The Methodology of Extraction and Analysis of Event Log Social Graph

Alexandra Soboleva
Peter the Great St. Petersburg Polytechnic University
St. Petersburg, Russia
sania.soboleva@mail.ru

Olga Tushkanova
Peter the Great St. Petersburg Polytechnic University
St. Petersburg Institute for Informatics and
Automatization of Russian Academy of Sciences
St. Petersburg, Russia
tushkanova.on@gmail.com

Abstract—**Process mining allows extracting information on the company's business processes from the event logs of its information systems, including the social resource graph. Such a social graph represents the communication of subjects (human resources) of various business process stages. The paper is focused on extracting a social graph from the ticket history of the customer support system, which stores the same actions of resources that have different business process functions. The paper presents the methodology and, as its realization, the package for extracting a social graph from the ticket history and analyzing the nature and quality of the communication between resources during the ticket solution. The package implements the functions of filtering the ticket history, extracting a social graph from it, social community detection in the extracted graph based on the structural similarity of the nodes and social graph leader identification. Two scenarios of applying the developed methodology are presented using the example of the ticket history of a large retailer.**

I. Introduction

Today, most companies have information systems that collect, store, process, and analyze data on their business processes. The architecture of such information systems, in most cases, is rather complicated due to their rich functionality and adaptation to each business process. This kind of information system requires technical support. The technical support, in this case, is a separate business process and often is performed by a whole department or a different company. As a rule, it usually needs to be optimized. Companies use customer support systems to simplify the management of the technical support business process. The support systems are information systems with a web interface that allows users of a service or software product to create tickets in case of any problems with the usage of the service or software product. Today, there are many different platforms for customer support systems, for example, Zendesk [1], Okdesk [2], Freshdesk [3], Jira Service Desk [4], Kayako [5], and others. These systems usually store the client ticket history in the form available for export as an event log for further analysis.

Event log analysis is usually performed in the scope of the Process Mining. The main goal of the analysis is to discover, monitor, and improve the real processes of the company using the information from the event logs of the company's information systems [6]. At the discovery stage, a process map is extracted from the event log. The process map shows the real traces of the business process object within the information system. The object of a business process is a product whose behavior management is the goal of modelling and modifying a business process. For example, a "manufactured product" is the object for the information system in production, and an "order" is the object for the online store information system. At the conformance checking stage, the business process map is compared with a reference model, which, as a rule, is created manually by an expert during the information system design stage. As a result of the conformance checking, new knowledge about the process allows us to adjust the reference model and obtain a new business process model that is more optimal according to key performance indicators (KPI). At the enhancement stage, the new business process model is expanded with identified bottlenecks, decision points, social networks, organization mining, model repairing, case predictions, and history-based recommendations. It allows us to investigate the business process in detail, therefore optimize it more efficiently and provide operational support of the business process.

Human resources play a crucial role in the business process of technical customer support because they find out the root causes and eliminate them. Therefore, the objects of analysis are, first of all, the employees of technical support teams and their interactions.

It is convenient to represent communication between people or groups of people as a social graph. The social graph is a theoretical construct designed to investigate the relationships between individuals, groups, organizations [7]. In a social graph, an object of research, for example, a resource, is a node, and a relationship between objects is an edge. Algorithms that allow construction and analysis of the social graphs relate to social networks analysis methods and graph theory.

The extraction of the social resource graph from the event log refers to the enhancement stage. It is an additional representation within the business process model. Analysis of the social graph allows identifying methods to optimize the ways and options for communication between resources and balance their workload.

The paper presents the methodology of extraction and analysis of event log social graph and, the realization of this methodology, PM_SN package (corresponds to Process Mining and Social Networks). The package implements two main functions: the first extracts a social resource graph from ticket histories of a customer support system, the second analyzes nature and quality of communication of various team members during customer ticket solving.

ISSN 2305-7254

The methodology and the package are useful for customer support team managers to balance team members workload and obtain additional criteria for teamwork assessment. The criteria are based not only on statistics, but on resource communication assessment. In addition, the developed methodology and PM_SN package can be used by companies with customer support systems to optimize the support process by reducing the time spent on communication between teams and customers.

