Process Mining in Manufacturing Company

With Focus on Process Simulation and Prediction.

Milan Pospíšil
Department of Information Systems
BUT, Faculty of Information Technology
Brno, Czech Republic
ipospisil@fit.vutbr.cz

Vojtěch Mates
Department of Information Systems
BUT, Faculty of Information Technology
Brno, Czech Republic
imates@fit.vutbr.cz

Tomáš Hruška
IT4Innovations Centre of Excellence
BUT, Faculty of Information Technology
Brno, Czech Republic
hruska@fit.vutbr.cz

***Abstract*—Simulation can be used for analysis, prediction, and optimization of business processes. Nevertheless, process models often differ from reality. Data mining techniques can be used for improving these models based on observations of process and resource behavior from detailed event logs. More accurate process models can be used not only for analysis and optimization, but for prediction and recommendation as well. This paper analyses process model in manufacturing company and its historical performance data. Based on that observation, simulation model is automatically created and used for analysis, prediction and for dynamic optimization.**

Keywords-business process simulation; business process intelligence; data mining; process mining; prediction; optimization; recommendation

I. Introduction

Classic simulation can be used for the analysis of business processes. It is useful to try many variants of processes, measure the effects, and then decide on the optimal process settings. For example, the process can be redesigned, it is possible to change resource allocation, and search for the most optimal configuration with respect to context-based requirements (price, effectiveness, customer satisfaction, etc.). The current process configuration can be tested for how many cases it can handle over periods of time.

These models can be built manually, which is time consuming and error prone. The main disadvantage is that this approach cannot be used for predictions for operational decision, but only for strategic decisions. The operational decisions are important for internal logistics purposes. The casual models have some simplifications – for example probabilities of routing and naive execution time of task. These parameters are set based on long observation of processes, so they can work in long-term simulation for strategic decisions. Nevertheless, operational decisions need short-term simulation. These two simulation types differ significantly. Short-term simulation starts in current state of the process with allocated resources, cases in progress with known parameters and with waiting cases to handle. Routing probabilities and execution times can differ significantly for different case parameters, thus mine deeper dependencies is needed.

For example, assume repair process, there are two tasks – repair basic item and repair advanced item, repair basic item is executed in 90% of cases, repair advanced only in 10% of cases. Execution time of basic item is about one hour and execution time of advanced item is about eight hours. If our case has known attributes and it is usually available in runtime, based on data mining, which these attributes lead to advanced repair with 80% probability, classic routing probabilities are precise enough to be used. And there is another problem – execution time of task is also influenced with case attributes – some case attributes leads to longer execution time. Resources have to be also taken into account, e.g. some people work faster, some slower.

Predictions, recommendations, and dynamic optimizations could be accomplished by operational simulation. The system can warn us, that some cases will be probably late based on historic performance data. Then some different scenarios can be simulated and evaluated, then the system can recommend us actions and provide dynamic optimization of current running cases – for example; assign extra resources from non-critical case to critical, or use a different sub-process – when we have a slower / cheaper version or faster but more expensive.

This paper analyses processes of manufacturing company. Simulation model is built using process mining and used for predictions. Based on these predictions, managers can change priorities (reallocation of resources) or better plan their storage space, because working front is known, therefore they can better predict manufacturing time.

This work is based on our previous research and verifies our theory on process mining and simulation field [15].

II. RELATED WORK

Data mining techniques can be used in Business Process Management. This new area was called Process Mining [3, 6, 12, 13, 14]. It was based on analysis of information from event logs that were produced by business processes. Process discovery (figure 1) is one of the methods and it is able to find a process model from an unknown process using many sequence examples of tasks and case parameters.

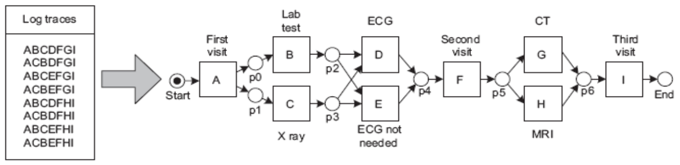

Figure 1. Process discovery (taken from [5]). It possible to discover a process model from log. The discovered process model must be able to replay most log traces.

