Real-time business process monitoring using formal concept analysis

Bokyoung Kang[1], Jae-Yoon Jung[2], Nam Wook Cho[3], and Suk-Ho Kang[1]

[1] Department of Industrial Engineering, Seoul National University, Korea
[2] Department of Industrial and Management Systems Engineering, Kyung Hee University, Korea
[3] Department of Industrial and Information Systems Engineering, Seoul National University of Science and Technology, Korea

Abstract:

Purpose - To help industrial managers monitor and analyze critical performance indicators in real-time during the execution of business processes, we propose a visualization technique using an extended formal concept analysis (FCA). The proposed approach monitors the current progress of ongoing processes and periodically predicts their probable routes and performances.

Design/methodology/approach - FCA is utilized to analyze relations among patterns of events in historical process logs, and this method of data analysis visualizes the relations in a concept lattice. To apply FCA to real-time business process monitoring, we extended the conventional concept lattice into a reachability lattice, which enables managers to recognize reachable patterns of events in specific instances of business processes.

Findings - By using a reachability lattice, expected values of a target key performance indicator (KPI) are predicted and traced along with probable outcomes. Analysis is conducted periodically as the monitoring time elapses over the course of business processes.

Practical implications - The proposed approach focuses on the visualization of probable event occurrences on the basis of historical data. Such visualization can be utilized by industrial managers to evaluate the status of any given instance during business processes and to easily predict possible subsequent states for purposes of effective and efficient decision-making. The proposed method was developed in a prototype system for proof of concept and has been illustrated using a simplified real-world example of a business process in a telecommunications company.

Originality/value - The main contribution of this paper lies in the development of a real-time monitoring approach of ongoing processes. We have provided a new data structure, namely a reachability lattice, which visualizes real-time progress of ongoing business processes. As a result, current and probable next states can be predicted graphically using periodically conducted analysis during the processes.

Keywords: Formal concept analysis (FCA), Concept lattice, Reachability lattice, Business process monitoring, Business activity monitoring (BAM)

Paper type: Research paper

Published in Industrial Management and Data Systems, Vol. 111, No. 5, Jun 2011, pp. 652-674.

1. Introduction

A business process is a set of activities performed in order to achieve a common goal according to well-defined objectives of an enterprise (Hammer and Champy, 1994; Keung and Kawalek, 1997). To sustain competitiveness in a rapidly changing business environment, it is necessary for companies to manage their business processes in real-time (Lam *et al.*, 2009). The need to incorporate real-time visibility into business processes has led to the increasing adoption of process monitoring systems in the business sector (Kang *et al.*, 2009a).

Business process monitoring provides real-time access to critical performance indicators of ongoing processes and supports decision-making and risk management during execution of the processes. Integrated with event processing technologies, the monitoring of current business processes is based on knowledge extracted from historical process logs in order to detect distinct patterns in event data, which enable businesses to operate proactively (Buytendijk and Flint, 2002; Grigori *et al.*, 2004). This knowledge-based approach to monitoring is called rule-based monitoring. It aims at analyzing data generated from interrelated activities executed within a business, so that relations between process attributes and performance outcomes can be defined in a form of an 'If (condition) - Then (status)' rule (Luckham, 2002; Lundberg, 2006; Mendling *et al.*, 2008). Utilizing these rules, a monitoring system can evaluate the current status and predict possible outcomes for various business processes (Wang and Romagnoli, 2005; Bose, 2006). Most existing process monitoring systems, however, still have limitations that need to be improved, since existing systems mainly deal with the monitoring of completed process instances in which all attributes are simultaneously recorded and analyzed (Kang *et al.*, 2009b). In executing real-time business processes, however, events are generated by sequences of relevant activities. If correlations among generated events foreseeably influence the performance outcome upon completion of the process, the monitoring system should predict the final performance based on the current status and real-time progress of the process in order to enable more proactive operation.

To improve these limitations, we proposed a novel approach for real-time business process monitoring using extended formal concept analysis (FCA). FCA is a mathematical tool for the investigation of objects comprised of multiple attributes in order to reveal relations among them (Boucher-Ryan and Bridge, 2006). FCA is often used to extract knowledge that is dependent on previous cases in order to support decision-making with current problems (Díaz-Agudo and González-Calero, 2001). A concept lattice developed from FCA serves as a data structure to visualize relations among previous cases. In this paper, we analyze historical process logs using FCA and investigate the effects of co-occurrences of events on performance outcomes of the processes. Structural visualization of event patterns on the concept lattice is also provided to facilitate effective process monitoring.

As the data structure of a conventional concept lattice also has limitations when applied to real-time monitoring, we devised an extended concept lattice, namely a reachability lattice, to visualize probable routes in event co-occurrences in the real-time progress of any ongoing business process.

This paper provides an approach to real-time business process monitoring using a reachability lattice; an event-based process monitoring model is presented along with the conceptual procedure of real-time process monitoring. FCA is applied to investigate the relevancy of past events in historical cases, and a concept lattice is

constructed to structurally represent the data. To modify the concept lattice, we subsequently developed an extended concept lattice, which we defined as a reachability lattice. During the real-time progress of an ongoing process, the current state is evaluated and possible next states are predicted. This is conducted periodically based on observed event logs. Finally, an example scenario is presented with a prototype system to show how the reachability lattice can be utilized in real-time process monitoring. The prototype system was developed on the open-source business process mining framework, ProM (van der Aalst *et al.*, 2007).

The rest of this paper is organized as follows. In Section 2, we present a literature review to describe existing approaches in business process monitoring and other applications of formal concept lattice data analysis. Based on this background, we explain our motivations for the proposed approach. The formal model of event-based process monitoring and the conceptual procedure of real-time process monitoring are described in Section 3. Subsequently, we explain construction of the reachability lattice in Section 4. How to apply the reachability lattice to real-time business process monitoring is then described in Section 5. In Section 6, a simplified real-world example scenario is illustrated with a prototype system. Finally, conclusions and suggestions for future work are summarized in Section 7.

2. Background

2.1. Business process monitoring

A study by Beckett *et al*. (2000) describes the history of business process monitoring as follows. Early studies on process monitoring focused on the collection and analysis of large data sets, mainly in a manufacturing domain. As changes in industry became more rapid and complexity increased, expert systems were introduced, but autonomous operation remained difficult. To overcome these limitations, approaches to process monitoring based on knowledge extraction were suggested. Recently, inductive techniques such as rule-based approaches using data mining, inference models using case-based reasoning, and machine learning methods have been adopted in process monitoring.

One of the most widespread monitoring approaches is knowledge extraction using data mining techniques. Grigori *et al*. (2004) suggested a business intelligence model based on a decision tree. Historical process logs, which are stored in business process management systems, are analyzed with a decision tree to extract rules for current process monitoring. Widodo and Yang (2007) reviewed approaches to machine condition monitoring using a support vector machine. Special features that cause particular faults can be derived by support vector machine classifiers in order to detect faults in current machine conditions. Peddabachigari *et al*. (2007) proposed a hybrid monitoring model combining a support vector machine with decision tree analysis. The study detected occurrences of intrusion by analyzing correlations among events that occurred during transactions. Abad-Grau and Arias-Aranda (2006) proposed a Bayesian classification algorithm to analyze relations between operation strategy and flexibility. A Bayesian network was constructed to visualize extracted knowledge. Ho *et al*. (2008) proposed a fuzzy-based generic algorithm to identify possible solutions for the improvement of quality in supply chain management. Numerical process parameters were indentified for quality control, monitoring workflow, and evaluation.

