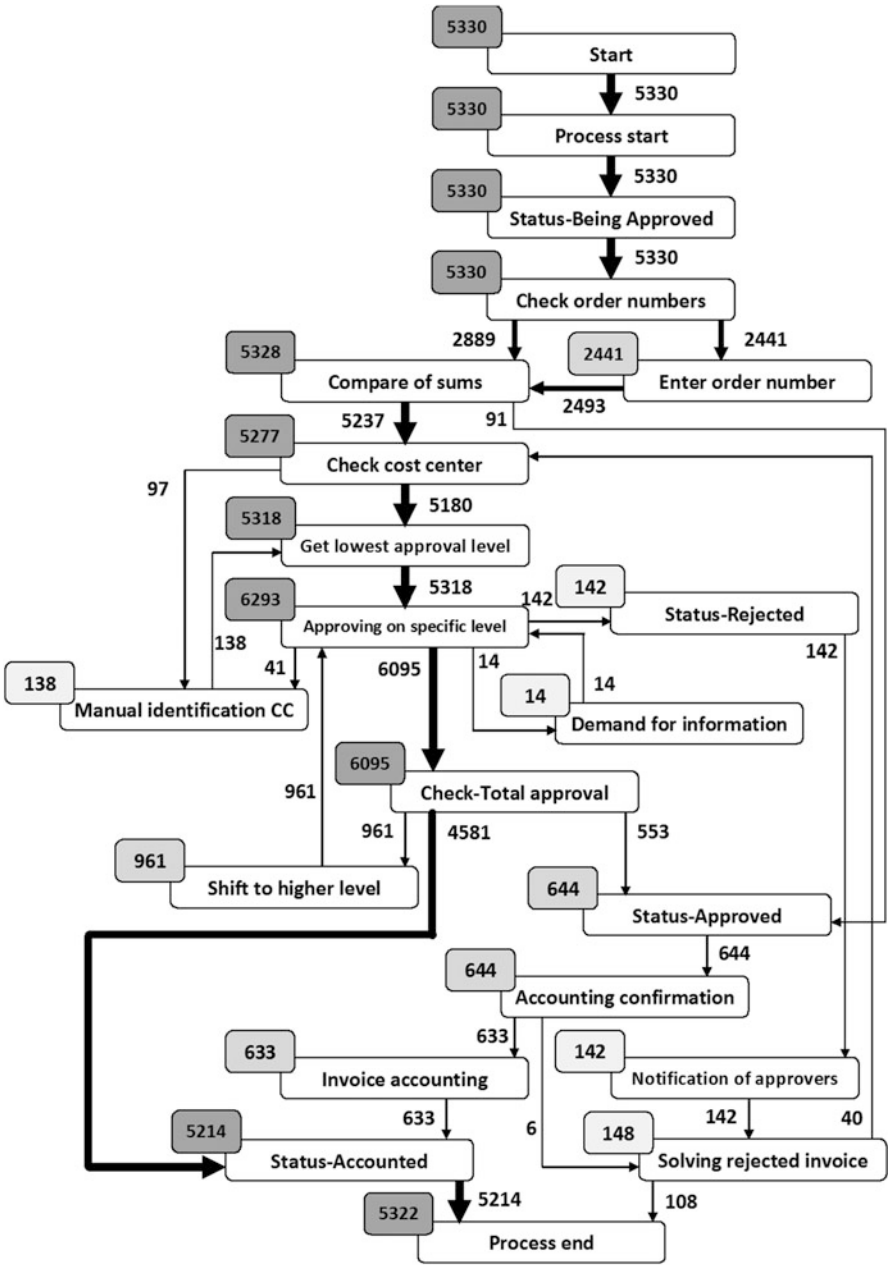

Fig. 5 Approve Invoice process map

4 Results Achieved

Our overall goal was to reconstruct and analyze process models from an event log. First, we identified the main process flows and their deviations. Then we analyzed social networks and selected activity processes. Statistical findings and analysis were performed only for completed process instances. Table 3 contains the basic statistics of the reconstructed processes.

4.1 Control Flow Identification

Processes' main streams were identified as their most numerous variants. A process variant is defined as the presence of unique activity sequences.

