Business State Model for Knowledge Work Process Mining

Takashi Kobayashi

School of Network and Information, Senshu University
t.koba@isc.senshu-u.ac.jp

Abstract. It is generally known that knowledge work processes include a lot of exception handling, and are non-repeated and unpredictable. Therefore, in case of applying process mining techniques to these processes, critical issues are generating stable business logic models and defining behavior metrics to assess exception handling. In order to resolve these issues, this paper proposes a business state model consisting of following two structures. The essential structure is constructed by abstracting only stable core events and by configuring a state transition model with them. The incidental structure is constructed by embedding the other changeable events into each state of this model and by defining behavior metrics with event attributes and their context information.

The proposed model was implemented as a software tool and confirmed by applying to the real consumer loan approval process in a financial institute.

Keywords. process model, process mining, knowledge work, business state model, language/action perspective, business logic, behavior metrics.

1 Introduction

Recently, it is a critical issue to improve knowledge work processes. Most of knowledge works have only the minimum rules or constraints and a large part of the procedure is left to workers' discretion. Therefore, knowledge work processes are generally non-repeated and unpredictable [1].

Process mining is a discipline that generates a process model, finds problems, and solves them based on event logs [2][3]. This discipline has possibilities to increase the effects and efficiencies of process improvement. One of significant themes in process mining is process discovery. This technique is to find event occurrence patterns such as serial, parallel, and synchronization, and to generate a consistent process model to replay every pattern on the model. A lot of process discovery algorithms have been developed. However, most of algorithms focus on routine works and not suitable for processing a lot of exception handling which exist in knowledge work processes.

Issues in the improvement of knowledge work processes are as follows.

i) Generating stable business logic models for knowledge work processes: In business process improvement, it is important to prepare a process model that every stakeholder shares and refers to. This model should not include organization-specific operation rules, but denote only a logical procedure for the target business. We call such process models "business logic models." In routine work processes, we can easily obtain business logic models based on past record data.

However, in knowledge work processes, a lot of exception handling makes it difficult to extract business logic models from execution results.

ii) Defining behavior metrics to assess exception handling: In routine work processes, problems are obtained as differences between a business logic model and real event behavior. However, there is a lot of exception handling in knowledge work processes and they do not always cause problems. It is one of powerful methods to define quantitative indexes characterizing various kinds of behavior of exception handling and to analyze their relationship with problems. We call this quantitative index “behavior metrics.”

This paper aims to propose a method to solve above two issues. Chapter 2 introduces conventional technologies and their problems, and Chapter 3 introduces LAP model – the key methodology for solving the issues. Chapter 4 presents a business state model to provide the solution based on LAP, and Chapter 5 assesses the effects of the proposed model with a real case.

2 Conventional Approach and Problems

2.1 Business logic model

A lot of business logic models are developed for routine work processes in the domain of SCM (Supply Chain Management) and ERP (Enterprise Resource Program) [5]. However, models for knowledge work processes are few. The most representative model is LAP (Language Action Perspective) [6][7], which was developed in the domain of CSCW (Computer Supported Cooperative Work). This paper applies the LAP model to generate an essential structure of the target business process. Different approach is an artifact-oriented approach [4]. In this approach, we focus on he main artifacts of the target business process and clarify the logical sequence to output them.

2.2 Behavior metrics

Traditionally, the behavior of a process model is compared to event logs and the degree of match between them is defined as fitness [2][3]. The fitness is the metrics of difference between ideal (process model) and reality (event logs). This approach is effective in either of following cases.

a. When a high-quality model for the target process is obtained, the behavior of the process can be assessed with the fitness.
b. When a typical and reliable event logs are obtained, the quality of process models generated from the event logs can be assessed with the fitness.

However, when neither process models nor event logs are reliable, it is difficult to assess the behavior of processes or the quality of models. In this paper, we do not model the behavior of exception handling as the difference between a process models and event logs, but as an incidental structure configuring a process model.