Not only process model could be discovered, but also decision rules and social networks [5, 6, 10] and simulation models [5, 10, 11, 15]. Resource behaviour is also point of interest [8, 9]. Example of simulation model [5] is depicted in figure 2. It is possible to see routing probabilities and decision rules (decision rules are used when case attributes are known – that leads to better routing rules) and it is possible to see time distribution of tasks.

Some other research on process prediction was published in [1, 2, 4, 7, 10]. Wetzstein [4] used decision trees to analyse process performance in figure 3. As it can be seen, response time of banking service is higher than 210, KPI (key performance indicator) is always violated. If customer id is 123, manager can observe process bottlenecks, he can try to make banking service faster or find out why customer 1234 has problems.

Grigori [1, 2] uses similar approach, not for analysis, but predictions. Huge classifier is learned based on case attributes, start, and end time execution of tasks. Classifier can predict final time execution of case based on case parameters and time information from executed tasks. Evaluation of that approach compared to our approach is discussed in [15]. In addition, our work uses similar approach as [1, 2, 4] but it combines it with process mining.

Finally, when we mine deeper dependencies about routing rules and execution time of cases, we can use it for simulation for decision support [15]. Our previous work [15] is extension of papers [5, 10] and it adds some important features, some inspired by papers [1, 2, 4]. For example, execution time of cases is also predicted by classifier like decision rules.

This paper shows theory of [15] can be applied in real large manufactory company.

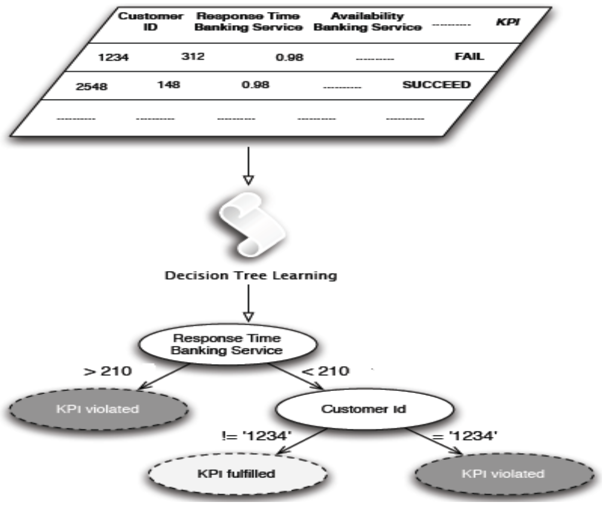

Figure 3. Process performance analysis (taken from 4). Decision tree is used for discovering factors that leads to KPI violation. We can see that KPI is violated when response time of banking service is larger than 210.

III. MORE PRECISE SIMULATION MODEL

As it is said in [15], there is need to build more precise model than the one described by papers [5, 10]. We will describe steps needed to accomplish that:

A. *Process Discovery*

If process is not known, it is possible to discover it using process discovery techniques. However, process discovery is not the most needed method for building simulation model. If explicit model is not present then it is possible to discover it, but the precision of the model will be lower than explicitly given real model. In some companies, discovered