Purchase Order In the order-approval process, five of the most numerous variants accounted for 88% of the overall behavior. Table 4 indicates the basic characteristics of chosen variants. *Variant 1* describes when an order is approved on the second level, so "Approving on a specific level" occurs twice. Then orders are approved without additional activity; that is, an order is approved, processed, and its goods are delivered. *Variant 2* describes when a purchase order is approved without the approval of a higher-ranked individual, so they are approved once,

Table 3 Overview of processes characteristics

	Orders	Invoices
Timeframe	166 days 7 h	48 days 3 h
Cases	720	5322
Events	12,328	65,985
Activities	20	20
Event attributes	3	3
Case attributes	17	20
Variants	28	54
Resources	42	45

Table 4 Characteristics of the most numerous variants in the approval process of Purchase Orders

Variant ID	Cases coverage	Events per case	Events coverage	Mean duration
Variant 1	36%	17	35%	10d 23 h 15 min
Variant 2	20%	14	16%	5d 21 h 51 min
Variant 3	18%	20	21%	12d 21 h 34 min
Variant 4	11%	18	12%	8d 5 h 55 min
Variant 5	3%	21	4%	7d 9 h 2 min

Table includes how many cases and events are included in the monitored variants, along with information about their mean duration

Table 5 Characteristics of the most numerous variants of the Invoice approval process

Variant ID	Cases coverage	Events per case	Events coverage	Mean duration
Variant 1	50%	11	44%	1d 3 h 54 min
Variant 2	28%	12	27%	1d 11 h 46 min
Variant 3	8%	15	9%	2d 22 h 48 min
Variant 4	4%	18	6%	8d 10 h 45 min

This table indicates how many cases and events were obtained in the monitoring of variants, along with mean duration values

without any additional activities. The approval of the Purchase Order thereafter is completed upon delivery of all goods. *Variant 3* describes when a Purchase Order is approved in the third level, so "Approving on specific level" takes place three times. Purchase Orders are approved without carrying out any further activities, and the approval of the Purchase Order thereafter is completed upon delivery of all goods. *Variant* 3 and *Variant* 4 describe behavior when Purchase Orders are approved in the second (*Variant 4*) and third (*Variant 5*) level. These Purchase Orders are approved and processing takes place, but goods delivered are carried out with discrepancies.

Invoice Approving Four of the most numerous variants accounted for 89% of the overall behavior in the Invoice Approval process. Table 5 indicates the basic characteristics of the selected variants. *Variant 1* describes when an invoice is successfully mapped to compare the difference in prices found, followed by the cost center's verifying the data. The invoice is approved at the first level without the need for a higher authority, and no further activities take place at this stage. All invoices in this variant are directed to a special Cost Center (CC). *Variant 2* describes when the invoice does not have a purchase order labeled and one that is manually labeled must be secured to complete the invoice's missing data. Price comparison reveals a difference. The CC's checking takes place, and the center that will take further action to process the order is correctly defined. At the first stage, the invoice does not need the approval of a higher, so there is no further activity. All invoices in this variant are directed to a special CC. *Variant 3* describes when a purchase order number pertaining to the invoice is not labelled, so a manually labelled order must be secured. Price comparisons reveal a difference, and an inspection is followed by the CC's checking and defining which center will process further. The invoice at the second level is approved without the need of a higher authority or further activity. All invoices in this variant are redirected to a special division of the CC. *Variant 4* describes when an invoice that has an unlabeled order number requires that a manually labeled order number be secured. The prices comparison reveals a difference. The CC is correctly selected. Invoices are approved at the second level. Invoices are approved without further activity, and no invoices in this variant are directed to the CC but they are approved and sent to accounts payable.

The most frequent behavior and performance properties were revealed. Both processes have a common bottleneck, where multi-level approvals take place. Both processes had a remarkable growth in throughput time when multiple approvals took place. In the Purchase Order process, throughput times in the second-level approvals were 1.8 times higher than an approval on the first level, and they were 2.2 times higher in the third level. In the Invoice Approval process, throughput times on the second level were 2.5 times higher than an approval on the first level and 4.5 times higher on the third level.

4.2 Points of Interest in the Purchase Order Process

In this section, we tend to the possibilities of optimizing the Purchase Order process. We identify areas in which we see processes that hold statistical value or performance problems, focusing on the approval of purchase orders, the people involved in the process, and their social network.

4.2.1 Approval Level

Activity seen in this section pertains to the approval of purchase orders. All orders must be approved when the approver can do so the first time (or request additional information). A problem was discovered in the distribution of work, as the approver with the highest number of approvals approved 288 orders in an average of 20 h, while the approver with the second highest number of approvals approved 208 orders in an average of eight minutes.