2.3 Process model stability

In generating a process model from event logs, it is important to balance between overfitting and underfitting [2][3]. Overfitting is the property that the model is too specific and only allows for the accidental behavior. On the other hand, underfiting is the property that the model is too general and allows for behavior unrelated to the behavior observed. A two-step approach is a new method of using a transition system and the theory of region in order to achieve this challenge [3]. With this method, the user can decompose the parts to be analyzed strictly and synthesize the parts to be analyzed roughly by setting parameters related to events and states. In this paper, we take a similar approach: to flexibly change the model detail level as necessary.

3 Language Action Perspective State Model

3.1 Workflow state machine

Language Action Perspective (LAP) is the methodology of system design proposed in 1987 and based on the model of conversation for action, which was developed combining the speech act theory and the system theory. LAP has been applied to the process improvement of knowledge works and the system design of human computer interaction [6][7]. We define LAP state model based on the state machine diagram of UML [8][9] in order to apply the LAP methodology to developing a business state model, which is introduced in Chapter 4.

Goals of every work are related to providing some products or services to a customer with appropriate conditions. Conditions are generally related to time and costs. It is necessary to make commitment between a customer and a performer for achieving a work successfully. The work is achieved according to four states: Preparation, Negotiation, Performance, and Acceptance as described in Fig. 1. State transitions occur in each state triggered by six acts (events): request, counteroffer, agree, report completion, declare satisfaction, and decline to acceptance according to predefined condition of act transition as described in Fig. 1. In each state, kinds of activities (tasks) are executed to make acts occur. We call this state transition model a "workflow loop." Business process is the series of activities executed according to the workflow loop. This workflow loop model can be applied to every business process.

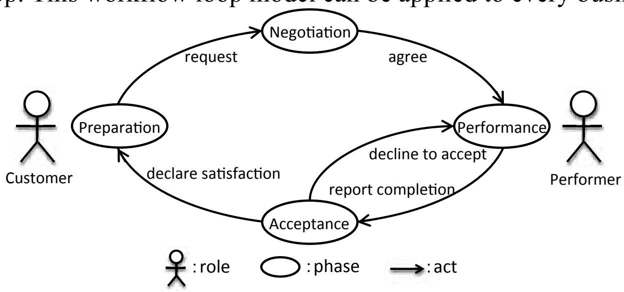

Fig. 1. Workflow loop model

The workflow loop model can be formalized as "workflow state machine (*W-SM*)" as follows:

$W = (A, S, T, R, G)$

- $A = (a_{req}, a_{cou}, a_{agr}, a_{com}, a_{sat}, a_{dec})$ is a 6-tuplet of events occurring corresponding to acts: *request, counteroffer, agree, report complete, declare satisfaction, and decline acceptance.*
- $S = (s_{Prep}, s_{Nego}, s_{Perf}, s_{Acce})$ is a 4-tuplet of states corresponding to phases: *Preparation, Negotiation, Performance, and Acceptance.*
- T is a set of act transitions tr, where $tr = \{(s_{before}, a_{trigger}, s_{after}) \mid s_{before} \in S$ *is a state before transition,* $a_{trigger} \in A$ *is an event triggering transition,* $s_{after} \in S$ *is a state after transition, and the transition model is as described in Fig. 1\} is a 3-tuplet of states and an event.*
- $R = (r_{Cust}, r_{Perf})$ is a duplet of actors corresponding to roles: *Customer* and *Performer*.
- G is a goal of the work of providing some products or services to a customer with appropriate conditions generally related to time and costs.

We call events to trigger act transitions "act events," and events to visit a state $s \in S$ "input events" of s, and events to leave a state $s \in S$ "output events" of s.

3.2 Map of workflow state machines

In a real business situation, a business process is not simple enough to be executed by only one customer and one performer. Therefore, the process is decomposed into simple activities and each activity is entrusted to other person. As a consequence, the business process becomes a hierarchical network model, in which *W-SM*s are combined recursively, as described in Fig. 2. These *W-SM*s also have a customer and a performer, and are executed according to the workflow loop model. In this figure, a *W-SM* at the top indicates a main process. We call it a "primary state machine (*P-SM*)." The customer and the performer of a *P-SM* are the original customer and the original performer. *W-SM*s placed in each state of the *P-SM* indicates activities entrusted to other persons. We call them "secondary state machines (*S-SM*s)."