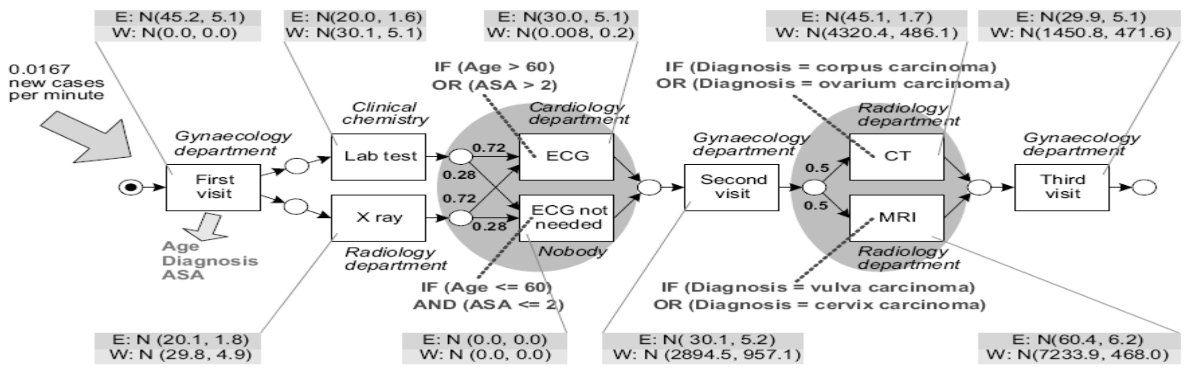

Figure 2. Simulation model [5]. Classic simulation model (taken from [5]) is enhanced by decision rules. Decision rules can make our routing probabilites more precise, because they depend on case attributes.

model could be more precise than official model but this is because these companies do not have their model formalized so well. This is not the case for manufacturing companies where prediction and usage of short time simulation is considered better.

B. *Decision Mining*

Decision mining is based on discovering routing rules in OR split nodes. These rules could be available too but sometimes they are not applicable. Assume situation, routing rule is based on one parameter that is inserted into system just before the decision. Thus, our predictor will know next path only in the time of decision – useless prediction. In these situations, decision mining has to be used. Decision mining is described in [5, 10, 15]. Classifier is learned on training data where inputs are case attributes and output is next path in process. Our work [15] describes another problem and that is missing attributes or 100% precise attribute known in the time of decision inserted by human (described earlier). If some attributes are missing then classic classifiers will not work in proper way. If there is 100% precise attribute then classifier is based only on that attribute. Solution is the same for both problems – it is necessary to build several classifiers for several milestones of the process – from the start (only subset of case attributes are known) to the end (all attributes are known).

C. *Execution Time of Tasks*

Execution time of tasks is the most important issue in short-term simulation. Process model and routing rules are important as well. However, in companies with predictable business processes (especially manufacturing companies), control flow and routing rules are used to be formalized.

Execution time of tasks will be described precisely in Section V.

IV. MANUFACTURING COMPANY

Our manufacturing company produces doors. Doors have their attributes (about twenty) and based on attributes, different operations are executed. Doors have different material, size, weight, different corner and edge types, different handle and glasses, etc. Every door has its ID and it can be modeled as case. Doors are manufactured in machines (tasks). Some machines work in parallel; some machines are bound to several tasks, so these machines must be treated as resources, because machine could be busy or working. People are working with machines or in manual workplaces. Routing probabilities are 100% accurate, because doors with specific attributes must be manufactured only by specific machine and with specific settings.

Resources are quite predictable, because they work on shifts and they are always available and planned several days ahead. The only unknown parameter is execution time of tasks that depends on case attributes – every case is modeled as one door, so case attributes are door parameters. Door parameters are known at the beginning of the process and are constant, so there is no need to build several classifiers for several periods of case execution [15]. Execution time also depends on people work rate, work queue and error rate (especially in manual workplaces), but this is issue beyond the paper.

Context-based predicting execution time of tasks quite precisely can help with several issues. First, managers can decrease storage spaces, because they could plan execution order of cases in order to decrease waiting times. Our prediction decreases variances of execution time and logistics have methods to plan storage spaces when there is low variance. They will also know if some doors will be probably late and for example, they can respond to that changing priorities, resource allocation, etc. Another useful issue is the analysis. Managers could measure which door types takes long time to produce and better calculate their price. For logistics, execution time is not as much important as influence of variance of execution time. It is possible to measure which door types (based on parameters) have high variance. Managers can focus on that door types and try to find out the cause of high variance, or produce them only in situations (if it is possible to wait) when variance is not such important issue.