The rest of the paper is organized as follows. Section II provides an overview of the current state of scientific research in the field of extracting a social resource graph from a business process event log. Section III represents the main steps of the methodology of extraction and analysis of event log social graph on the example of PM_SN package functions, including the developed technique for extracting a social graph from ticket histories from the customer support systems. In Section IV, the functionality of the developed methodology and package is demonstrated by the example of optimizing the customer support system of a large retailer. Finally, conclusions are drawn regarding the obtained results, and directions for further research are indicated.

II. The overview of approaches to social graph extraction from event log

Despite the fact that today Process Mining is a popular research area, the topic of extracting a social graph from the event log is not covered much. This may be due to the rather recent emergence of this area, while the general attention is mainly focused on the process map discovery.

Apparently, the topic of extracting the social graph from the event log was first touched upon in [8]. The authors introduced four metrics for social graph extraction: a metric based on a possible causality, a metric based on joint cases, a metric based on special event types, and a metric based on joint activities. In addition, as part of this work, the MiSoN tool was developed to discover relationships between people based on corporate information systems analysis, and its application was demonstrated using the example of the Dutch national public works department. The metrics introduced by the authors are applicable for extracting social graphs from event logs of a specific type of corporate information systems. In these systems, each action requires the appropriate qualifications of the employee. Unfortunately, the metrics are poorly suited for dealing with ticket history that includes the same types of actions. The MiSoN is suitable for building social graphs but does not provide functionality for the analysis of the extracted graph with methods of social networks analysis or functionality for graph export.

The paper [9] describes the framework that represents an event log as a stream and builds a social resource graph within a time window. It allows to split the graph into communities and observe their evolution in consecutive time windows. The paper demonstrates the usage of the framework on the example of the Dutch Academic Hospital. The graph is extracted based on the Pearson correlation coefficient of each resource pair, which shows how the resources perform the same actions. This approach is not applicable to the extraction of the social graph from the ticket histories of the customer support system. Also, monitoring the evolution of social communities is a resource-intensive operation that limits its applicability to large social graphs.

The authors of [10] focused their research on large event logs; the social resource graphs for such logs are massive and not suitable for the visual analysis. The authors propose aggregating users in clusters using modularity community detection algorithms with consequent visualization. The paper also describes the result of applying the approach to the events log of the Portuguese hospital in Santa Maria de Feira. The developed approach can be considered applicable for the analysis of the communication of various technical support teams. Still, it does not allow each resource to be investigated separately, which is necessary for analyzing the social graph extracted from the ticket histories of the customer support system to balance the workload of the resources.

The authors of [11] extract a social graph of knowledge transfer between resources from e-mail chains. As a tool, they chose the MiSoN package developed as part of [8], which included in the software product for Process Mining ProM [12]. Authors modify the metric for graph construction and use correlation metrics and timestamps of events. It should be noted that correlation metrics do not preserve the transfer order of the process object but track the work of resources on the same object. In case of extracting a social graph from the ticket histories of customer support system, it is essential to maintain the order of the ticket transfer since it reflects templates of the standard procedure for solving the ticket. Therefore, this approach is not applicable to extracting a social graph from the ticket histories of customer support system.

It can be concluded that existing tools and approaches for extracting a social graph from the event log are mostly focused on corporate information systems. That is, they cannot be directly applied to the ticket histories of the customer support systems; besides, they do not provide tools for applying methods of social networks analysis to the extracted social graph.

III. Methodology

A. The event log

An event log L is a set of events related to a specific process P [9]. Each event e is identified by three mandatory attributes: an identifier of the process object – case ID, an identifier of action that is performed on object – activity, and a timestamp. Additionally, the space for describing events is expanded by adding identifiers of action executors (subjects or resources) R and other characteristics of an object or event. All events in the order of increasing the timestamp related to one object represent a trace T that constitutes an instance of a process execution variant [9].

In the context of a customer support system, the object is the ticket itself. The event log, in this case, contains the ticket histories. The event of such a process is any ticket update, for example, a change in a status, title or priority of an application, adding a comment or attaching a document. The identifier of the ticket is the case ID; the activity is the type of ticket change; the timestamp is the unified time that the customer support system saves when recording the change; the resource is represented by the identifier of the subject (resource) who made the ticket update.