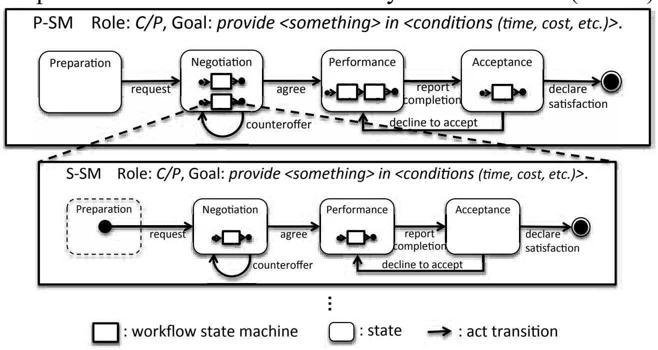

Fig. 2. Map of workflow state machines

When a *S-SM* is again decomposed into smaller *S-SM*s and entrusted to other persons, the original *S-SM* has these smaller *S-SM*s in its each state and is called an "upper *S-SM*", which indicates a sub-process. Generally, when a *W-SM* has smaller *W-SM*s in its each state, we call the former *W-SM* a "parent state machine," and the latter smaller *W-SM*s “child state machines.” When a *W-SM* having no child state machine, we call it a “*lowest S-SM*”, which indicates a task. When multiple child state machines are placed in the same state of a parent state machine, the execution order among child state machines are defined with “link transitions.”

Hierarchical placement of *W-SM*s is formalized as a “map of workflow state machines” as follows:

$M = (W, P, L)$

- W is a set of workflow state machines.
- P is a set of workflow state machine placements ps, where ps = *{(*w_{child}*,* w_{parent}*,* s_{parent}*) |* w_{child} *and* $w_{parent} \in W$ *are a child state machine and its parent state machine respectively,* $s_{parent} \in S$ *is a state of* w_{parent}*, and* w_{child} *is included in* s_{parent}*} is a 3-tuplet.*
- L is a set of link transitions lk, where lk = *{ (*$\mathbf{l}_{src}$*,* $\mathbf{l}_{dst}$*,* λ*, cd) |* $\mathbf{l}_{src}$ *is a set of source events of each link transition,* $\mathbf{l}_{dst}$ *is a set of destination events of each link transition,* λ *is one of link transition types: serial, AND-split, OR-split, AND-join or OR-join as described in Fig. 3, and cd is a condition of the link transition} is a 4-tuplet.*

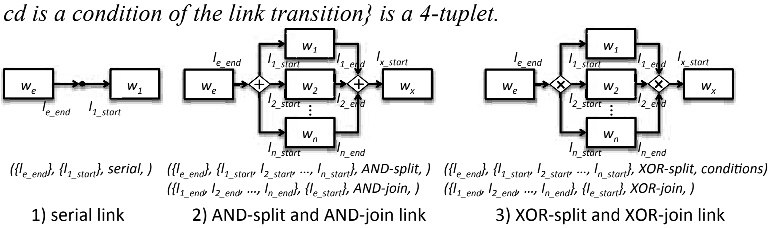

Fig. 3. Types of link transition

We call events to trigger link transitions, i.e. $\mathbf{l}_{src}$ *and* $\mathbf{l}_{dst}$ “link events.”

4 Business State Model

This chapter proposes a business state model applying the LAP model in order to achieve two challenges described in Chapter 1.

The minimum rule obeyed by knowledge workers is to provide products or services with conditions agreed with customers. Therefore, the important events to configure a business logic model are related to the coordination between a customer and a performer, namely, six act transitions of the LAP model: request, counteroffer, agree, report completion, declare satisfaction, and decline to accept. Between act transitions, knowledge workers can execute their work with their own style. This is the same way in the coordination between individuals and across organizations. Therefore, in the business state model, the target business process is modeled as hierarchical structure of workflow state machines and appropriate events in event logs are assigned to each act of every workflow state machine. We call this model an essential structure of the business state model.

In accessing exception handling, it is easier to analyze the results occurred by exception handling than to track the sequence patterns of exception handling. The amount of information, which can be used in defining behavior metrics depends on attributes included in each event. Unfortunately, events usually include a few attributes such as activity, resource and timestamp. Therefore, by coupling these event attributes with context information (in which W-SM and which state is each event observed?), varieties of behavior metrics can be acquired. We call this model an incidental structure of the business state model.