All of these approaches, however, have limitations that prevent effective application to real-time process

monitoring because they assume completed, as opposed to ongoing, processes. Most importantly, extracted knowledge is simply not available until entire events are recorded, which leaves any knowledge based monitoring system idle during the execution of processes (Grigori *et al*., 2001; Grigori *et al*., 2004). Consequently, real-time feedback cannot be obtained. Moreover, comprehensive indicators representing real-time progress are not periodically provided (Curtis *et al*., 2008).

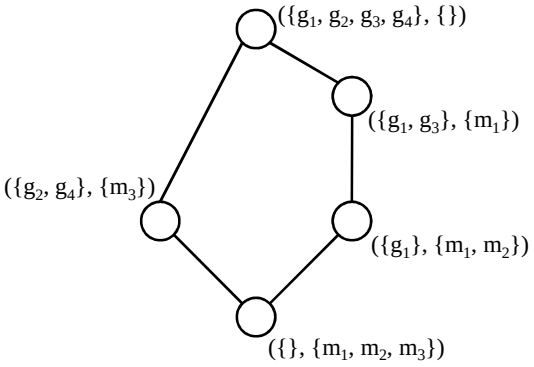
2.2. Formal concept analysis

FCA is a mathematical tool that visualizes data and their inherent structures, implications and dependencies (Wormuth and Becker, 2004; Wille, 1982). FCA analyzes relations among objects that have several attributes.

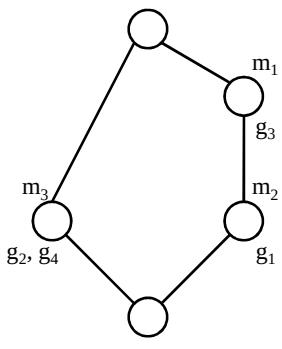
The basic notions of FCA are a formal context and a formal concept (denoted as context and concept, respectively). A context (G, M, I) consists of a set of objects G, a set of attributes M and the relations I between G and M. Thus, a relation *gIm* means that an object g ($\in$G) has an attribute m ($\in$M). A context with finite sets can be represented using a cross table as described in Figure 1 (a). For a set A$\subseteq$G, A' is defined as the common attributes of objects in A. Similarly, for a set B$\subseteq$M, B' is defined as objects that have all attributes in B. Using these properties, a concept of the context (G, M, I) is defined as a pair (A, B) where A$\subseteq$G, B$\subseteq$M, A'=B and B'=A, and A is defined as the extent and B is defined as the intent of the concept (A, B). When (A_1, B_1) and (A_2, B_2) are two concepts of a context, (A_1, B_1) is defined as a sub-concept of (A_2, B_2), provided that $A_1 \subseteq A_2$ and equivalently $B_2 \subseteq B_1$. Respectively, (A_2, B_2) is a super-concept of (A_1, B_1) and is represented as (A_1, B_1) $\leq$ (A_2, B_2), which is defined as the hierarchical order. Using the hierarchical orders of all concepts in the context, a concept lattice can be generated as a graphical representation of relations among concepts, as shown in Figure 1 (b). Figure 1 (c) is another representation of (b), called a concept lattice with reduced labeling. To improve the readability of the diagram, duplicate objects and attributes are removed in Figure 1 (c). As the concept becomes more detailed, visual representation is located closer to the bottom of the lattice. Therefore, objects in the concept inherit all attributes from their parent node. In Figure 1 (c), for example, object m_1 has an attribute g_3 and object m_2 has two attributes - g_1 and g_3.

	m_1	m_2	m_3
g_1	1	1	
g_2			1
g_3	1		
g_4			1

(a) cross table



(b) concept lattice



(c) concept lattice with reduced labeling

Figure 1: Context, concept lattice and concept lattice with reduced labeling

There are also other mathematical or statistical methods, such as analytic hierarchy process, analytic network process, or other optimization models that are used for similar purposes (Carpineto and Romano, 2004). Compared to these methods, one of the distinct advantages of FCA is that it analyzes structural similarities among cases in question and enables a more accessible visualization of data (Ganter and Wille, 1999; Priss, 2006; Solesvik and Encheva, 2010).

Thanks to capabilities such as the investigation and visualization of large sets in a database, FCA has been applied to a number of engineering domains (Hernández *et al*., 2002). Beydoun (2009) utilized FCA as a knowledge acquisition tool to visualize dependencies between web pages based on keywords, so that a navigation system for internet searches was effectively constructed from previous virtual surfing trails. Jiang *et al*. (2003) developed an ontology-forming support system in a clinical domain using FCA, in which FCA was used to construct a class hierarchy by integrating linguistic knowledge information with contextual knowledge. Hsieh and Wang (2010) utilized FCA in a web-based learning support system to construct a suitable learning path by computing mutual relationships among documents based on extracted keywords. Arévalo *et al*. (2010) suggested an automatic schema generation for software reengineering systems, in which FCA was used to identify class hierarchy schemas by revealing undocumented hierarchical dependencies between classes according to their behaviors and states. Park (2000) presented a FCA-based approach to software engineering in which software components were analyzed using FCA in terms of their inputs and outputs so that users could retrieve desired components from the component library. Boucher-Ryan and Bridge (2006) applied FCA to collaborative recommendation systems by analyzing relations between users and items in order to identify user experiences and preferences. Díaz-Agudo and González-Calero (2001) showed how FCA could support case-based reasoning. By analyzing historical cases and visualizing their relations, FCA was used to foster the query formulation process to find patterns, regularities, and exceptions among the cases. As shown in existing studies, FCA effectively analyzes relations among large cases and also visualizes the results in comprehensible graphs. Thus, in our research, we applied FCA to investigate business cases and visualized relations among event occurrences in terms of business performance management.

2.3. Motivations

Based on the literature review, requirements for a real-time monitoring system are identified as follows. A business process monitoring system should represent real-time progress based on observed events throughout any monitoring period, as well as predict subsequent possible outcomes with appropriate indicators. Also, it should operate periodically during the progress of entire monitoring periods. Although some studies that recommend real-time monitoring approaches using a decision tree have been conducted, (Kang *et al*., 2009a; Kang *et al*., 2009b), they are limited in their representation of probable routes and co-occurrences of events. A structured inference model is more efficient than are other statistical or artificial intelligence methods (Abad-Grau and Arias-Aranda, 2006). Therefore, a visualization technique is arguably a more effective approach.

In this paper, we applied FCA to the analysis of historical process logs. By using FCA, we were able to analyze relations between the co-occurrences of events during process execution periods and the final performances of completed processes. Additionally, hierarchical orders in distinct patterns of co-occurrences

were analyzed and visually represented as a concept lattice.