B. Methodology steps

The methodology is focused on extracting a social graph from the event log and its subsequent analysis. Its realization the PM_SN package is developed using a high-level programming language R. The purpose of their creation is to obtain a tool for analyzing the nature of the resource's communication within a business process for its further optimization. By the time of paper publication, the methodology includes the following steps:

1) extracting a social resource graph from event log,
2) filtering the event log,
3) identifying a leader in the social graph of resources,
4) social resource communities detection,
5) resource social graph visualization.

Each step of the methodology has corresponding function in the PM_SN package.

1) Extracting a social graph from the event log: Based on the event log, it is possible to construct different types of social graphs, classifying them according to the types of connections between nodes. The authors of [8] distinguish four types of event log social graphs:

1) handover of work graph,
2) working together graph,
3) subcontracting graph,
4) similar task graph.

In the context of customer support systems, the most informative for analysis is the handover of the work graph. Therefore, the package currently supports the extraction of this type of graph. As mentioned above, the records of the event log concerning ticket histories are used.

In the extracted oriented graph G the node represents the resource that solves the ticket; therefore, such a graph can be called a social graph. Graph edge represents the work transfer from one resource to another. The edge e_{vw} connecting the nodes v and w shows that the resource w has completed some stage of ticket solving after the resource v. The strength of communication between resources is determined by the frequency of ticket transfer between them. Note that only edges that represent "strong" connections between resources are added to the final graph – the frequency of occurrence for the corresponding work transfer in the event log should be higher than the median of the frequency of all transfers. We proceed from the assumption that edges with weights less than the median reflect those work transfers that are rare and most likely are random. For example, a client created a ticket and assigned it to an incorrect technical support team, or a problem for which a ticket was created is so rare that the optimization of its solution is not advisable. On the contrary, within the extracted graph, which is used for further analysis of the company business processes, the most frequent traces of work transfer are of high interest because they describe standard behavior within the company. And their optimization helps to reduce the ticket solution time.

To extract the graph edges, it is necessary to identify all the traces of the tickets and to determine the weight of the edges; it is necessary to calculate the frequency of each work transfer occurrence. Additionally, for each graph edge, the time that was spent by edge ending node resource after the actions of the edge starting node resource is calculated. The following time characteristics are selected as indicators:

1) the total ticket solving time for this pair of resources,
2) the minimum ticket solving time for this pair of resources,
3) the maximum ticket solving time for this pair of resources.

The average processing time of a ticket can also be calculated using the total time and edge weight. The pseudocode of the developed algorithm for extracting the handover of the work social graph from the event log is presented as Algorithm 1.

Algorithm 1 Extract social graph algorithm

Input: event log L, strength threshold *th*;
Output: social graph *G*;
Initialization:*G*;
for all resource $r \in L$ **do**
  add node for r into *G*;
end for
for all trace $T \in L$ **do**
  sort events $e_i \in T$ in timestamp t_i ascending order;
  $v \leftarrow r(e_1)$;
  $t1 \leftarrow t(e_1)$;
  for $i \leftarrow 2$,event count in T **do**
    $w \leftarrow r(e_i)$;
    $d \leftarrow t(e_i) - t1$;
    if (*v*,*w*) edge exists **then**
      update (*v*,*w*) edge strength and time indicators in *G*;
    else
      add (*v*,*w*) edge in *G*;
      initialize edge strength and time indicators;
    end if
  end for
end for
Delete edges from *G* which strength is below the threshold *th*.

We emphasize again that the removal of edges whose weight is less than a given threshold allows us to exclude edges corresponding to random connections, thereby simplifying the resulting social graph. The package also provides the ability to remove self-loops from the social graph.

2) Event log filtering: The event log filtering function enables excluding records of the event log that have one of the following properties:

1) go beyond the specified time interval passed to the function as a parameter,
2) relate to a specific case with the identifier passed to the function as a parameter,
3) relate to a specific activity with the identifier passed to the function as a parameter,
4) related to an activity that is performed by a specific resource with the identifier passed to the function as a parameter.

Filtering reduces the noise level in the data and helps to focus on specific resources, for example, to exclude system accounts for teamwork analysis.