V. PREDICTION OF EXECUTION TIME OF TASKS

The time deviation is sometimes high, but it can be decreased by data mining techniques. Thus, it is useful to examine data and find relationships between case parameters and execution time for each task in process. This can be solved as a classification problem, where case parameters are input attributes and execution time is the target attribute.

A. *Classification and Prediction Models*

There are number of classification models, every model has its advantages and disadvantages based on data type used for classification. Our problem is rather prediction than classification, but both terms are similar and many models support both of them.

In our case, we have 18 case attributes and one numerical target attribute. All attributes are categorical. Yes, some of them are numerical (width, height), but they are standardized to only few distinct values, so they can be numerical or categorical depending on requirements of classification/prediction model. What is more difficult for prediction, it is also our case, is that target attribute varies even for cases with the same attributes. This is typical for execution time, because work is performed not only by machines, but also by people and people do not work in coherent speed.

Another problem is high variability of door types. In manufacturing company, it is possible to make several millions variants of doors. This causes problem in prediction, because it is difficult to obtain enough data for prediction, it needs many examples. Attributes can also pod up high number of distinct values, it corresponds with high variability of door type (this is problem for neural network classification).

In the next section, some prediction models will be described and discussed its applicability.

B. *Neural Network*

We tested Neural network approach, but results were not satisfied. Neural network was not able to learn. It was caused by high number of input neurons - 303. Every categorical column had to be transformed to new columns. Every distinct value of that column created new column, which holds 1 or 0. So for example, column corner has four distinct values – left, right, top, and bottom. It creates four new columns that can acquire value 1 only once for a row (for the columns that belong to one categorical column). That transformation was necessary, because neural network can handle only numerical attributes. Target attribute was divided into several intervals and every interval was modeled as a single output neuron.

We think that network was not able to learn because of high number of inputs compared to number of training examples and mainly because of variability of output, (even identical training examples had little different outputs). Thus, we think network is not sufficient for our problem because of high number of categorical attributes and variability of target attribute.

C. *K-Nearest Neighbour (KNN)*

The method is based on simple idea of finding several examples from training set closest to input pattern. We simply computed number of differences of attributes between training example and input pattern. These differences (0 or 1, equals and not equals) were weighted. Weight of every attribute was computed by the same method described below in regression tree. Higher weight means that attribute have higher influence of execution time and it is considered more important. Then twenty nearest examples were given and mean, min, max and deviation of time was computed (we measured only mean, but deviation is also important in simulation and it is good indicator of supposed reliability of prediction).

Results (figure 4) were quite satisfied (there is only subset of real workplaces). We have compared prediction to simple algorithm – prediction based on mean of all execution time. Simplest predictor is the predictor that assumes mean value for every example. Differences in table are mean of all differences between real value and predicted value for every tested example.

Result was computed as follows: prediction was compared to most simple predictor that supposes always-average value of task execution time for all records. So:

Mean diff = $\sum \lvert \text{mean} - \text{real value} \rvert$
Predictor diff = $\sum \lvert \text{predicted value} - \text{real value} \rvert$
Final Score = Predictor diff / Mean diff

Mean and Predictor difference is computed as sum of differences over all tested examples. Mean difference is absolute value of mean and real value and Predictor difference is computed from predicted value and real value. Final ratio equals ratio of predictor difference and mean difference.

We have run test with 600 examples and compared them to dataset that contained about 10-20 thousands records for every workplace. Rating was computed as a ratio between difference computed by algorithm and difference computed by mean. So, result 0.5 means that we have decreased the variance of execution time of task about 50 %.