For seventeen process instances, research found procedural irregularities, where one user approved on multiple levels, although the rules state that the approver cannot approve at multiple levels.

In one instance, 23 orders were approved for one supplier with the same amounts in each other and three users carrying out the same order of processing. Seeing this procedure take place allowed a cyclic procedure to be implemented to save time in process.

4.2.2 Resources in the Process

Over-allocated resources were discovered in the Purchase Order process. Resources in the process provide multiple tasks, where they carry out all or some of the required human activity. Unclearly defined working tasks for employees show strained and overworked employees comparing one another. Within the process, 42 resources were seen, but five employees performed 55% of all human activities and 57% of all process instances. Process owners were advised to relieve these over-allocated resources. The research also revealed that some employees had to communicate with a larger number/group of employees to process purchase orders and that a pair of resources shifted their work among other areas of work in 27% and 20% of process instances, respectively. Over-allocated resources are clearly shown in the social network of the Purchase Order process (Fig. 6). Their communication is also compared to that of others.

When purchase orders are created for individual resources, there is often insufficient knowledge regarding the work process. The slowest employee performed an

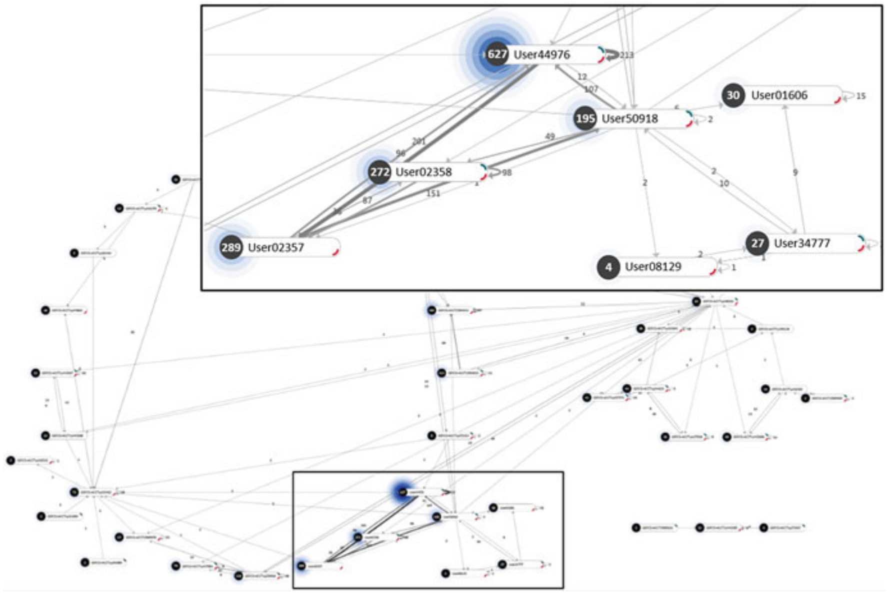

Fig. 6 Social network for the Purchase Order process with event-count metric

activity 30 times slower than the fastest employee. In addition, the duration of activities ranged widely, with the longest difference being 17 s for the shortest duration compared to 21 days and 18 h for the longest. Over-allocated resources were more likely to take longer than average to complete an activity.

4.2.3 Suppliers' Characteristics

Through the "Delivery confirmation" activity, we identified discrepancies in the delivery of goods/services for individual suppliers and information about the number of discrepancies that took place. Contractors were also identified who reported discrepancies in all of their deliveries of goods/services. From this information we were able to predict the time it would take to deliver goods/services for individual contractors together and the likelihood that the order would be completed. This information helps process owners accurately plan and schedule orders for the delivery of their goods/services.

4.3 Points of Interest in the Invoice Approval Process

In this section, we tend to the Invoice Approving process. We investigated activity that took place in "Manual enter order number" and saw that it accounted for approximately 50% of all process instances. Focusing on the "Solving rejected invoices" activity, we applied text-mining techniques to see the most common causes of rejected invoices.

4.3.1 Manual Entry of Order Numbers

The system identifies invoices with incorrect or missing IDs. For unidentified invoices, the system allocates tasks among the staff to enter correct order numbers manually. We identified the suppliers that needed the most such manual entries. By introducing these suppliers to the process owners, we reduced the number of invoices that had to be manually labeled, saving up to 7 h in invoice-processing time.