3) Identifying a leader in the social resource graph: Although there are system resource identifiers in the source event log, information about the type of resource may not always be available. Restrictions may be associated with the data confidentiality for some support teams. Besides, in the customer support system, roles are often assigned based on the functions performed, e.g., an 'agent' or an 'administrator', and not according to the team role. Identifying the leader in the social resource graph helps to find the nodes corresponding to the connecting elements of the customer support system – system accounts and managers since they are the intermediaries between members of different teams. Typically, such nodes are relevant at the first stage of analysis, during which their number and influence on the response time of other participants in the support system are determined. A more in-depth analysis shows that such nodes are redundant and complicate the understanding of the nature of the relationship between other resources since neither managers nor system accounts can be influenced by optimizing ticket solving.

The methodology involves the usage of any algorithm for identifying a leader in a social community using the entire social resource graph as a community. The PM_SN package utilizes the PageRank algorithm [7] to identify social resource graph leaders. The algorithm was designed to rank webpages for search engines based on the number of inbound and outbound links. In the context of the handover of work social graph analysis, edges are used instead of links, and nodes are ranked. Based on the specifics of the algorithm for extracting the handover of work social graph, we can conclude that the higher value of the leadership coefficient obtained using the PageRank algorithm is more likely to correspond to the node representing system account or manager.

4) Detection of social communities in the resource social graph: Today, there is a large variety of algorithms for dividing a social graph into social communities, but there is no universally accepted definition for the term "social community". In a large number of works on this topic, the social community is determined through its main property: the social community is a subgraph of the social graph in which internal connections are stronger than external ones [14]. In this case, modularity is used to assess the quality of the partition [7].

By dividing the graph into communities, it is possible to identify resources that usually work on the same ticket, that is, they perform different actions one after another in one or several scenarios. For example, if some part of the application, that receives data from one service and processes it using another service, does not work for the client, then it is necessary to check both services, which are served, as a rule, by different teams. In this case, the technical support of the first service checks its service and add a comment about its status, then the technical support of the second service checks it and also add a comment about its status.

Dividing a social resource graph into communities based on modularity can affect the optimization of different teams' communication but cannot help to balance the workload within one team. Another approach is required to combine resources, that perform the same activities within the ticket solution scenarios and can potentially replace each other, into one community, for example, the use of graph-cutting algorithms. One way to cut a graph is to sequentially remove from the graph the edges with the largest weight or the edges with a weight greater than a given threshold [15]. Such algorithms usually use a distance metric to calculate the weight of the edges.

In the case of cutting the social resource graph, the distance metric has to ensure that the nodes performing the same work get into the same community. That is why the value of such a metric for two nodes performing the same activities should be generally less than the value of a metric for nodes performing different activities. In addition, axioms defined by formulas (1) – (3) must be satisfied for the metric of the distance d between the nodes a, b and c:

$$d(a,b) > 0, d(a,a) = 0 \tag{1}$$

$$d(a,b) = d(b,a) \tag{2}$$

$$d(a,b) + d(b,c) \geq d(a,c) \tag{3}$$

Since the ticket solution scenarios are used in most cases, resource groups that execute the same part of the script have a similar structure: they are connected with the same nodes and the similarly directed edges. We introduce the following distance metric represented by formula (4):

$$d(a,b) = \begin{cases} 0 & a = b \\ 1 - \frac{|out_{adj\_node_a} \cap out_{adj\_node_b}|}{max(|out_{adj\_node_a}|, |out_{adj\_node_b}|)} + \gamma & a \neq b \end{cases} \tag{4}$$

where $out_{adj\_node_a}$ are the nodes that include the edges incoming from a, $out_{adj\_node_b}$ are the nodes that include edges incoming from b, γ is the regularization parameter, $\gamma > 0$.

Values of metric (4) lie in the interval [0; 1], while the value 0 is possible only if the nodes a and b coincide. It can be argued that the proposed metric satisfies all three axioms.

The algorithm of community detection in the handover of work social graph includes the following steps:

1) Construct an adjacency matrix for the social graph node. The matrix cells contain the values of the distance metric (4) between the pair of nodes.
2) Build a histogram of the distribution of distances between nodes and visualize it to the user to determine the threshold. If the graph is strongly structured, the histogram has a minimum between the two peaks [15]. However, since the communities within the graph are not always well separable and for the flexibility of graph partitioning, the user can assign a threshold manually.
3) Equate to 1 all values of distances that are greater than a given threshold and invert all values of the adjacency matrix by subtracting them from 1; thus an adjacency matrix with the weight of each edge is obtained.
4) Construct a new undirected graph based on the adjacency matrix by adding all nodes and edges with a weight greater than zero.
5) The resulting connected components are the desired social communities.