A conventional concept lattice, however, also has limitations similar to those of existing process monitoring approaches, as a conventional concept lattice does not effectively represent probable routes or co-occurrences of events. To mitigate these limitations, we developed an extended concept lattice and defined it as a reachability lattice. When real-time process monitoring is compared to other applications of FCA, one significant difference is that, in real-time monitoring, attributes noted as events are obtained sequentially during the real-time progression of the process, and therefore courses can and should be traced in real-time. In this study, probable transition information between distinct event patterns was added to the conventional concept lattice, which was extracted from possible sequences in event generation.

3. Event-driven business process monitoring

As many of modern enterprise information systems such as enterprise resource planning (ERP) and supply chain management (SCM) systems are adopting the concept of business process modeling and automation, they are often called process-aware information systems (PAIS) (Dumas *et al.*, 2005). PAIS uses business process models to execute, monitor, and analyze core business processes. For example, SAP ERP systems contain hundreds of business process reference models that are generally customized when the systems are implemented for domain-specific organizations. In this paper, we propose an event-driven approach to business process monitoring.

Before introducing the proposed framework, we first describe the assumptions of PAIS upon which the proposed approach is based. First, PAIS supports business activities of workers on the basis of predefined business process models. A business process model typically contains the flow of tasks and the characteristics of each task (for instance, the worker roles, pre- and post-conditions of each task). In addition, for the purpose of business process monitoring and analysis, the business process model indicates significant variables and events such as cycle time and overdue. All the data are managed and controlled on the basis of business process models.

Second, PAIS continually records the results of business process execution in its system logs. Each case of business process execution is called a business process instance, and the results of the process instance also include the values of KPI and the information of event occurrences with their time-stamps. The two kinds of information are the core components of the proposed process monitoring method.

Finally, event occurrences in business process execution can be intimately related to the performance of business process, that is, the values of KPI. Therefore, it is assumed that any significant state which may affect business performance can be chosen as an event for business process monitoring by business analysts. For example, the complaint of a customer surely affects the satisfaction of the customer in service processes. And, the delay of order confirmation may cause the overdue of service processes. Such significant events are recorded in the system log of PAIS to effectively monitor and analyze business processes.

Under these assumptions and system environment, we present a business process monitoring framework based on event processing and analysis, and then provide a formal model that can be used in the monitoring framework.

3.1. Framework

The framework of the proposed business process monitoring is illustrated in Figure 2. PAIS executes business tasks based on predefined *business process models*. It is assumed that the significant events which may affect the performance of the business process have already been predefined by business analysts. In addition, the events can be mapped to the corresponding activities in the process model, which are represented in *event-activity models*. Upon completion of business processes, the results of the business process execution including event occurrences and business performances are recorded in the *historical process log* of PAIS. From the system log, the results are represented in *event history models* and then they are used to construct a reachability lattice, which will be introduced as a new knowledge representation model in Section 4. Based on the real-time event occurrences, the lattices are utilized to predict their performances and control business processes in the stage of *real-time process monitoring*.

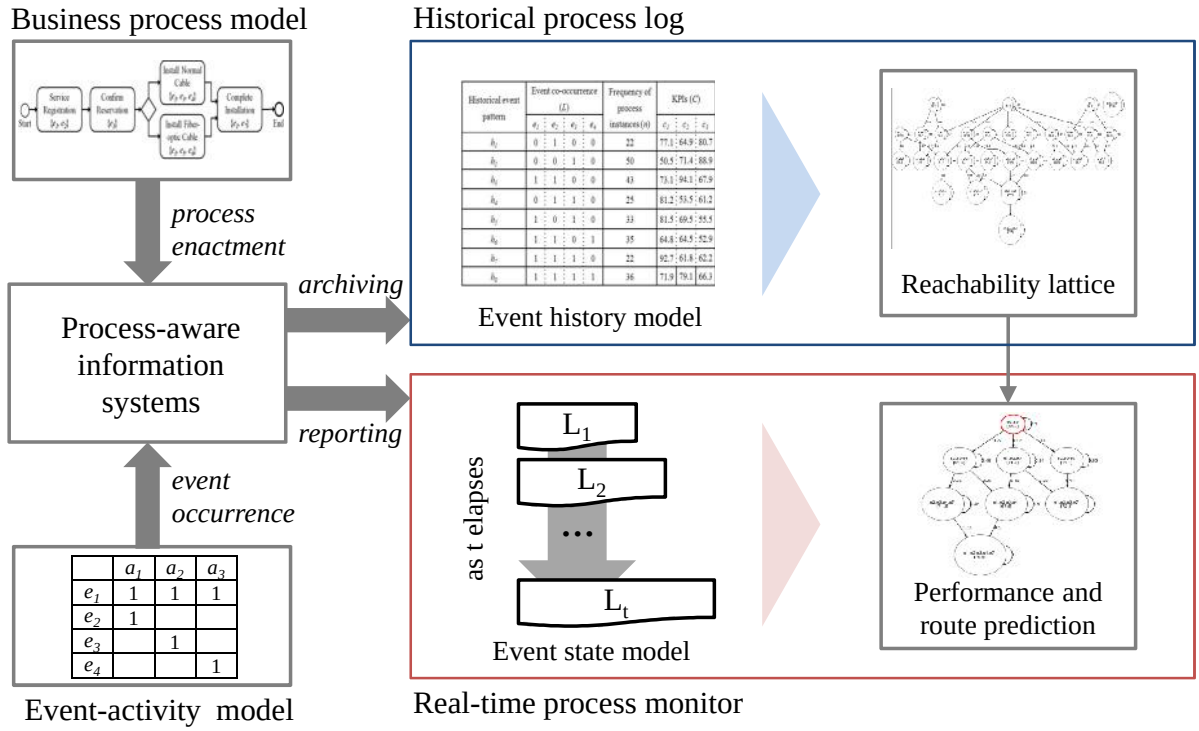


Figure 2: The framework of real-time business process monitoring

The overall procedure of the proposed real-time business process monitoring is illustrated in Figure 3. The procedure is largely divided into two phases. In the first phase, business analysts decide the scope of business process monitoring and construct a reachability lattice from the event history models. First, a user selects a business process as a target of business monitoring. For the business process, KPI and its related events are also selected as objects to be observed. However, there may be too many events to cope with in real-time so that KPI-oriented event evaluation is conducted to filter events related to the KPI. The filtering aims at excluding events having little effects on predicting the final value of the KPI. From the historical process log of the PAIS, a reachability lattice is constructed to analyze the relations between co-occurrences of events related to the KPI

prediction. The reachability lattice is composed of nodes connected with directed edges. Each node implies a process pattern as co-occurrences of events, and two nodes are connected when one pattern can reach another if an additional event is generated. The lattice is used in monitoring in the business process and predicting the KPI.