With these two structures, stable business logic models for knowledge work processes and behavior metrics to assess exception handling can be obtained. Moreover, the business state model provides the verification and adjustment function, with which users can verify the validation of the model referring to event logs and if necessary adjust the model.

Section 4.1 proposes the essential structure, Section 4.2 proposes the method of verifying and adjusting the model, and Section 4.3 proposes the incidental structure.

4.1 Essential structure

The essential structure of the business state model is developed to decide elements of each workflow state machine $W = (A, S, T, R, G)$ and the map of workflow state machines $M = (W, P, L)$, which are already described in Chapter 3.

4.1.1 Method to assign act events

As described in the previous chapter, act events have business specific intentions: request, counteroffer, agree, report completion, declare satisfaction, and decline to acceptance. Therefore, the act events are related to the business logic for understanding event logs from the viewpoint of business goals. For example, in the typical retail process, the period from request (inquiry) to agree (contract) is the term of sales activities, and it is a crucial issue to win a customer's trust. On the contrary, the period from agree (contract) to report complete (deliver) is the term of logistics activities, and it is a fundamental issue to increase efficiency.

In order to assign an appropriate event in event logs to each act transition of every *W-SM*, we select the optimal event that has the same intention as each act transition. If each act event of every *W-SM* is assigned, states and transitions of the *W-SM* are uniquely defined. The method to assign act events is described as follows.

(1) Recognizing workflow state machines

Based on specifications of the target business process, *W-SM*s and their goals and roles can be recognized. The specifications of a *P-SM* (main process) can usually be obtained from users as the fundamental specifications of the target process. If the target process is related to consumer services, the customer is a consumer and the performer is a corporation itself. If the target process is related to corporate internal services, the customer is an employee of the corporation and the performer is an administration department. The goal of a *P-SM* is related to specifications of products and services to be delivered to the customer and conditions about time and costs. The goal and role of *lowest S-SM*s (tasks) can be obtained as an activity name and actors of the target process. As an *upper S-SM* (sub-process) coordinates multiple *S-SM*s that have relational goals, its role and goal are the one to integrate *S-SM*s' roles and goals.

(2) Recognizing intentions

Each intention of act transitions configures the goal that *a performer provides something to a customer in some conditions*. Table 1 demonstrates specifications of each intention of act transitions: an actor to take action, a message to communicate, and a medium for the message. The intentions of each *W-SM* can be obtained by applying its role and goal to the specifications of intentions.

Table 1. Specifications of each intention of act transitions

Role: ***Customer/Performer***, Goal: ***provide <something> in <conditions (time, cost, etc.)>.***			
Act transition	Actor	Message	Medium
request	*C*	request <smt X_0> with <cinditions Y_0>.	application, submission, inquiry
counteroffer	*P*	counteroffer <smt X_1> with <cinditions Y_1>.	offer, estimate, substitute
agree	*C & P*	agree to <smt X_1> and <cinditions Y_1>.	order, contract, acceptance
report completion	*P*	provide <smt X_2> in <cinditions Y_2>.	result, product, delivery, approval
declare satisfaction	*C*	satisfied with <smt X_2> and <cinditions Y_2>.	payment, acknowledgement, thanks
decline to accept	*C*	dissatisfied with <smt X_2> or <cinditions Y_2>.	return, complaint, claim

(3) Matching intentions

An appropriate event in event logs is assigned to every act transition of *W-SM*s recognized in (1), by matching its intention recognized in (2) with attributes of the event. Generally, events can be described by 6-tuplet $E = (CI, EI, TS, AN, RN, LC)$, where *CI* is a set of *case ID*s, *EI* is a set of *event ID*s, *TS* is a set of *timestamp*s, *AN* is a set of *activity name*s, *RN* is a set of *resource name*s, and *LC* is a set of *activity lifecycle*s [2]. The elements of *E* is called "event attributes" and the value of event attribute *n* is described by $\#_n(e)$, e.g. the resource associated to event *e* is described by $\#_{resource}(e)$. Cases have also some attributes and the value of case attribute *n* is described by $\#_n(c)$. Some of these attributes are useful to match act transitions with events as described below.