6) Move community tags to the original social graph.

The correctness of the algorithm can be validated during experiments involving an expert on the example of specific teams of the customer support system.

5) Resource social graph visualization: Visualization tools are an essential component of almost all existing graph analysis packages. High-quality graph visualization helps to detect dependencies and draw the right conclusions based on them. The package uses the opensource tool Cytoscape [13] to visualize the social graph.

Cytoscape meets all the requirements that have been presented to the visualization tool of the resource social graph:

1) The ability to display large graphs (more than 1,000 nodes) in an interactive mode.
2) The ability to dynamically set the color of the graph nodes depending on the parameter value.
3) The ability to dynamically set the graph edges width depending on the parameter value.
4) The ability to dynamically set the size of graph nodes depending on the value of the specified parameter.
5) The function of displaying the specified parameter as the name of the graph nodes.
6) The function of displaying the specified parameter as the name of the graph edges.

A procedure included in the PM_SN package exports the graph with all its attributes to a CSV file suitable for loading into Cytoscape. This file is a list of edges with attributes. The columns of the file and their purpose are presented in Table I.

TABLE I. SOCIAL GRAPH FILE FORMAT FOR CYTOSCAPE

Column	Purpose
from	The name of the start node for the edge.
to	The name of the end node for the edge.
freq	The edge strength.
total_time	The total work transfer time from “from” node to “to” node.
avg_time	The average work transfer time from “from” node to “to” node.
max_time	The maximum work transfer time from “from” node to “to” node.
min_time	The minimum work transfer time from “from” node to “to” node.
width	The edge width. It equals the multiplying “freq” by a a coefficient ($a = 0.01$).
group	The social community identifier of the start node for the edge.
sys_group	The customer support system internal team identifier of the start node for the edge.
leader	The leadership coefficient of the start node for the edge.
size	The start node size. It equals the multiplying “leader” by a b coefficient ($b = 1,000$).
color	The start node color in hex format.

The Cytoscape tool has a CyREST API that allows rendering a graph automatically without using a transfer file. Implementation of this functionality is planned for the next version of the PM_SN package.

IV. CASE STUDY

Next, we consider two scenarios of applying the developed methodology for the analysis of communication within the technical support team of the Oracle Retail Price Management (RPM) subsystem [16] of the Oracle Retail Merchandising System (RMS) [17] of a large European retailer during the second half of 2019. To organize technical support, the retailer uses the Zendesk system as a customer service system and its database stores the event log under investigation.

A. Description of the event log

The event log was obtained by uploading the ticket histories for the RPM technical support team from 09/01/2019 to 12/31/2019 from the Zendesk system using the Support REST API [18]. The event log is a CSV file, which contains 182,611 records for 2,283 tickets and 646 unique resources. The format of the records is presented in Table II.

TABLE II. EVENT LOG FORMAT

Column	Purpose
case_id	The ticket number.
activity	The activity identifier, for example, “Comment” – the comment was added,“ChangeStatus” – ticket status was changed.
timestamp	The timestamp when the event was recorded in the Zendesk system.
resource	The identifier of the resource who executed the activity. It may be a technical support team member or system user.
value	The new value of the ticket changed field.
via	The identifier of the channel which was used to execute the activity, for example, web-interface, API or mobile app.

In addition to the event log, a list of resources, that participated in the ticket solution, was uploaded with the names and identifiers of the corresponding Zendesk groups. As a rule, users who are not related to any technical support team and only create tickets do not belong to any Zendesk group. The uploaded event log has been anonymized.

The first records of the uploaded event log were dated 2017 due to the several tickets that were created before September 1, 2019, and were closed after September 1, 2019, since the upload was made by the ticket closing date. Ticket solution that takes longer than 1 month according to the business rules of the company is unacceptable and requires a separate analysis; therefore such tickets were identified and removed from the event log using the filtering function of the PM_SN package.

After filtering the event log, 160,778 entries remained for 2,103 tickets and 568 unique resources. The time distribution of the events from the log is shown in Fig. 1.

