Detecting Bottleneck and Fraud in Agile Development by using Petri net Performance and Trace Clustering

Muhammad Ar Razi
Informatics
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
rezaarrazi@gmail.com

Kelly Rossa Sungkono
Informatics
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
kelly@its.ac.id

Riyanarto Sarno
Informatics
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
riyanarto@if.its.ac.id

Cahyaningtyas Sekar Wahyuni
Informatics
Institut Teknologi Sepuluh Nopember
Surabaya, Indonesia
cahyaningtyas.18051@mhs.its.ac.id

***Abstract*— Software Development Life Cycle model is one of the basic idea of the software development. SDLC is a continuous process, the cycle starts from the moment when it is decide to release the project, and it ends when its full remove from the exploitation. Agile is one of the mostly used SDLC models. Combination of incremental and iterative process models with a focus on process customer satisfaction and can adapt to rapid delivery of working sofware is called Agile SDLC. This contribution of this paper is evaluating the performance, such as occurring bottleneck and fraud, of an Agile model to handle the management of the project. This paper uses Petri net to analyze Agile SDLC model performance. The petri net use to detect the bottleneck that occur in the model. If an event has longer waiting time tahen others, it can identify that the bottleneck occur in that task. Then, the fraud is detected based on trace clustering. Based on the experiment, the cluster which has lowest member is indicates as fraud processes.**

Keywords—SDLC; Agile; bottleneck; alpha algorithm; Petri net

I. INTRODUCTION

Software Development Life Cycle model is one of the basic idea of the software development. SDLC is a continuous process, the cycle starts from the moment when it's decide to release the project, and it ends when its full remove from the exploitation.

We create an Agile model using ProcessMaker. The event log results from the Agile model are extracted to evaluate the Agile model performance. The event log informs the event that occurred [1]–[3]. the event log is obtained from ProcessMaker. event logs are stored in the database belonging to the ProcessMaker and can be exported for further use. Because Prom can only read event logs in MXML format, it is necessary to do a conversion for data exported from ProcessMaker using Disco.

From Disco we get the correct format of the event log. Then, Prom is used to mining the process. ProM is an open source framework for process mining. ProM has a variety of process mining technique in the form of plug-ins. Prom can be defined as a framework that can be expanded to support many technical plugins. Some of the algorithms contained in the Prom are Alpha.

The basic algorithm used for mining processes is Alpha. By using the Alpha algorithm, the activity of a case can be analyzed and produce a process pattern.

II. LITERATURE REVIEW

A. Agile Model

Combination of incremental and iterative process models with a focus on process customer satisfaction and can adapt to rapid delivery of working sofware is called Agile SDLC [4]. It breaks the development process into smaller incremental processes. These processes are provided in iterative manner. The duration of iteration is about one to three weeks. Every iteration involves Planning, Requirements Analysis, Design, Coding, Unit Testing, and Acceptance Testing.

B. ProcessMaker

ProcessMaker is a process management and workflow software. This software is easy to use. It can automate, design and deploy many business processes. It also record the event log of the process, and can be exported into CSV file.

C. Disco

Disco is software for process mining. Powerful, easy to use, and lightning-fast. Disco has the fast process mining algorithm, and the most efficient log management and filtering framework.

D. Event Logs

Historical records of a system of information or tools stored in the event log. In the information system, evidence of ongoing transactions is stored in the event log. Activities are recorded sequentially and parallel in activity records, such as tasks performed simultaneously with different resources. Every event log will be saved at any level.

Information about a process are provide on the event logs. Events that occur are described in the event log [3], [5], [6]. Establishment of an actual business model will be easier if by utilizing the event log generated in an information system.

978-1-7281-3832-9/19/$31.00 ©2019 IEEE

E. ProM

ProM is open source framework for process mining algorithms. This is platform that easy-to-use, very stable and easily to develop. This platform also support an optimal process mining that can be developed. ProM [7]–[9] has many kinds of plug-ins that can be used to the latest developments in research of process mining with our own data. ProM framework only can receive the input log with format file XES or MXML format. The interface can be seen in Figure 1.