- Activity name indicates what task or state the event is related to. If any event has the activity name related to the message or medium of any act transition, the event is suitable for the act transition.
- Resource name indicates who makes the event occur. If any event has the resource name related to the actor of any act transition, the event is suitable for the act transition.
- Case ID, event ID, and timestamp indicate in what order the event occurred. There is an order relationship among act transitions of each *W-SM* as described in Fig. 1. Therefore, we can select events to be consistent with this relationship.

4.1.2 Method to map workflow machines and assign link events

Link events are related to the execution control of child state machines that are decomposed from a parent state machine and entrusted to other persons. In order to assign an appropriate event to each link transition, we need clarify the relationship of decomposing and entrusting among *W-SM*s as described below.

(1) Placing workflow state machines

The business process is, as described in Chapter 3, modeled as a hierarchical network structure in which a *P-SM* is placed on the highest layer and *S-SM*s are placed in each state of the *P-SM* or *upper S-SM*s. Therefore, we decide the placements of *W-SM*s according to the following procedure. First, we place *S-SM*s in appropriate states of a *P-SM* comparing the goal of each *S-SM* and the message included in the intention of the *P-SM*. For example, we place *S-SM*s having the goal related to the message of fulfilling the work in the Performance state of the *P-SM*. Next, we attempt to add an *upper S-SM* when multiple *S-SM*s are placed in the same state. These *S-SM*s may configure the *upper S-SM* (sub-process) having a common goal. Finally, we place *S-SM*s in appropriate states of the *upper S-SM* in the same way as in the case of a *P-SM*.

(2) Ordering workflow state machines

When multiple *S-SM*s are in the same state of the *P-PM* or *upper S-SM*, we can order them using three types of link transitions as described in Fig. 3. More complicated control flows of *W-SM*s can be described by combining these three transition types. However, the control flows defined with the link events do not indicate business logic models but operation rules of individual organizations. Therefore, assigning link events should be restricted to a minimum.

(3) Clarifying embedded events

We can configure a framework of the essential structure of the business state model by using above procedures: (1) and (2). However, not all the *S-SM*s are linked with each other, so a lot of events remain unordered. These S-SMs are placed in some state of their parent state machines but their execution orders in the state are not specified. We call these unordered events "embedded events," which means the events embedded in the state. The behavior of the embedded events indicates deviations from the business logic and is key information for process improvement. They are modeled as an incidental structure of the business state model in Section 4.3.

4.2 Verification and adjustment

The essential structure of the business state model is generated by assigning act events and link events and by placing embedded events as described in Section 4.1. In order to make the essential structure an effective model for the process improvement, we had to verify this model by using event logs as described below.

(1) Verifying act events

The results of assigning act events can be verified by comparing the behavior of event logs with the predetermined sequence patterns. We define Suitability of act events (*SoA*) to access the matching degree between the behavior of event logs and the act sequence patterns. *SoA* shows the ratio of number of event sequences matching act sequence patterns of a *W-SM* in total number of event sequences of the *W-SM*.

$SoA = \sum_w NA_w / \sum_w NAA_w$

- NA_w is number of event sequences matching act sequence patterns of a *W-SM* $w \in W$.
- NAA_w is total number of event sequences of the *W-SM* $w \in W$.

(2) Verifying embedded events

The results of locating embedded events can be verified by checking whether embedded events placed in each state truly occur only in the state or not. We define Suitability of embedded events (*SoE*) to access the matching degree between the behavior of event logs and the placement of embedded events. *SoE* shows the ratio of number of events of a *W-SM* occurring in a placed parent state in total number of events of the *W-SM*.

$SoE = \sum_w NE_w / \sum_w NEE_w$

- NE_w is number of events of a *W-SM* $w \in W$ occurring in a placed parent state.
- NEE_w is total number of events of the *W-SM* $w \in W$.

(3) Verifying link events

The results of assigning link events can be verified by comparing the behavior of event logs with the predetermined sequence patterns according each link transition type: serial, parallel, and conditional. We define Suitability of link events (*SoL*) to access the matching degree between the behavior of event logs and the link sequence patterns. *SoL* shows the ratio of number of event sequences matching link sequence patterns of a link transition in total number of event sequences of the link transition.