The second phase of the procedure is to event-driven monitoring in run-time. When monitoring an ongoing business process, the progress of the business process is observed according to a set of events occurred. The corresponding pattern of event occurrences can be derived from the reachability lattice. If the current state of event occurrences is matched with the reachability lattice constructed in build-time, values of KPI can be predicted along with the probable next patterns. By considering historical data for each pattern such as frequency or average performance, practicable outcomes can be periodically analyzed based on occurred events in real-time. These are conducted periodically when additional events are observed with the real-time progress.

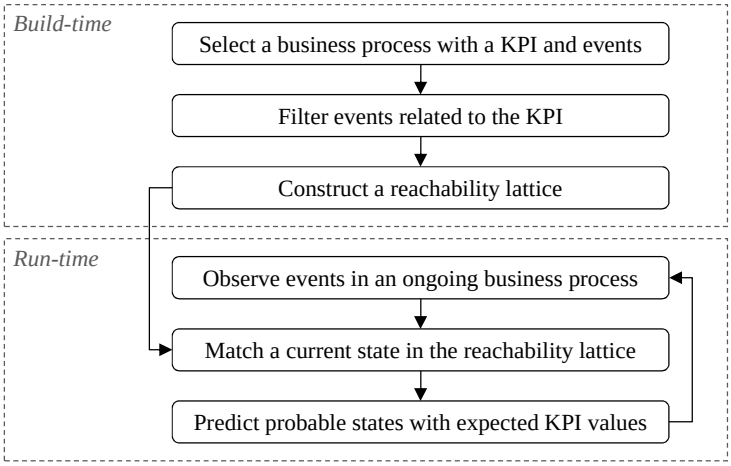


Figure 3: Overall procedure of real-time business process monitoring

3.2. Formal model

An event is defined as the occurrence of a significant incident during the execution of any given business process. Through the aggregation and analysis of process logs, knowledge about the quality of circumstances occurring can be extracted (Grigori *et al*., 2004). In this study, we assume that correlations of event occurrences influence performance of business processes, which are measured using KPI. Under these assumptions, the event-activity model in Definition 3.1 is described to define the scope of occurrences for each event in a process model.

Definition 3.1. *An event-activity model of a process design is a tuple* $M=<E, A, S, D>$.

- $E=\{e_i\}$ *is a set of events which can occur in the execution of the process.*
- $A=\{a_j\}$ *is a set of activities which are executed in the process and may generate events in E.*
- $S=A\times A$ *is the relation between activities which restricts the execution order of two activities, e.g.,* $s=(a_j, a_{j'})$ *in S means that activity* a_j *is executed earlier than activity* $a_{j'}$.
- $D=E\times A$ *is the relation between events and activities representing those activities events that can be generated;* $d=(e_i, a_j)$ *in D means that event* e_i *can be generated during the execution of activity* a_j.

Figure 4 shows an example of a business process diagram including activities and the corresponding events. According to the diagram, the event-activity model is described as $E=\{e_1, e_2, e_3, e_4\}$, $A=\{a_1, a_2, a_3\}$, $S=\{(a_1, a_2), (a_1, a_3), (a_2, a_3)\}$, and $D=\{(e_1, a_1), (e_1, a_2), (e_1, a_3), (e_2, a_1), (e_3, a_2), (e_4, a_3)\}$.

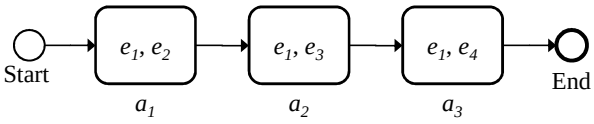


Figure 4: Event-based process diagram

In general, it is highly likely that a KPI is influenced by only a few process events, not by all of the events generated during any given process. To facilitate efficient monitoring, a KPI-oriented event analysis utilizes historical process logs to identify process events that have significant effects on the KPI. Each event can be evaluated for a given KPI *c* using the following metric.

$$eval(e) = \log_2 \frac{|I^+(e)|/|I^+|}{|I^-(e)|/|I^{L-}|}$$

In the evaluation metric, I^+ and I^- are the respective sets of positive and negative process instances, and $I^+(e)$ and $I^-(e)$ are the respective subsets of I^+ and I^- which contain the process instances with the observation of event *e*. Therefore, $|I^+(e)|/|I^+|$ (and $|I^-(e)|/|I^-|$) is the probability of the process instances containing event *e* in positive instances I^+ (and negative instances I^-). Therefore, *eval*(*e*) is the logarithmic ratio of the two probabilities, implying a relative frequency of positive cases to negative cases. For instance, if *eval*(*e*) is zero, *e* has nearly no influence on the value of KPI. If *eval*(*e*) is positive, *e* has a positive influence, and if negative, e has a negative influence on KPI. Moreover, since the evaluation metric is a logarithm with base 2, *eval*(*e*)=1 implies that the probability of event occurrence *e* in positive instances are double that in negative instances.

After selecting a target KPI from the KPI list of a given business process, the KPI-oriented event evaluation identifies significant events to be observed during real-time monitoring. When a process is initiated, a process instance is created for its execution, and relevant events are stored in the process log during execution of the process. To represent process instances and events in the log, an event history model *H* is formalized in Definition 3.2.

Definition 3.2. *An event history model H for an event-activity model M is a tuple, H=<L, n, c>.*

- *A binary function f: $E \rightarrow Q_H$ specifies the occurrence of an event in a historical process instance, that is, $q_H \in Q_H=\{0,1\}=\{$nonoccurrence, occurrence$\}$.*
- *$L= Q_H^1 \times Q_H^2 \times \ldots \times Q_H^{n(E)}$ is a binary relation of event occurrences Q_H where n(E) is the number of events in M; $l \in L$ represents a case of event co-occurrences in process instances stored in the process log.*
- *n is the frequency of the process instances which contain the event co-occurrence $l \in L$.*
- *c is an average value of the target key performance indicator of the n process instances which contain l.*

Table I. Example of an event history model H

Historical event pattern	Event co-occurrence (L)				Frequency of process instances (n)	KPI (c)
	e_1	e_2	e_3	e_4		
h_1	0	1	0	0	22	77.1
h_2	0	0	1	0	50	50.5
h_3	1	1	0	0	43	73.1
h_4	0	1	1	0	25	81.2
h_5	1	0	1	0	33	81.5
h_6	1	1	0	1	35	64.8
h_7	1	1	1	0	22	92.7
h_8	1	1	1	1	36	71.9

Table I shows an example of an event history model for the example process in Figure 4. Each row contains event co-occurrences (L), the frequency of process instances for L(n), and average values of KPI (c). To construct an event history model, completed process instances are grouped in a log according to event co-occurrence. Subsequently, the average value of KPI is calculated from the grouped process instances. For example, historical event pattern h_1 shows that there were 22 completed process instances, which have the event occurrence of only e_2. The average value of KPI for the event pattern is 77.1.

During process execution, a process instance is monitored through events that occur up until a specific monitoring instant. To represent temporal process monitoring, a monitoring instant t is defined as a moment at which the event log is actively investigated for observation. A monitoring interval Δt is a predefined time interval or the period in which a new event is generated after occurrence of the previous event. Therefore, an event state model O at monitoring instant t is formulated in Definition 3.3. Note that the monitored instance is not yet completed, but rather it is still in progress in real-time, so that event occurrences are obtained only in completed activities of the process instance.