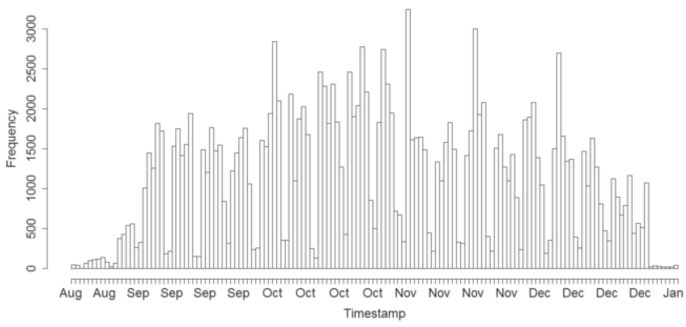

Fig. 1. The day distribution of the events from the log

The histogram shows that the technical support team works 7 days a week, and, on weekends, the team workload is significantly less than on weekdays. According to the expert opinion, business users work and accordingly create tickets only on weekdays. On weekends, tickets are created automatically at the urgent priority level and must be resolved within a few hours. Also, note that during one week the workload is distributed unevenly, but there is no periodicity between different weeks. Thus, more resources are required on weekdays than on weekends, when one or two people are enough.

B. Use case 1: Determining the success of a probationary period for an employee

As the first scenario of applying the methodology, we determine whether an employee who is going through a probationary period has the same functions as his team. This helps to make a decision about the success of the probationary period, relying not only on the subjective opinions of the support team manager and the employee mentor. To do this, we identify subgroups of employees within technical support teams by constructing a social graph based on the event log described in the previous subsection. We use the corresponding function of the PM_SN package, passing as a threshold value equal to 5 since the calculated average value of the node weight is 4.95, with a median of 1. The resulting graph, which contains 136 nodes and 653 edges, is shown in Fig. 2.

Fig. 2. The social graph extracted from the event log

In Fig. 2, in the center of the graph, a cluster of nodes with a large number of incoming and outgoing edges is clearly distinguished; these nodes are involved in the standard trace of ticket solution. The nodes that are closer to the boundaries of the image and contain only 1 or 2 thin incoming or outgoing edges are nodes corresponding to the users who only create tickets. The creation of such tickets takes place in unusual situations since such an action was performed only 5 times.

Next, we detect the communities in the resulting social graph using the function of the PM_SN package. To determine the threshold value, we construct a histogram of the distance matrix values distribution excluding a column with values from 0.9 to 1, since most of the distances fall into it, which does not allow us to determine the dynamics at other intervals. The resulting histogram is shown in Fig. 3. Peaks are visible on it in the intervals 0 – 0.05, 0.45 – 0.5 and 0.6 – 0.75. The peak at zero refers to the nodes that completely coincide, the next peak is the threshold for distances within the community and between communities, therefore, we choose 0.45 as the threshold value.

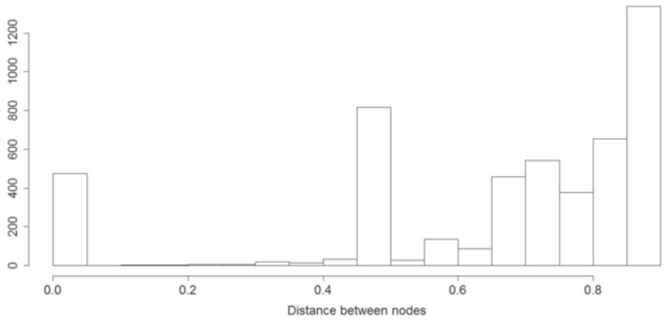

Fig. 3. The histogram of the distribution of distances between resources

In addition to the detected social communities' labels, we use the labels of the system groups for each resource. We visualize the result by grouping the nodes by belonging to a Zendesk system group. Also, we mark the labels of detected social communities using nodes' shape. Let us consider in more detail the RPM technical support group since we are interested in the employees of this particular team. The result is presented in Fig. 4.