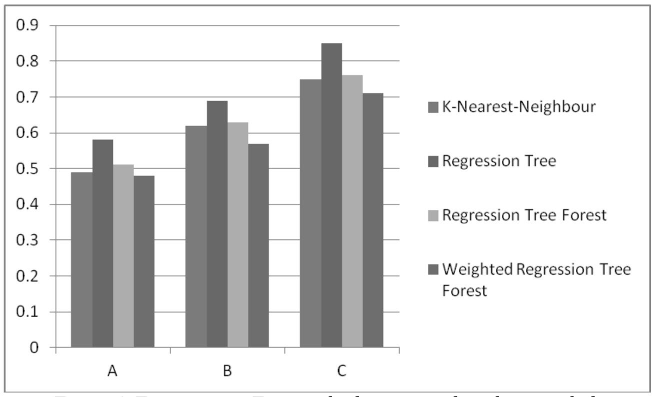

Figure 4. Experiments. Four methods were used on three workplaces. Note that a higher column means lower precision. Workplace A is machine does not depended on resource skills, workplace B is workplace with dependence on resource skills and workplace C is manual workplace (packaging) that does not depend so much on door type, but on resource performance.

Figure 4 shows that some results are satisfied, others not. For example, ratio of workplace A is good, Workplace C is not. Nevertheless, it is not the problem related to method, execution time is not based so much on attributes. It is because workplaces C perform packaging and that type of work is naturally quite independent of door types.

D. *Regression Tree*

Decision tree is popular model. It is simple, readable by human, and quite fast. Precision has not as satisfied results as K-Nearest Neighbour. However, the classification speed is several hundred times faster. Regression tree is decision tree with numerical target value. Nodes contain information about mean, min, max, and deviation of predicted value. Learning algorithm is similar to decision tree, but selection of split nodes differ. We have numerical target attribute so algorithm can be like that:

- For every column.
- For every distinct value of column.
- Take all target values of column grouped by current distinct value and compute deviation.
- 1 / Mean of all deviations is decision power for column.

This algorithm is similar to entropy computation, which is computed for categorical target value. The deviation is closed to entropy, because lower deviation points to better decision power. Computing of deviation can be also weighted by count of rows of groups divided by distinct values of column – distinct value with more rows should be more important. We have tried both approaches, but no significant precision difference was observed, even maybe precision was little lower. Algorithm described above works similar to ID3 algorithm. C4.5 algorithm has been also tried, but no significant difference has been found. Post-pruning was based on removing nodes with low row count (every node corresponds to subset rows of whole data set), because nodes with low row count are not representative.

Regression Tree had worse precision than K-Nearest-Neighbour (Ratio was about 1.2 – 1.3 times worse), but had also several advantages. It is more readable to human and it can be used to examine some properties of tasks – for example which combination of attributes positively or negatively affects execution time or which combinations of attributes have little ratio of prediction – that is represented by deviation of target values corresponding to some node of tree.

E. Regression Tree Forest

Regression Tree Forest is based on several Regression Trees. One extreme example is Random Forest. Random Forest creates many decision trees (more than one hundred) using classic (ID3 or C.45) algorithm with several differences:

- Every tree randomly selects subset of rows from training set (about 2/3).
- Every tree randomly selects subset of attribute columns (about 2/3)
- Every tree is not pruned and full-grown.
- Predictions made by voting of all trees by computing mean.

It is known that Random Forest is very precise model and still quite fast, because it is semantically similar to K-Nearest-Neighbour algorithm. Because learning time is quite long (it requires more than one hundred trees), we found it not suitable for real-time decision support. However, we have tried some trade-of between Random Forest and normal Decision Tree. We created several (about ten) trees and enforced different first splitting column for every tree. Enforced columns were ordered by their decision power. Thus, first tree root node begins with first (best) column; second tree root node begins with second column, etc. In addition, every tree randomly selects 70% of dataset and 70% decision attributes as it is said in Random Forest algorithm. Trees were pruned (opposite to Random Forest, which is not pruned) to about 10 min rows in a node.

It should be stress out that in normal Random Forest, result is computed by mean of all tree results. We selected best tree result by looking to the deviation of tree node. Best prediction could be measured by deviation of particular rows covered by tree node. Node with lowest deviation wins. This rule was necessary, because mean of all tree votes gave terrible results – mainly because we had only low count of trees compared to Random Forest.

Then we chose another improvement – tree result (mean and deviation) was not computed only by looking at the leaf node, but also taken into account the parent node. We computed mean by both node mean values weighted by their deviation (if child node had much better result – low deviation – than parent, its result will have bigger vote). That improvement was tested also in single ordinary Regression Tree and it increased precision too, but only slightly.