Figure 1 ProM Interface

F. Process Mining

Combination of technical knowledge from data mining, process analysis, and computational intelligence is called process mining. The process mining process uses an event that recorded by information system [8], [10], [11]. The mining process is used for various systems. The event logs used in process mining to perform the analysis of the real process. Extracting knowledge from the event log can improve the real process [12]–[15].

G. Alpha Algorithm

The common used algorithm for mining processes is Alpha. By using the Alpha algorithm, the activity of a case can be analyzed and produce a process pattern.

This algorithm has many benefit, such as it can construct casualty from a series of sequeances event. This method can mine the perspective of the flow of the process with respect to Petri Nets [2], [16], [17]. The Alpha algorithm was first introduced by Professor Wil van der Aalst from Technishe Universiteit Eindhoven in The Netherlans.

III. PROPOSED METHOD

A. Agile Model

Agile models are created using ProcessMaker. Then simulate the model. The event log results from the Agile model are extracted to evaluate the Agile model performance. Extraction results in CSV format files. TABLE I describe the data extracted in CSV file format:

1) *CaseId : Unique number to identify every case.*
2) *Activity: an event/task that has been done.*
3) *Timestamp: the time of an activityis done.*
4) *Resource: The person/role who did the activity.*

TABLE I. CASE ACTIVITY

CaseId	Activity	Timestamp	Resource
1	User Story Upload	3/25/2019 0:40	client
1	Define requirement	3/25/2019 0:41	product
1	Prioritization	3/25/2019 0:41	product
1	Requirements Approval	3/25/2019 0:42	client
1	CDM and PDM Upload	3/25/2019 0:53	system
1	CDM and PDM Approval	3/25/2019 0:54	product
1	CDM and PDM Upload	3/25/2019 0:56	system
1	CDM and PDM Approval	3/25/2019 0:57	product
1	UML Diagrams Upload	3/25/2019 0:59	system
1	UML Diagrams Approval	3/25/2019 1:01	product
1	Software Design Document Upload	3/25/2019 1:07	system
1	Customer Approval of Software Design	3/25/2019 1:10	client
1	Budget Proposal Upload	3/25/2019 1:11	product
1	Budget Aproval	3/25/2019 1:11	client
1	Budget Proposal Upload	3/25/2019 1:12	product
1	Budget Aproval	3/25/2019 1:13	client
1	Team Formation	3/25/2019 1:13	product
1	Team Approval of Software Design	3/25/2019 1:23	developer
1	CDM and PDM Upload	3/25/2019 1:24	system
1	CDM and PDM Approval	3/25/2019 1:25	product
1	UML Diagrams Upload	3/25/2019 1:25	system
1	UML Diagrams Approval	3/25/2019 1:27	product
1	Software Design Document Upload	3/25/2019 1:28	system
1	Customer Approval of Software Design	3/25/2019 1:29	client
1	Budget Proposal Upload	3/25/2019 1:30	product
1	Budget Aproval	3/25/2019 1:31	client
1	Team Formation	3/25/2019 1:32	product
1	Team Approval of Software Design	3/25/2019 1:36	developer
1	Job Assignment	3/25/2019 1:37	product
1	Development Process	3/25/2019 1:40	developer

B. Data Convertion

Event logs are stored in the database belonging to the ProcessMaker and can be exported for further use. Because Prom can only read event logs in MXML format, it is necessary to do a conversion for data exported from ProcessMaker using Disco. To do this, you only need to input event log data in CSV format on Disco. Then export in MXML format.

C. Models with alpha algorithms

After getting the correct format, then import the event log to ProM. After that, model based on data by using algorithm alpha in ProM is created. The models are shown in Figure 4. Those models show that the flow of SDLC goes to the final stage.