$SoL = \sum_l NL_l / \sum_l NLL_l$

- NL_l is number of event sequences matching sequence patterns of a link transition $l \in L$.
- NLL_l is total number of event sequences of the link transition $l \in L$.

4.3 Incidental structure

The incidental structure of the business state model is developed to define behavior metrics based on the essential structure, especially the embedded events. This model is the source where we can find many kinds of problems, critical causes, and excellent solutions. Because it is difficult to find sequence patterns of embedded events, we adopt following quantitative values to indicate the results of exception handling.

(1) Performance metrics

Elapsed time measured in various situation of the business process is fundamental metrics for finding problems and analyzing their causes. We can easily calculate elapsed time by taking the difference of timestamps of any two events. Because the essential structure is a hierarchy of *W-SM*s, we can drill down from main process to sub-processes and tasks using performance metrics.

(2) Frequency metrics

Repetition of specific events is also fundamental metrics to indicate significant problem occurrences such as return back in performance or difficulty in negotiation. This metrics is related to the trust that customers have for their performers and the deterioration of frequency metrics decreases customer satisfaction.

(3) Behavior metrics

Many kinds of behavior metrics can be defined according to the target business process, combining the performance metrics and the frequency metrics with event attributes (activity, resource, timestamp, etc.). Moreover, behavior metrics becomes much powerful when coupled with context information (in which W-SM and which state is each event observed?) described in Table 2.

Table 2. Behavior metrics

Attributes	Performance Metrics		Frequency Metrics	
	P-SM (process)	S-SM (task)	P-SM (process)	S-SM (task)
Context	State lead-time State cycle-time State work in progress	Task executing time Task waiting time	Process return back frequency Process negotiation failure frequency	Task return back frequency Task negotiation failure frequency
Activity	Task processing time		Task repetition frequency	Event repetition frequency
Resource	Resource operating time Resource availability		Resource use frequency	

5 Evaluation

We developed a software tool to define and verify a business state model and to analyze event logs with it using MS-Excel and VBA. We show that our approach can achieve two challenges described in Chapter 1 by following case study.

This case study is related to the consumer loan approval process in a Dutch financial institute [10]. A loan application is submitted through a webpage. Then, some automatic checks are performed, after which the application is complemented with additional information. This information is obtained trough contacting the client by phone. If an applicant is eligible, an offer is sent to the client by mail. After this offer is received back, it is assessed. When it is incomplete, missing information is added by again contacting the client. Then a final assessment is done, after which the application is approved and activated. Required challenges are to find problems in this process and improve them. Provided event logs are related to the operational data for 165 days and contain some 262,200 events in 13,087 cases.

5.1 Business state model

(1) Essential structure

At first, we extract following nine *W-SM*s based on the specifications described in the outline of the business process and the event logs.

- one *P-SM*: showing a main process to approve an application.
- one *upper S-SM*: showing a sub-process to offer and assess an offer.
- seven *S-SM*s: showing tasks to follow up submission, complete application, investigate fraud case, follow up offer, assess application, seek additional information, and modify contract.

Next, we assign act events to each *W-SM*, place every *W-SM* hierarchically, and assign a link event to connect "ACCEPT" event of the *P-SM* and "SEND" event of the *upper S-SM*. At last, we verify the essential structure by the event logs. We can confirm that the value of *SoA* (Suitability of act event), *SoE* (Suitability of embedded event), and *SoL* (Suitability of link event) all indicate more than 0.8. Therefore, the essential structure is confirmed to be the model reflecting the real event logs.

(2) Incidental structure

We can evaluate the incidental structure whether it provides effective metrics to improve the process. In the process improvement, we usually use a problem analysis diagram, as described in Fig. 4. Firstly, we write down the problems founded in the process on the leftmost side in this diagram. Secondly, we find causes of each problem and add them on the right of it. Thirdly, we repeat the second procedure regarding each cause as a new problem until we cannot find any causes. The behavior metrics of the incidental structure are used to quantitatively confirm the validity of causality relationship between each problem and its causes.