Our Tree Forest greatly improved accuracy of classifier, ratio was only about 1.05 times worse than K-Nearest-Neighbour and still order of magnitude faster that K-Nearest-Neighbour, which applicability could be problematic in real time monitoring for every tasks. Similar results can be explained, because random forest works similar to K-Nearest-Neighbour. It returns items that are close (by attributes) to predicted item, but it uses tree searching instead of searching in whole table.

VI. Execution Time and Resources

There is a little problem with resources. The resource information can be treated as normal case attribute, because it surely has impact on execution time of task, but there is a catch. For example, if we allow decision tree to build tree using resource attribute, final leaf will contain only records that have been executed only by that resource. This could enable problems, because sometimes, it is better to look for more examples, even from another resource. However, if we do not have such training examples and resource performance does not differ too much from other resources, it is good idea to look also to another resource records and consider them.

Second problem is related to dynamic changes. Even if the process is the same (e.g. technological process), workers performance could change over time. More experienced workers may be faster, so our algorithm could be prepared for that. We recommend following method, which little improved prediction in our manufacturing company.

Suppose K-Nearest-Neighbour or Regression Tree (or Forest) classifier. All that classifiers could be implemented to return set of records rather than final prediction (mean and deviation). The result (mean, deviation) could be implemented over those records, but with different weights. First, records that belong to the resource, which performance time is now predicting, should have bigger weight (for example two times higher) than other records. And second, these records (of our resource) should be considered in time plane. Newest records should have also bigger weights (for example two times bigger than oldest). Why is not possible to take into account time plan also to other records (another resources)? It is because we do not know about them so much in order to take into account their improvement and skills compared to our resource. This could be issue to another paper.

VII. Final evaluation

We have tested three tasks: One machine with little human interaction, second machine with manual work and third packaging with little dependence on door type, but with dependence on resource. As it was presented, Regression tree is always worse than other methods, while Regression Tree Forest is as good as K-Nearest-Neighbour, because it is optimized K-Nearest-Neighbour. Last method was weighted Regression Tree Forest. Weighting was described at Section VI. As it has been shown, weighting on workplace A did not improve result at all, because machine works independent on resources and time (it does not learn to work faster). In Workplace B and C, there was improvement. Workplace C had worst results, because packaging is not dependent on door type too much, but it is dependent on resource – we can see that weighting slightly improved performance.

VIII. Conclusion

We have tested our theory and our results were quite promising. It has been shown that the quality of results does not depend only on our methods, but mainly on manufactory itself. For example, if execution time cannot be predicted from case attributes in wanted precision, prediction will be useless. In our company, predictions helped lower execution time variance, which is very useful in internal logistics planning, but there is a question what precision is needed to implement some better planning techniques, which enables significant saving especially in space and time need for manufacturing production by improving input data for planning algorithms. We can also find subset of case parameters that have low time deviation and try to optimize their production. Other cases could be produced in another time or in other machines in parallel with another approach (slower but more robust). Some workplaces had bad time variance, but that were some manual workplaces like packaging, that were at the end of the process, so variance was such important issue.

Resources working speed was the biggest issue. There is not so much research in that very important area. In addition, dynamic aspect of process (new machines, resource improvement) is problem to solve. We believe these methods could reach maturity and could be used in some manufactories in future.

Acknowledgment

This research was supported by the grants of MPO Czech Republic TIP FR-TI3 039, the grant FIT-S-10-2, the research plan no. MSM0021630528 and the European Regional Development Fund in the IT4Innovations Centre of Excellence project (CZ.1.05/1.1.00/02.0070).

References

[1] D. Grigori, F. Casati, M. Castellanos, U. Dayal, M. Sayal, and M.C. Shan, “Business Process Intelligence”, Computers in Industry, Volume 53, Issue 3, Process / Workflow Mining, April 2004, Pages 321-343, ISSN 0166-3615, DOI: 10.1016/j.compind.2003.10.007.