IV. RESULT AND ANALYSIS

A. Bottleneck identification

To identification stages of the bottleneck are describes on this section. Figure 4 and Figure 5 shows an overview of agile models based on alpha algorithms along with bottleneck locations. The image shows a task that has a long average waiting time, has a magenta color and makes it a bottleneck. For tasks that have a fairly fast waiting time, they have a yellow color, while blue shows the task has a fast waiting time. The bottleneck is shown only on magenta colored nodes. The time difference required between tasks connected to other tasks is calculated to determine whether there is a bottleneck or not. Then, the average waiting time between tasks will be compared. Picture. Figure 2 shows the execution time through the process. The results of the calculation and output of ProM in determining the bottleneck are calculated based on the token in place. The algorithm follows to determine the bottleneck that occurs that is related to the exact or not algorithm. The final step is to calculate the average waiting time in each place.

Long waiting times occur because of the decision-making process by different resources. This happens when going to the next stage and requires validation. Because of this process, bottlenecks occur between decision making processes. But besides that, the stage that has the longest waiting time is the development stage. Details of this stage are shown in the Figure 6.

	Throughput time (seconds)
avg	4022100.0
min	3978720.0
max	4065480.0
stdev	61348.58
fast 25...	3978720.0
slow 2...	4065480.0
norma...	0.0

Figure 2 Throughput time

B. Fraud Identification

After carrying out an analysis bottleneck using an algorithm, fraud analysis of every activity in a business process is carried out using the Trace Clustering algorithm. Figure 3 shows the Comparison matrix of traces that have been clustered. Fraud is identified in clustering that has few members.

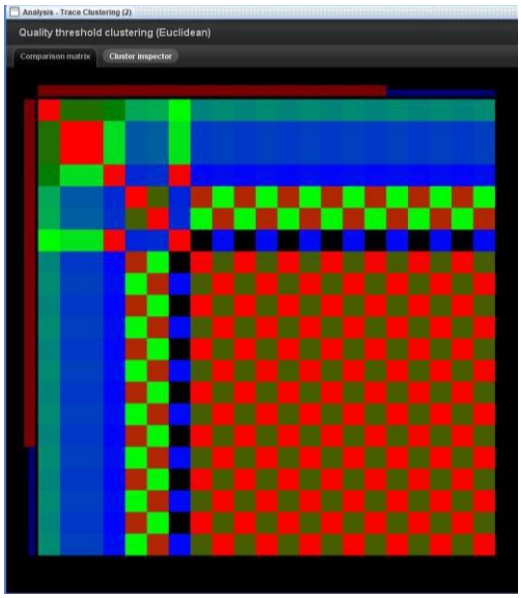

Figure 3 Comparison Matrix

The results of clustering produce 2 clusters. One cluster has 16 traces and the other has 5 traces. Clusters with 16 traces have a complete process from beginning to end. Whereas clusters with 5 traces have a short and incomplete process. This is shown in Figure 7 and Figure 8.