Definition 3.3. *An event state model O observed at monitoring instant t is defined as L_t, to specify event occurrences during process execution up to time t.*

- *A binary function g: $E \rightarrow Q_O$ specifies the occurrence of an event in an ongoing process instance, that is, $q_O \in Q_O = \{-1,0,1\} = \{$undetermined, nonoccurrence, occurrence$\}$.*
- *$L_t = Q_O^1 \times Q_O^2 \times \ldots \times Q_O^{n(E)}$ is a binary relation on event occurrences Q_O where n(E) is the number of events in M; $l_t \in L_t$ represents a state of event occurrences in the process instance at time t.*

For instance, suppose a process instance which is executed for the example model in Figure 2 has the event state L_t = (-1, 1, 0, -1) at monitoring instant t. This means that for the ongoing process instance, only event e_2 has occurred, but e_3 has not at time t, and the occurrences of e_1 and e_4 are not determined because activity a_4 is not complete. Note that a_4 may generate e_1 or e_4 or both when the activity is executed completely.

4. Reachability lattice

In this section, we explain how to construct a reachability lattice from a conventional concept lattice. As described in previous sections, there are limitations when a concept lattice is utilized for real-time process monitoring. Although a concept lattice graphically shows relations among event occurrences of completed process instances, it is not suitable to describe the incremental changes in event occurrences of process instances. In this paper, we developed an extended concept lattice, the reachability lattice, defined as a directed network extending from the conventional concept lattice in order to represent reachable relations among event occurrences in historical process logs.

There are four steps to constructing reachability lattices for a given business process. First, a concept lattice is constructed from a historical process log. Second, a state is defined for each node of the concept lattice to describe the probable status of event occurrences. Third, an undirected edge representing the hierarchical order of states is modified to a directed edge representing reachability among the event patterns, which is defined as a reachable order. Finally, the probability of the state being changed by the orders is estimated according to the cumulative frequencies of the states.

Step 1: Construct a concept lattice

Based on historical process instances, as shown in Table I, a *context (H, E, I)* is derived to construct a concept lattice. A set of historical instances H and a set of events E can be considered sets of objects and attributes in a concept lattice, respectively. The relation I is defined as hIe between H and E when object h has attribute e. For example, in Table I, h_1 has the relation of events as (e_1, e_2, e_3, e_4) = (0, 1, 0, 0) when event e_2 is generated in historical process instances. This is interpreted to mean that object h_1 has attribute e_2, denoted as h_1Ie_2. A concept lattice is constructed by following the hierarchical order of all objects in the context. Each node on the concept lattice represents one *concept (A, B)* where $A \subseteq H$ and $B \subseteq E$.

Step 2: Define a state

To identify a distinct event pattern for each concept, we define a state on each node to represent an event pattern that occurred in a historical process instance. Therefore, a state can be considered as a distinct event pattern in an event history model. On each node of the concept lattice constructed in step 1, the state is formulated as described in Definition 4.1.

Definition 4.1. *A state is defined as a condition of event co-occurrence for a corresponding concept on each node in a concept lattice. For each concept (A, B), the state is defined as* $s=\{e_i|f(e_i)=1\}$*, which equals the set of events, B, of the concept. A is a set of event patterns that have event occurrences at least equal to or greater than B. Therefore, state s is a common set of occurred historical instances corresponding to event patterns in A, which is the smallest set of occurred historical instances among the event patterns in A.*

Step 3: Define a reachable order

Based on execution orders between activities, probable temporal sequences are identified between

generations of events. According to the event-activity model, when there exist (e_2, a_1) and (e_3, a_2), the occurrence of e_2 should precede that of e_3 because a_1 is executed before a_2, which is denoted by $e_2 \rightarrow e_3$. Based on such temporal sequences, reachable orders between two states are described using a lattice as formulated in Definition 4.2.

Definition 4.2. *A reachable order means that a state can be transited to another probable state after an additional event is generated. Suppose that there are two states s_A and s_B corresponding, respectively, to two concepts $C_A(A, A')$ and $C_B(B, B')$. The reachable order from s_A to s_B, denoted by $s_A \rightarrow s_B$, is defined by satisfying the following properties:*

- *C_A is the super-concept of C_B.*
- *$e_A \rightarrow e_B$ for all pairs of $e_A \in A'$ and $e_B \in B' \backslash A'$.*

Using the reachable order, a reachable state set and a directed edge are formulated as described in Definition 4.3.

Definition 4.3. *The reachable state set of state s, denoted by S, is defined as a set of states including s and other states satisfying reachable orders with s. Then, a directed edge is drawn from s to each of S. For example, if there are two states s_A and s_B with $s_A \rightarrow s_B$, the reachable state set of s_A is $S=\{s_A, s_B\}$. Therefore, the directed edge from s_A to s_B and the recursive edge from s_A to s_A are defined.*

In a conventional concept lattice, concepts are connected according to hierarchical order. Therefore, a concept is connected to all of the concepts, including additional events, from the concept. However, when considering temporal sequences in event generations, some hierarchical orders might not be permitted; therefore, these edges should be eliminated from the reachability lattice. States in a reachability lattice are connected by partial, reachable orders. Moreover, to represent the status maintained in real-time progress without generation of additional events, we added a recursive directed edge. As a result, a reachability lattice can periodically trace changes in co-occurrences of events as monitoring instants elapse.

Step 4: Estimate a probability of reachability

For each directed edge developed from step 3, a probability of reachability is estimated as a probability that a state can obtain a reachable state rather than remain in the current state, along with the directed edge. Using the frequency of each event pattern corresponding to the state in Table I, a probability of reachability is estimated as follows.

Definition 4.4. *On each directed edge, probability of reachability r is defined as a probability that a state can obtain a reachable state. Suppose state s_0 has a reachable set $S=\{s_1,\ldots,s_k\}$. The probability of reachability from s_0 to s_i, denoted by r_i $(0 \le i \le k)$, is estimated using the following formula, where n_i is the frequency of historical process instances corresponding to s_i.*

$$r_i = \begin{cases} n_0 / N_0, & for\ i = 0 \\ N_i / N_0, & for\ i = 1,...,k \end{cases}, \text{ where } \quad N_i = n_i + \sum_j N_j \quad \text{for every } s_j \text{ which satisfies } s_i \rightarrow s_j$$

Figure 5 shows the four steps in the construction of a reachability lattice using the data in Table I. In the concept lattice of Step 1, each node, a so-called concept, is composed of two sets of objects and attributes, which correspond to historical event patterns (*h*) and occurred events (*e*), respectively. These concepts are connected with undirected edges as hierarchical orders, which represent only relations of inclusion. The concepts at the bottom of a lattice are more detailed and have more attributes than those at the top. In Step 2, we first define a state of occurred events that can be matched to an ongoing instance. As execution periods elapse, additional events are generated, which means that the state of an instance changes to another state on an ongoing basis. However, existing hierarchical orders might not enable these changes. Therefore, in Step 3, we modified hierarchical orders as reachable orders to better represent the progress of events in real-time. Subsequently in Step 4, a probability of reachability is estimated using the frequency of each state as it corresponds to the historical event pattern.