[2] D. Grigori, F. Casati, U. Dayal, and M.C. Shan, “Improving Business Process Quality through Exception Understanding,Prediction, and Prevention”, Proceedings of the 27th VLDB Conference,Roma, Italy, 2001, 1-55860-804-4

[3] W.M.P. van der Aalst, H.A. Reijers, A.J.M.M. Weijters, B.F. van Dongen, A.K. Alves de Medeiros, M. Song, and H.M.W. Verbeek, “Business process mining: An industrial application”, Information Systems, Volume 32, Issue 5, July 2007, Pages 713-732, ISSN 0306-4379, DOI: 10.1016/j.is.2006.05.003.

[4] B. Wetzstein, P. Leitner, F. Rosenberg, I. Brandic, S. Dustdar, F. Leymann, "Monitoring and Analyzing Influential Factors of Business Process Performance," Enterprise Distributed Object Computing Conference, 2009. EDOC '09. IEEE International, pp. 141-150, 1-4 Sept. 2009, doi: 10.1109/EDOC.2009.18

[5] A. Rozinat, R.S. Mans, M. Song, and W.M.P. van der Aalst, “Discovering simulation models”, Information Systems, Volume 34, Issue 3, May 2009, Pages 305-327, ISSN 030

[6] M Song and W.M.P. van der Aalst, “Towards comprehensive support for organizational mining”, Decision Support Systems, Volume 46, Issue 1, December 2008, Pages 300-317, ISSN 0167-9236, DOI: 10.1016/j.dss.2008.07.002.

[7] W.M.P. Van der Aalst, “Business Process Simulation Revisited”, 2010, ISSN: 1865-1348

[8] J. Nakatumba, A. Rozinat, and N. Russell, “Business Process Simulation: How to get it right”, 2010,Springer-Verlag,doi=10.1.1.151.834

[9] J. Nakatumba and W.M.P.V.D. Aalst, "Analyzing Resource Behavior Using Process Mining", in Proc. Business Process Management Workshops, 2009, pp. 69-80.

[10] A. Rozinat, M.T. Wynn, W.M.P. van der Aalst, A.H.M. ter Hofstede, and C.J. Fidge, “Workflow simulation for operational decision support”, Data & Knowledge Engineering, Volume 68, Issue 9, Sixth International Conference on Business Process Management (BPM 2008) - Five selected and extended papers, September 2009, Pages 834-850, ISSN 0169-023X, DOI: 10.1016/j.datak.2009.02.014.

[11] W.M.P. van der Aalst, M.H. Schonenberg, and M. Song, “Time prediction based on process mining”, Information Systems, Volume 36, Issue 2, Special Issue: Semantic Integration of Data, Multimedia, and Services, April 2011, Pages 450-475, ISSN 0306-4379, DOI: 10.1016/j.is.2010.09.001.

[12] W. M. P. van der Aalst, and A. J. M. M. Weijters, “Process mining: a research agenda”, Computers in Industry, Volume 53, Issue 3, Process / Workflow Mining, April 2004, Pages 231-244, ISSN 0166-3615, DOI: 10.1016/j.compind.2003.10.001.

[13] W. M. P. van der Aalst, B. F. van Dongen, J. Herbst, L. Maruster, G. Schimm, and A. J. M. M. Weijters, “Workflow mining: A survey of issues and approaches”, Data & Knowledge Engineering, Volume 47, Issue 2, November 2003, Pages 237-267, ISSN 0169-023X, DOI: 10.1016/S0169-023X(03)00066-1.

[14] W. M. P. van der Aalst, “Process Mining”, Berlin, Heidelberg 2011, ISBN 978-3-642-19344-6

Pospisil, M., Hruška, T., “Business Process Simulation for Predictions” In: BUSTECH 2012 : The Second International Conference on Business Intelligence and Technology, Nice, FR, IARIA, 2012, s. 14-18, ISBN 978-1-61208-223-3