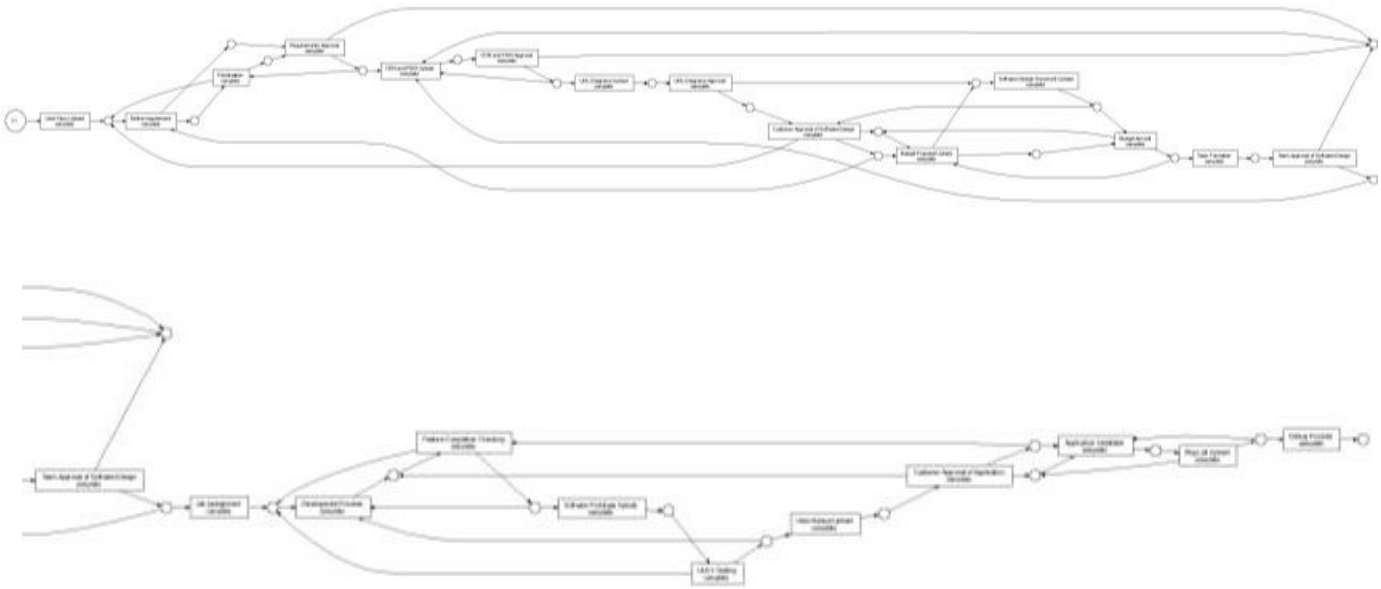

Figure 4 Model by Alpha algorithm

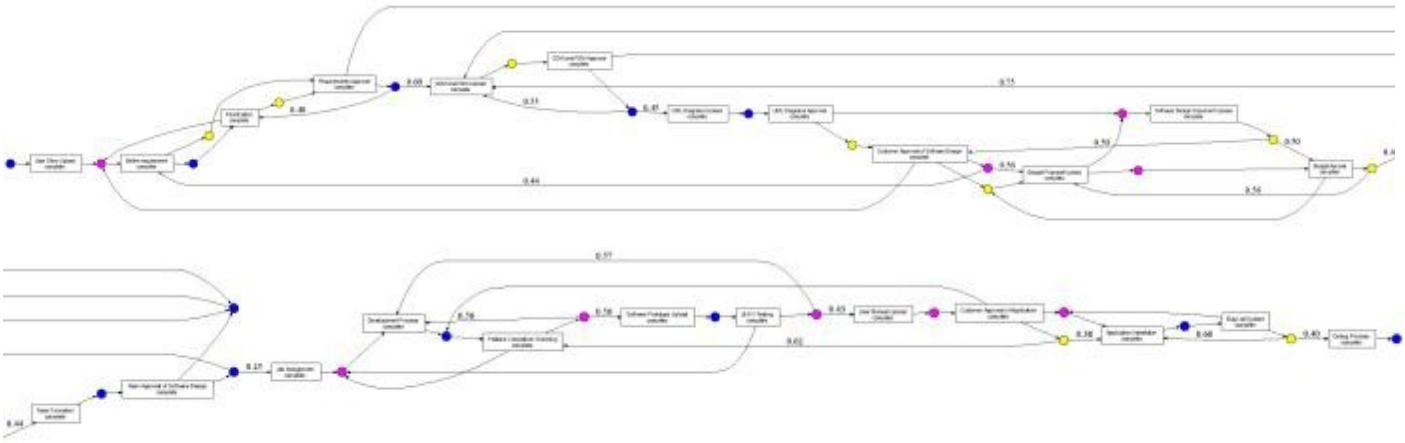

Figure 5 Bottleneck with alpha algorithm

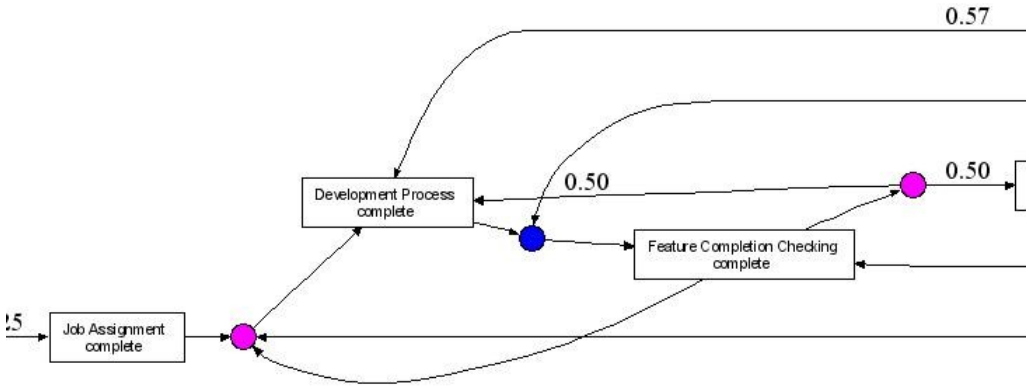

Figure 6 Bottleneck on Development Proces

Figure 7 Cluster with complete process

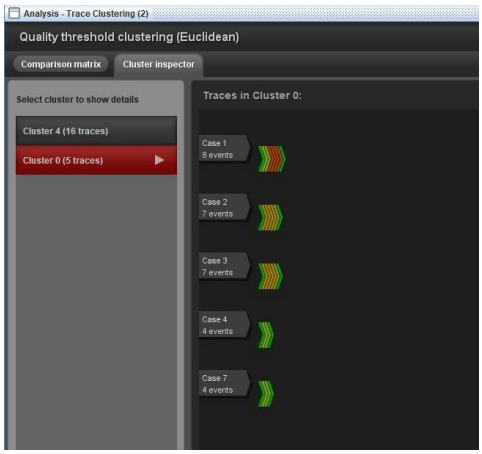

Figure 8 Cluster with fraud

V. CONCLUSION

This study identifies bottlenecks and fraud in agile development. The method used to identify bottlenecks is Petri net Performance analysis and alpha algorithms to mine the process. Whereas to identify fraud using the Trace clustering method. To identify bottlenecks and fraud, both are done with Prom. Based on the evaluation, there are two conclusions that can be derived.

The first conclusion is that in agile development there are several bottlenecks. Long waiting times occur because of the decision-making process by different resources. This happens when going to the next stage and requires validation. Because of this process, bottlenecks occur between decision making processes. But besides that, the stage that has the longest waiting time is the development stage.

The second conclusion is to identify fraud can be done by clustering the trace generated from the process. In this research, results of clustering produce 2 clusters. One cluster has 16 traces and the other has 5 traces. Clusters with 16 traces have a complete process from beginning to end, whereas clusters with 5 traces have a short and incomplete process.

VI. CONCLUSION

[1] K. R. Sungkono, R. Sarno, and N. F. Ariyani, "Refining business process ontology model with invisible prime tasks using SWRL rules," in *2017 11th International Conference on Information & Communication Technology and System (ICTS)*, 2017, pp. 215–220. http://doi.org/10.1109/ICTS.2017.8265673.

[2] N. Lohmann, E. Verbeek, and R. Dijkman, "Petri Net Transformations for Business Processes -- A Survey," in *Transactions on Petri Nets and Other Models of Concurrency II: Special Issue on Concurrency in Process-Aware Information Systems*, K. Jensen and W. M. P. van der Aalst, Eds. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009, pp. 46–63. http://doi.org/10.1007/978-3-642-00899-3_3.