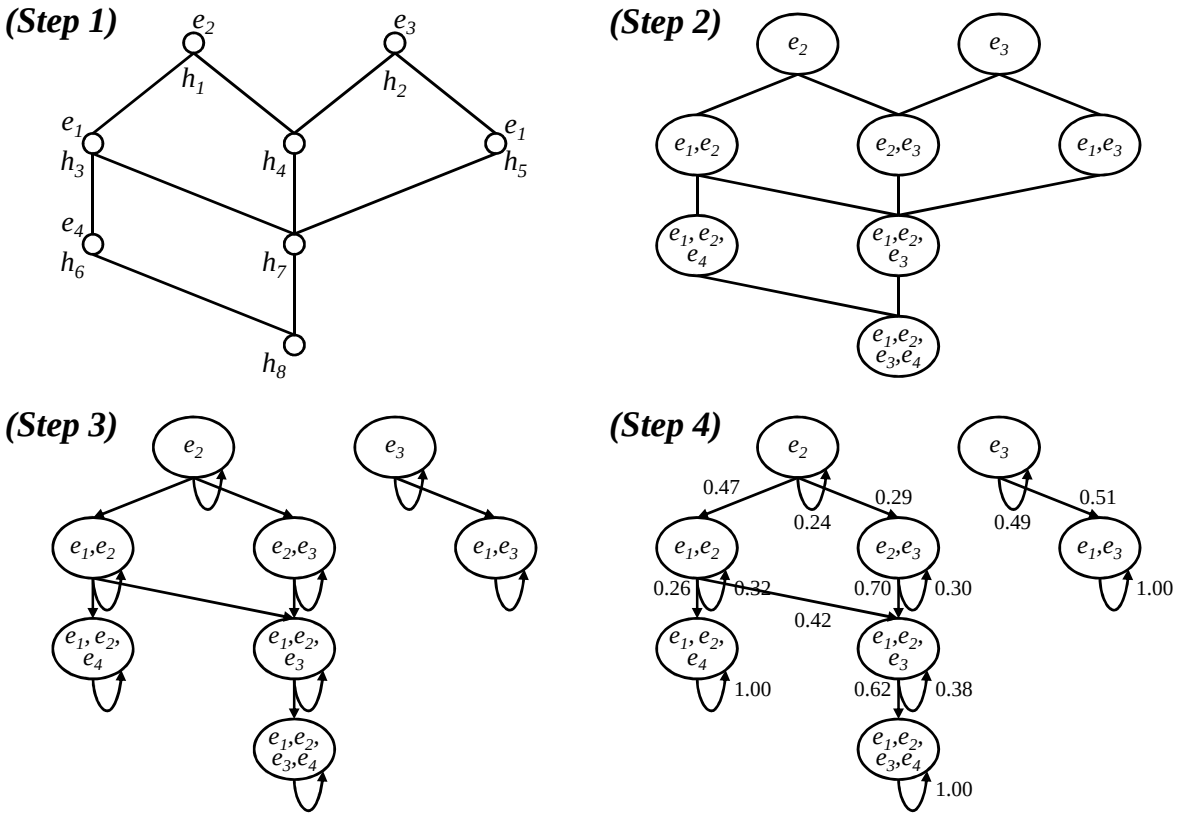


Figure 5: Constructing reachability lattice from concept lattice

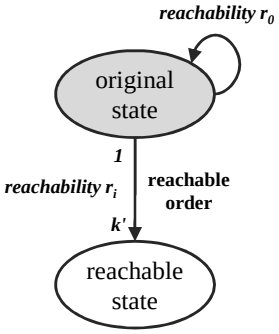


Figure 6: Basic notions of reachability lattice

Figure 6 shows basic notions of the reachability lattice extended from the concept lattice. The original state can remain unchanged or acquire another state, such as those with reachable orders represented as directed edges. Possible transitions are estimated as probabilities for each directed edge. Compared with a conventional concept lattice with undirected hierarchical orders, a reachability lattice better represents reachability among historical event patterns.

5. Applications

In this section, we illustrate how a reachability lattice can be applied to real-time process monitoring. At a monitoring instant t, an event log of an ongoing process instance is obtained as L_t, which is an incrementally increasable set of events in midcourse. During entire process execution periods, L_t is monitored periodically as t elapses. When the observed partial event log L_t is obtained, it is possible to diagnose a current state and predict next probable states using a reachability lattice. First, a state of the ongoing process instance at t is retrieved as $s_t=\{e_i|g(e_i)=1 \text{ in } L_t\}$, which is a set of events that occurred up to t.

5.1. Performance prediction

Based on L_t, an expected performance upon completion is predicted by considering reachable states S of s_t corresponding to L_t. When s_t is obtained, the possible state is any in a reachable state set, which is monitored periodically throughout the generation of additional events. The purpose of KPI prediction is to foresee changes in the expected value of the target KPI along with reachable orders.

The expected value of target KPI c_t is defined as the predicted value of KPI when the current state s_t is maintained through completion. It is predicted as the average KPI value of historical instances h for the state corresponding to the event log of the ongoing process instance L_t.

At the next monitoring instant $t+\Delta t$, expected value of target KPI $c_{t+\Delta t}$ is defined as the average value of KPI when the possible state at $t+\Delta t$ is maintained until completion. This value is predicted as follows. Suppose that an ongoing process instance is monitored in s_t. If the instance is executed continuously after t, $s_{t+\Delta t}$ can be one of the following two cases: 1) If any additional event has not yet been generated up to $t+\Delta t$, the current state s_t is maintained so that $s_{t+\Delta t}$ is equal to s_t. 2) If an additional event has been generated in the monitoring interval Δt, the probable next state $s_{t+\Delta t}$ is a union of s_t and the occurred event satisfying temporal sequences with events

in s_t. Therefore, $s_{t+\Delta t}$ becomes $s_t \cup \{e_{t+\Delta t}\}$ that $e_{t+\Delta t}$ satisfies $e_t \rightarrow e_{t+\Delta t}$ for all e_t in s_t. After taking these two cases into consideration, the ongoing instance with s_t can reach one state in the reachable state set $S_{t+\Delta t}$ of s_t, which can be described as follows:

$$S_{t+\Delta t} = \{s_t\} \cup \{s_{t+\Delta t} | s_{t+\Delta t} = s_t \cup \{e_{t+\Delta t}\} \textit{ that satisfies } e_t \rightarrow e_{t+\Delta t} \textit{ for } \forall e_t \in s_t\}$$

Then, $c_{t+\Delta t}$ can be estimated according to the weighted sum of reachabilities to states in $S_{t+\Delta t}$ and the average values of KPI from the states. The pseudo-code of the real-time KPI prediction algorithm is described as follows:

1: **PROCEDURE** *KPI_Prediction*(L_t)
2: $s_t \leftarrow$ *state* s_t *from observed event log* L_t;
3: $S_{t+\Delta t} \leftarrow$ *reachable states of* s_t *in reachability lattice*;
4: $c_t \leftarrow$ *average value of KPI in* s_t;
5: $c_{t+\Delta t} \leftarrow \sum_{i \in S_{t+\Delta t}} r_i c_i$, *where* r_i *is the probability on reachability from* s_t *to* s_i ($\in S_{t+\Delta t}$) *and* c_i *is the average value of KPI in* s_i;
6: **return** c_t *and* $c_{t+\Delta t}$;

5.2. Route prediction

Along with reachable orders, we can predict possible routes by periodically tracking the sequences of reachable states as an ongoing process instance progresses with additional occurrences of events. We defined a route as an ordered set of states and their reachable orders, excluding a recursive reachable order, based on the assumption that an additional event should be generated at each monitoring interval. From s_t, each route is retrieved by following the reachable orders to other states. Each state has an average KPI value when the current status of an event generation is maintained until completion. Then, we can track probable changes in the expected value of the target KPI after *t*, along with the sequences of reachable states in the route.

The purpose of tracking the expected value of KPI is to monitor risks that might be present in probable progress. In our research, real-time monitoring focuses on the expected value of the KPI when the ongoing process instance is completed.

If an acceptable threshold for KPI is given, practicable routes from s_t can be categorized into three cases: 1) $Route_{non\text{-}risk}$: the expected values of the target KPI are greater than the threshold. 2) $Route_{partial\text{-}risk}$: the expected values of the target KPI can be lower than the threshold in the course of the route, but they become higher than the threshold by the end of the route. 3) $Route_{risk}$: the expected values of the target KPI are lower than the threshold and they cannot recover by the end of the route.

If the goal of a monitoring policy is to maintain the expected value of the target KPI above the threshold, $Route_{partial\text{-}risk}$ and $Route_{risk}$ should be monitored continuously, and appropriate reactions should be carried out after each monitoring instant. On the contrary, $Route_{non\text{-}risk}$ does not have to be monitored continuously. In case of $Route_{partial\text{-}risk}$, additional resources might be allocated to subsequent activities after monitoring instants in order to promote transition to a reachable state and thereby produce a better value of KPI. Ideally, the midcourse state of $Route_{partial\text{-}risk}$ with a lower value of target KPI will progress to a reachable state so that it can produce a value that is better than the threshold. In the case of $Route_{risk}$, however, the expected value of the target KPI is difficult to achieve if it has already decreased because the route does not have any reachable state of

improvement beyond the threshold. Therefore, it may be better to cancel the ongoing process at the monitoring instant as soon as the expected value of the target KPI decreases. If the process is difficult to cancel, the process should be re-designed or a meta-process should be substituted in order to avoid the ongoing route.

6. System implementation with an example scenario

In this section, we present an example scenario to describe how the proposed approach can be applied to business process monitoring with a prototype system. Using the prototype system with the example scenario, the historical process log of a business process was analyzed to construct the reachability lattice, which was in turn used to monitor an ongoing instance of the process.

The example scenario was simplified from an existing real-world business process in a Korean telecommunications company which provides network service provisioning. To improve the quality of customer service, the telecom company contracts service level agreements for service delivery time. If the installation of network services is delayed more than three days from the contracted date, the company has to pay a penalty for the delay. For this reason, on-time network service delivery is critical in the service provisioning process, and the company monitors the KPI of "on-time delivery rate" for all service provisioning processes in the country to reduce penalties for overdue service delivery. Events related to "on-time delivery rate" were filtered as monitoring objects using the evaluation metric *eval*(*e*) described in Section 3. The related events include 'customer's schedule change' (e_1), 'work confirmation delay' (e_2), 'facility failure' (e_3), 'urgent processing' (e_4), 'change to experienced worker' (e_5), 'part shortage' (e_6), and 'rapid confirm' (e_7).

Figure 7 shows a business process diagram in BPMN (OMG, 2008) of the example scenario composed of eight activities and the corresponding events. The business process is composed of eight activities: 'Order Entry,' 'Telephone Customer,' 'Assign a Worker,' 'Install Normal Cable,' 'Report Normal Installation,' 'Install Fiber-optic Cable,' 'Report Special Installation,' and 'Confirm Service.' During execution of each activity, events can be generated as described in Figure 7. For example, the activity of 'Assign a Worker' can generate two filtered events e_1 and e_2, 'customer's schedule change' and 'work confirmation delay.'

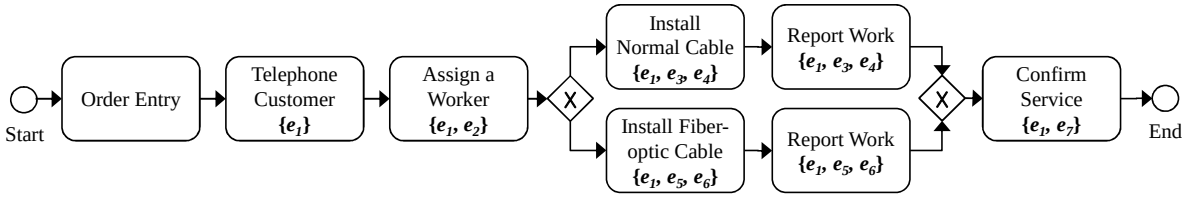


Figure 7: Example scenario of network service provisioning

The historical process log of the example scenario was designed according to the event history model described in Section 3. As shown in Figure 8, the event history data of the example scenario is loaded into our prototype system. The prototype system was developed on the open-source process mining framework, ProM (van der Aalst *et al*., 2007). In our prototype, historical process log data is first converted to an XML file, called MXML format, which can be used as an input file of the ProM framework.

Figure 8 also shows settings for event evaluation and monitoring scope of a reachability lattice. If users

give the rates of positive and negative process instances, denoted by r^+ and r^-, respectively, events are evaluated using the metric introduced in Section 3. Subsequently, users can filter the events for construction of the reachability lattice. The positive and negative process instances, C^+ and C^-, can be extracted from process instances I as follows:

$$I^+=\{i\in I \mid c(i)\geq c^{lb} \text{ where } c^{lb} \text{ is the minimum value satisfying } \Pr(c\geq c^{lb})\leq r^+\} \text{ and}$$

$$I^-=\{ i\in I \mid c(i)\leq c^{ub} \text{ where } c^{ub} \text{ is the maximum value satisfying } \Pr(c\leq c^{ub})\leq r^-\}.$$