4.3.2 Resolving Rejected Invoices

Invoices can be rejected at the first level of approval or at the payment-processing stage. Rejected invoices are directed to a human activity that examines why the invoice was rejected and determines whether to end the activity (Decline the invoice) or to correct the problem and begin the process again. The process owner allocated three employees to complete the tasks involved in this activity, planning to divide tasks equally among the three. However, the reality was that the three employees differed in terms of how often they were assigned a task, as one performed 56% of all tasks, another performed 34%, and the third performed only 10%. The person with the highest number of tasks in this activity obtained the shortest average duration time in resolving rejected invoices, taking only 13 h, while the employee with the lowest percentage of assignments took an average of 15 days. These “reality checks” were presented to the process owner so the work habits of the most efficient employee could be presented to the other employees to improve their effectiveness.

Through analysis of this information, we found that, if an employee devoted less than 2 days to solving a rejected invoice, the invoice was declined 85% of the time and only 15% were approved. A ratio of 50% approved and 50% declined were obtained if an employee was solving a rejected invoice more than 2 days.

In order to establish the most common grounds for refusal regarding invoices, an expansion of the event log was made to contain “User comments.” Data was extracted and reserved for this activity alone using a frequency analysis of phrases in the users comments to obtain the most common causes of rejection. The most frequent phrase was “wrong verification,” which appeared in 29% of the rejections. The second most frequent phrase was “wrong amount,” which appeared in 24% of the rejections. Less frequent phrases included “missing date” and “invoice in the wrong language.” With these phrases identified, the process owner was able to address these frequent shortcomings and help the overall process to run more smoothly.

5 Lessons Learned

For analyses of processes to have value, the logs must be of high quality, so a significant challenge lies in pre-processing the data from multiple systems to create high-quality logs. All events and information that were recorded in the midst of all operations in processes had to be included, and the higher the quality of the information in the logs, the higher the quality of the resulting details about the processes.

We focused on the extraction, pre-processing, and analysis of data that was stored in a process-driven application. This software automatically executes business processes and logs the executed activities. In most of the systems, one can find design-time data, that is, process definitions and runtime data, including information on activities executed during the process. We defined the structure for stored data and defined the attributes attached to analyzing the processes.

Data was extracted using a connector we developed to identify design-time data automatically and extract runtime data into the desired process log structure, which must at least include an instance identifier and process steps in the form of an activity identifier to enable frequency analysis. If the process log does not include a timestamp attribute, events in the log must be ordered chronologically before the log is imported into a process-mining tool. Performance analysis can show the time between executions of two consecutive activities if at least one timestamp attribute is logged. However, when two timestamp values are logged—one describing the start and one describing the end of an event—we can reconstruct the process model, including the active time and waiting time for activities and the paths between them. One of the most important features of analytical tools is their ability to aggregate data, because sometimes the mean value does not have the same informative value as the median value. The process logs of two processes were exported using a comma-separated values format. Thus, we can gain insights using process-mining techniques for analyses and evaluations.

Process mining, along with business process modeling and analysis, provides an important bridge between computational intelligence and data mining. The idea of process mining is to discover, monitor, and improve real processes by extracting knowledge from event logs stored in information systems. One of the major benefits of using the technology is improved process transparency. By reconstructing the process models, we discovered the main process flow, as well as its deviations. Identified delays and bottlenecks become visible for both processes. Social networks were also reconstructed, and over-allocated resources were identified.

As process-mining techniques also include data mining, they can help to identify how input parameters influence process flow. Historical data can be used to predict the duration of process instances. We also focused on attributes that describe invoice rejections, where text-mining techniques helped to uncovered the most frequent reasons for rejecting an invoice.

Process discovery via process-mining methodology is faster and more cost effective than process reconstruction using a team of consultants. The goal should be use process mining and its advantages daily in order to conduct process-mining based on real-time event data. Using the real-time data makes it possible to enrich process model in several ways. For example, users involved in a process can navigate through the process in non-standard situations, and real-time data can be projected on process map to show a process's status, thereby informing employees about delays and other problems. Process mining should be viewed as a continuous process that provides actionable information.

One of the possibilities that was not pursued in this work is how to analyze processes within a BPM system. Our study analyzed primarily human activity, but system activities can also be analyzed to detect performance characteristics, communication, and bottlenecks that pertain to individual systems in a process. By modifying logs or using selected filters in the process-mining tool, we can select and analyze parts of processes that most needed attention.