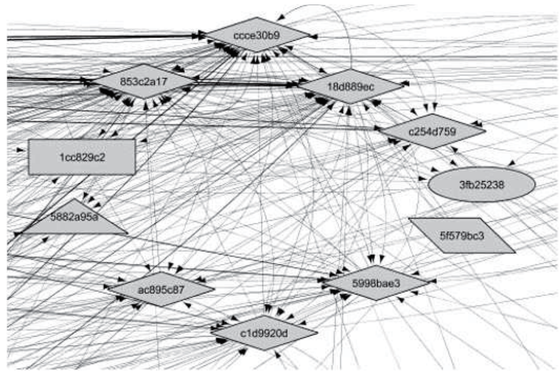

Fig. 4. The social resources communities within the RPM Technical Support Team

The employees completing the probationary period are 853c2a17, 18d889ec, 3fb25238 and 1cc829c2. Note that most of the team fell into one social community, it is marked with a rhombus, which means that these resources can replace each other to balance the workload. The nodes 18d889ec and 853c2a17 fell into it, therefore, they perform the same functions as most of the technical support team members and their probationary period can be considered successfully passed.

The nodes 3fb25238 and 1cc829c2 did not fall into the community marked with a rhombus, besides, they fell into different social communities since the shapes of their nodes differ: oval and rectangle, respectively. We can conclude that both of them do not yet perform the functions that are required from a member of the support team.

Note that the nodes 5882a95a and 5f579bc3 did not fall into the community marked with a rhombus, because the first is just a bridge between the manager and the team, and the second is not an employee of the support team but helps the team to implement the support of new RPM system's element.

In this way, the manager of the RPM technical support team gets an additional tool and an objective criterion for deciding whether the employee completed the probationary period successfully or unsuccessfully, which is essential because the personnel change rate in the team is quite high.

C. Use case 2: Analysis of the efficiency of system notification

As a second scenario for applying the methodology, we analyzed the response time of the RPM technical support team to the system notification user 3c507a8c compared to the average response time to determine the efficiency of using the notification function in the Zendesk system. This type of notification is used for a ticket with high priority creating – according to the regulations, such a ticket must be responded within 15 minutes. It is not possible to analyze the response time of the technical support team to the notification system user using Zendesk tools or using the Zendesk-integrated reporting and analytics service GoodData [19].

To determine the response time of RPM technical support team members, the resource social graph was visualized using the Cytoscape tool. The nodes' shape indicates that the node belongs to a Zendesk system group. Using the package filter, the node corresponding to the system user 3c507a8c and all its neighboring nodes along outgoing edges were left in the graph. Each edge displays the average response time of resources to notification in minutes. The result is presented in Fig. 5.

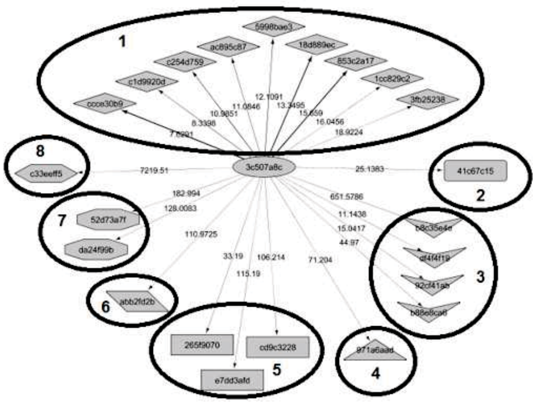

Fig. 5. The subgraph of the resource social graph showing the response time of resource to the notification

Fig. 5 shows that the RPM technical support team, the nodes of which are grouped in region 1, on average responses to the notifications within 12.7 minutes with a time limit of 15 minutes, while the average response time for a team is 20 minutes. Note that the nodes 853c2a17, 1cc829c2 and 3fb25238 exceed the time established by the regulations, but these nodes, as mentioned above, have been passing a probationary period, and therefore such excesses of time are permissible.

In this way, the results of the case study confirm compliance with the notification response regulations by the RPM technical support team and demonstrate the effectiveness of this Zendesk system functionality. The notification functionality can be extended to other ticket events, not only its creation.

V. CONCLUSION

The methodology and its realization, PM_SN package, presented in the paper make it possible to extract a social graph from the event log of the customer support system for the subsequent optimization of the resource communication and balancing resource workload. With their help, managers of the support teams and managers of the entire company can discover the stages of the ticket solution by different teams or employees that take place in practice. Managers, also, can quickly obtain information about delays at each stage of the solution to determine the degree of workload and interchangeability of resources or to decide whether it is time to open a new vacancy or to dismiss an employee or to assess the success of an employee probationary period. In addition, with the methodology, it is possible to evaluate the feasibility of using additional features of the customer support system, for example, the system notifications, based not only on general statistics.