Fig. 4 shows that the problem found in the process is "customer lead time is so long." The causes of the problem can be "time to check application form is long" and "time to examine application is long." In order check the validity of these causes candidates, we calculate the performance metrics: state lead-time (SLT) of the SUBMITTED state and ACCEPTED state. As SLT of ACCEPTED state is much longer than SUBMITTED state as described in Fig. 5, the cause of the problem can be "time to examine application is long." Regarding this cause as a new problem, more detailed cause candidates: "time to negotiate offer is long" and "time span to finish assessment is long" are added. Both of these cause candidates are confirmed to be true using SLTs related to them as described in Fig. 5.

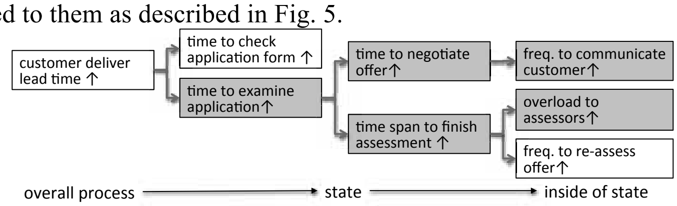

Fig. 4. Problem analysis diagram of consumer loan approval process

The causes on the rightmost side in Fig. 4: "frequency to communicate customer is high," "overload of assessors is large," and "frequency to re-assess offer is high" are related to the frequency of events. Therefore, we calculate the frequency metrics of specified tasks and resources, and check the correlation between the frequency metrics and the performance metrics as described in Fig. 6. As a result, the fundamental causes of the problems prove to be "frequency to communicate customer is high" and "overload to assessors." Therefore, the incidental structure is confirmed to be the powerful tool for the process improvement.

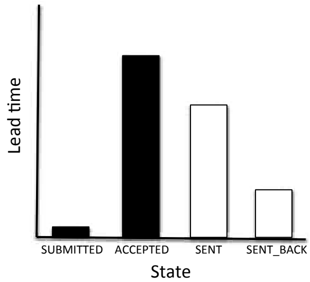

Fig. 5. State lead-time

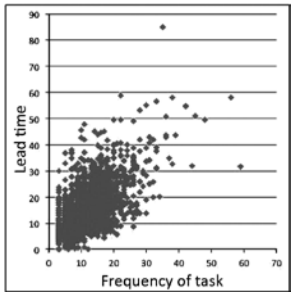

Fig. 6. Correlation between freq. and perf.

6 Conclusion

In order to apply process mining to process improvement of the knowledge work, this paper proposed a business state model consisting of an essential structure and an incidental structure. This model was implemented as a software tool and evaluated with real event logs. The result shows the essential structure to be the business logic model reflecting event logs and the incidental structure to be a powerful tool for process improvement. Future works aim to establish a methodology of business process improvement through applying to varieties of real business processes. This work was supported by a Senshu University overseas' research program grant in 2013.

References

1. Keith D. Swenson et al.: Mastering the Unpredictable, Meghan-Kiffer Press (2010).
2. W. M. P. van der Aalst: Process mining – Discovery, conformance and enhancement of business process, Springer-Verlag, Berlin (2011).
3. W. M. P. van der Aalst et al.: Process mining – two-step approach to balance between underfitting and overfitting, Software system model, pp. 87-111, Springer (2010).
4. A. Nigam, N. S. Caswell: Business artifacts - An approach to operational specification, IBM Systems Journal, Vol. 42, No. 3 (2003).
5. P. Borstorff and R. Rosenbaum: Supply chain excellence – A handbook for dramatic improvement using the SCOR model, Amacom Books (2003).
6. J. L. G. Dietz: Enterprise Ontology – Theory and methodology, Springer-Verlag, Berlin (2006)
7. R. Medina-Mora, T. Winograd, R. Flores, and F. Flores: The action workflow approach to workflow management technology, Proc. of CSCW92, pp. 281-288 (1992).
8. D. Harel: Statecharts – A visual formalism for complex systems, Science of computer programming 8, pp. 231-274, North-Holland (1987).
9. http://www.uml-diagrams.org/state-machine-diagrams.html
10. http://www.win.tue.nl/bpi/2012/challenge