[3] M. Von Rosing, S. A. White, F. Cummins, and H. De Man, "Business process model and notation-BPMN," in *The Complete Business Process Handbook: Body of Knowledge from Process Modeling to BPM*, vol. 1, 2014, pp. 429–453. http://doi.org/10.1016/B978-0-12-799959-3.00021-5.

[4] S. Sharma, D. Sarkar, and D. Gupta, "Agile Processes and Methodologies: A Conceptual Study," *International Journal on Computer Science and Engineering*, vol. 4, no. 5, 2012.

[5] S. De Cnudde, J. Claes, and G. Poels, "Improving the quality of the Heuristics Miner in ProM 6.2," *Expert Systems with Applications*, vol. 41, no. 17, pp. 7678–7690, 2014. http://doi.org/10.1016/j.eswa.2014.05.055.

[6] T. Savickas and O. Vasilecas, "Belief network discovery from event logs for business process analysis," *Computers in Industry*, vol. 100, pp. 258–266, 2018. http://doi.org/10.1016/j.compind.2018.04.020.

[7] W. Bose, R P Jagadeesh Chandra, Van der Aalst, "Filtering out Infrequent Behavior from Business Process Event Logs," *Business Process Management Workshops (1) (Vol. 99, pp. 165-166).*, pp. 165–166, 2011. http://doi.org/10.4121/uuid.

[8] A. Burattin, A. Sperduti, and W. M. P. van der Aalst, "Heuristics Miners for Streaming Event Data," *ArXiv CoRR*, 2012. http://doi.org/10.1109/CEC.2014.6900341.

[9] L. Hakim, R. Sarno, and K. R. Sungkono, "Modified Alpha++ Algorithm for Discovering the Hybrid of Non-Free Choice and Invisible Task of Business Processes," *International Journal of Intelligent Engineering and Systems (IJIES)*, vol. 12, no. 3, pp. 31–40, 2019.

[10] R. Sarno and K. R. Sungkono, "Coupled Hidden Markov Model for Process Mining of Invisible Prime Tasks," *International Review on Computers and Software (IRECOS)*, vol. 11, no. 6, pp. 539–547, 2016. http://doi.org/10.15866/irecos.v11i6.9555.

[11] S. Park and Y. S. Kang, "A Study of Process Mining-based Business Process Innovation," *Procedia Computer Science*, vol. 91, no. Itqm, pp. 734–743, 2016. http://doi.org/10.1016/j.procs.2016.07.066.

[12] K. R. Sungkono and R. Sarno, "Constructing Control-Flow Patterns Containing Invisible Task and Non-Free Choice Based on Declarative Model," *International Journal of Innovative Computing, Information and Control (IJICIC)*, vol. 14, no. 4, 2018.

[13] W. M. P. Van Der Aalst and A. H. M. Hofstede, "YAWL : yet another workflow language," vol. 30, pp. 245–275, 2005. http://doi.org/10.1016/j.is.2004.02.002.

[14] R. Sarno and K. R. Sungkono, "Hidden Markov Model for Process Mining of Parallel Business Processes," *International Review on Computers and Software (IRECOS)*, vol. 11, no. 4, pp. 290–300, 2016. http://doi.org/10.15866/irecos.v11i4.8700.

[15] R. Sarno, K. R. Sungkono, R. Johanes, and D. Sunaryono, "Graph-Based Algorithms for Discovering A Process Model Containing Invisible Tasks," *Intelligent Networks and Systems Society (INASS)*, vol. 12, no. 2, pp. 85–94, 2019.

[16] L. Wells, "Performance analysis using CPN tools," *Proceedings of the 1st international conference on Performance evaluation methodolgies and tools - valuetools '06*, p. 59, 2006. http://doi.org/10.1145/1190095.1190171.

[17] W. M. P. der Aalst, "Decomposing Petri nets for process mining: A generic approach," *Distributed and Parallel Databases*, vol. 31, no. 4, pp. 471–507, 2013.