A review of methods and approaches of social graph extraction from the event log of the company's information system (for example, WFM, ERP, CRM systems) highlighted their main differences from approaches to extracting social graphs from the event log of customer support systems. The last focuses on the uniformity of employee activities with different qualifications, the importance of analyzing the communication of different support teams and individual employees, and maintaining the order of transfer of the tickets, which must comply with standard templates in the ticket solution.

At the time of the paper publication, the package implements 4 main functions:

- the function of extracting a social graph from the event log;
- the function of filtering the event log by time, a case, an event or resource;
- the function of communities detection in a social graph;
- the function of identifying a leader in a social graph.

In addition, the PM_SN package supports the file export function for visualizing a social graph using the Cytoscape tool.

Some scenarios of applying the methodology have been demonstrated in the example of the RPM subsystem technical support team of a large European retailer. The paper describes scenarios of event log extraction for analysis, detection of a subgroup of employees within the technical support teams that perform the same functions, and analyzing the reaction of the technical support team to the system notification user.

In the course of further PM_SN package development, it is planned to implement the function of visualizing the social graph and supporting the automatic adjustment of threshold parameters for the functions of extracting the social graph from the event log and community detection in the social graph.

REFERENCES

[1] Zendesk, Web: https://www.zendesk.com/

[2] Okdesk, Web: https://okdesk.ru/

[3] Freshworks Freshdesk, Web: https://freshdesk.com/ru/

[4] Atlassian Jira Service Desk, Web: https://www.atlassian.com/ru/software/jira/service-desk

[5] Kayako, Web: https://www.kayako.com/

[6] W. van der Aalst, *Process Mining: Data Science in Action*. Heidelberg: Springer, 2016.

[7] K.A. Slavnov, "Analiz social'nyh grafov (Social graph analysis)": Bachelor's thesis, Moscow, 2015. http://www.machinelearning.ru/wiki/images/6/60/2015_417_SlavnovKA.pdf

[8] W. van der Aalst, H. Reijers and M. Song, "Discovering Social Networks from Event Logs", *Comput. Supported Coop. Work*, vol. 14, Oct. 2005, pp. 549-593.

[9] A. Appice, M. Di Pietro, C. Greco and D. Malerba, "Discovering and Tracking Organizational Structures in Event Logs", *in New Frontiers in Mining Complex Patterns. NFMCP 2015. Lecture Notes in Computer Science*, vol. 9607, May 2016.

[10] D.R. Ferreira and C. Alves, "Discovering User Communities in Large Event Logs", *in Business Process Management Workshops. BPM 2011. Lecture Notes in Business Information Processing*, vol. 99, 2012, pp. 123-134.

[11] M. Creemers and M. Jans, "Social Mining as a Knowledge Management Solution", *in Proceedings of the CAiSE'16 Forum, at the 28th International Conference on Advanced Information Systems Engineering (CAiSE 2016)*, vol. 1612, Ljubljana, Slovenia, June 2016, pp. 57-64.

[12] Process Mining Group of Eindhoven Technical University, ProM Tools, Web: http://www.promtools.org/doku.php

[13] P. Shannon, A. Markiel, O. Ozier, N.S. Baliga, J.T. Wang, D. Ramage, N. Amin, B. Schwikowski and T. Ideker, "Cytoscape: a software environment for integrated models of biomolecular interaction networks", *Genome Research*, vol. 13, issue 11, Nov. 2003, pp. 2498-2504

[14] S. Fortunato, "Community detection in graphs", *Physics Reports*, vol. 486, num. 3-5, 2010, pp. 75-174.

[15] I.D. Mandel', *Klasternyj analiz (Cluster analysis)*. Moscow: Finansy i statistika, 1988, 176 p.

[16] Oracle Retail Price Management, Web: https://www.oracle.com/industries/retail/products/merchandise-management/price-management/

[17] Oracle Retail Merchandising System, Web: https://www.oracle.com/ru/industries/retail/products/merchandise-management/merchandising-system/

[18] Zendesk Support API, Web: https://developer.zendesk.com/rest_api/docs/support/introduction

[19] GoodData, Web: https://www.gooddata.com/