References

Bose, R. P. J. C., & van der Aalst, W. M. P. (2009). Abstractions in process mining: A taxonomy of patterns. In U. Dayal, J. Eder, J. Koehler, & H. Reijers (Eds.), *Business Process Management (BPM 2009)*, *volume 5701 of Lecture notes in Computer Science* (pp. 159–175). Berlin: Springer.

Dumas, M., La Rosa, M., Mendling, J., & Reijers, H. H. (2013). *Fundamentals of business process management*. Berlin: Springer.

Ferreira, D. R., & Gillblad, D. (2009). Discovering process models from unlabelled event logs. In U. Dayal, J. Eder, J. Koehler, & H. Reijers (Eds.), *Business Process Management (BPM 2009)*, *volume 5701 of Lecture notes in Computer Science* (pp. 143–158). Berlin: Springer.

Günther, C. W. (2009). *XES Standard definition*. www.xes-standard.org

Günther, C. W., & van der Aalst, W. M. P. (2007). Fuzzy mining: Adaptive process simplification based on multi-perspective metrics. In G. Alonso, P. Dadam, & M. Rosemann (Eds.), *International conference on Business Process Management (BPM 2007)*, *volume 4714 of Lecture notes in Computer Science* (pp. 328–343). Berlin: Springer.

IEEE Task Force on Process Mining. (2011). Process mining manifesto. In *BPM Workshops*, volume 99 of Lecture notes in Business information processing. Berlin: Springer.

van der Aalst, W. M. P. (1998). The application of petri nets to workflow management. *Journal of Circuits Systems and Computers, 8*, 21–66.

van der Aalst, W. M. P. (2011). *Process mining: Discovery, conformance and enhancement of business processes*. Berlin: Springer.

van der Aalst, W. M. P. (2016). *Process mining: Data science in action*. Berlin: Springer.

van der Aalst, W. M. P., & ter Hofstede, A. H. M. (2005). YAWL: Yet another workflow language. *Information Systems, 30*(4), 245–275.

van der Aalst, W. M. P., & van Hee, K. (2004). *Workflow management: Models, methods, and systems*. Cambridge, MA: MIT Press.

Wohed, P., et al. (2006). On the suitability of BPMN for business process modelling. In S. Dustdar, J. L. Fiadeiro, & A. P. Sheth (Eds.), *Business process management*. Berlin: Springer.

Jan Suchy is process and data analytic, while he is experienced in the field of development and implementation of the analytic algorithms. He focuses on the analysis of the data obtained from the enterprise systems, such as LOB, CRM, ERP and various BPM tools. Identification of “business values” of data, their extraction, transformation and processing by the means of Process Mining and Data Mining technologies are essentially covered by the analysis performed by Jan. Outcomes of his analysis form basis for optimisation, control and effective management of the processes. His experience consists of various analytical projects in different business sectors (Insurance, Banking, Public sector, Gaming industry, IT, Media, etc.). Jan contributes to academic publications and actively participates on the professional conferences in BPM sector. His added value lies in the prompt adaptation of the most modern analytical methods and their implementation in practice.

Milan Suchy is data analytic and expert in the field of process analysis. He has rich experience with extraction, processing and visualisation of data. In the past he participated on projects focused on reconstruction and optimisation of business processes, creation of machine learning models and identification of patterns in data. His strong areas include ability to apply data science and process science methods to various business sectors and provide to the clients valuable information hidden in data.

Michal Rosik As Product Visionary for Minit, he defines the Research & Development direction for this process mining solution, develops close ties to the academic community in this area and evangelises process mining benefits to enterprises worldwide. He received his master's degree in Management of Information Systems and Banking and Finance at Comenius University in Bratislava, Slovakia. Michal previously lead Microsoft Consulting department in Siemens and was involved in several large enterprise projects as a consultant and project manager. In his free time, he is a passionate trail runner.

Agnes Valkova is the head of marketing at Resco.net. With over thirteen years of experience in the Sales/Marketing industry and 10 years of direct experience in the field. She is an award winning sales/marketing executive with extensive global online, digital, and entertainment expertise. Proven success in developing cross platform content and marketing strategies, which exceed industry standards and deliver significant ROIs. A multi-disciplinary with a unique combination of creative, commercial, lingual and technical acumen. Skilled at high-level strategy, as well as tactical implementation. Agnes is also a public speaker at many worldwide conferences pertaining to her field in IT—Microsoft Envision, European SharePoint Conference, and more.