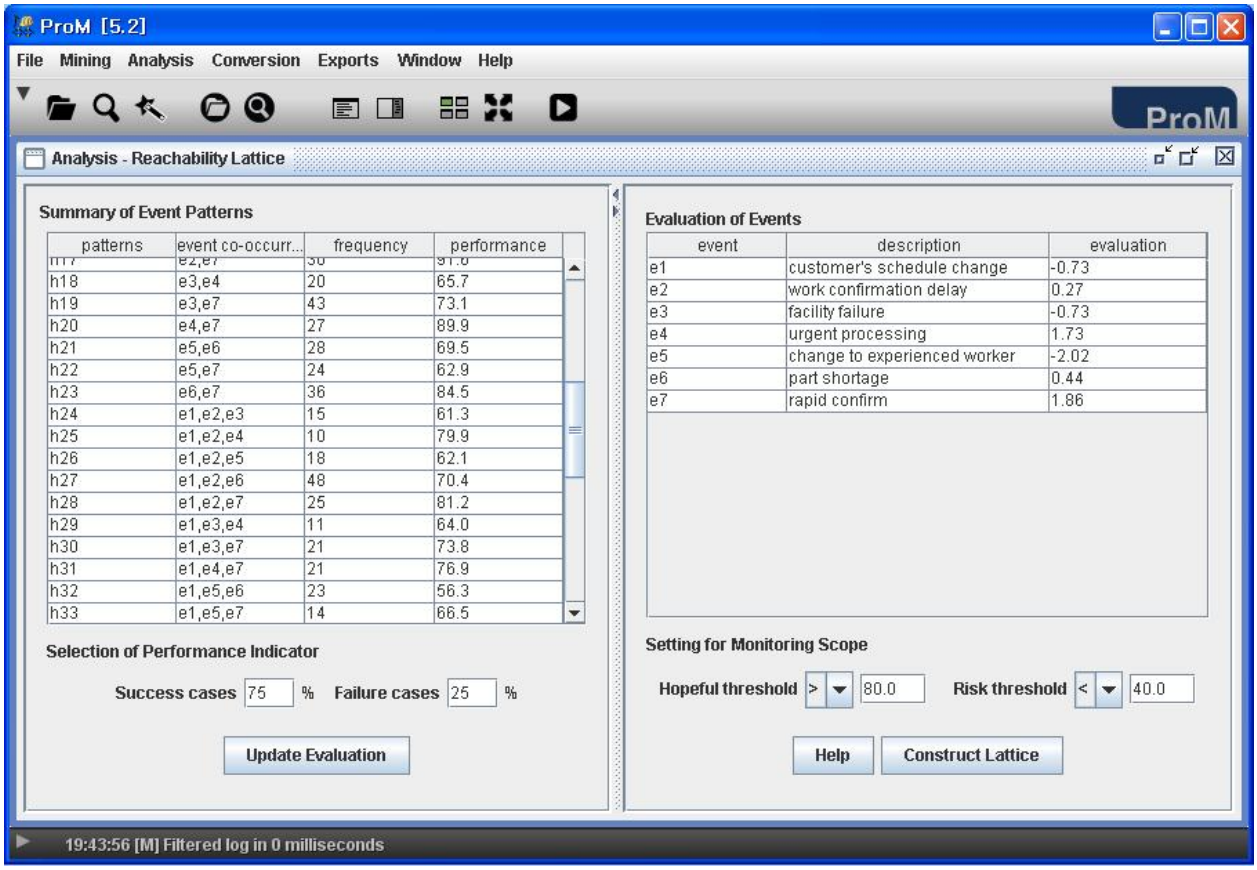
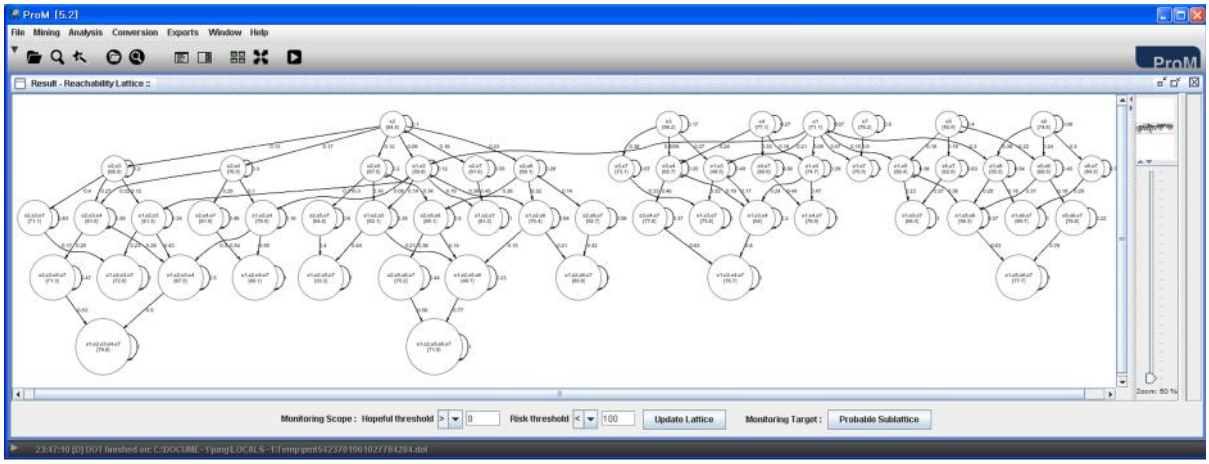
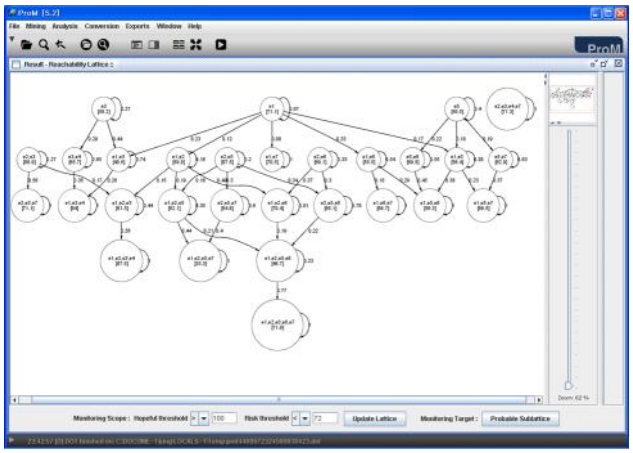


Figure 8: Setting for reachability lattice construction

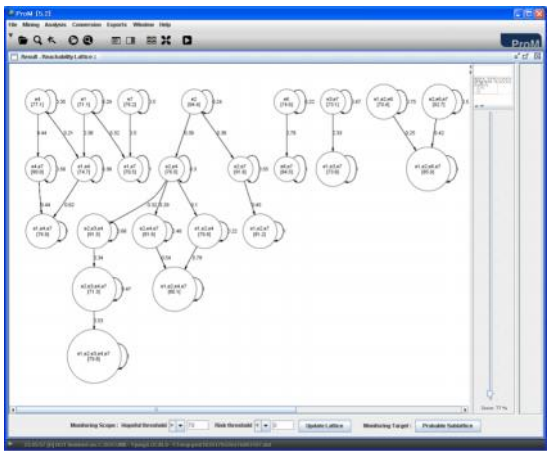
Figure 9 shows the reachability lattice constructed from the example history process log in Figure 8. The whole lattice shown in Figure 9 (a) can be reduced according to preferences in the scope of business process monitoring. For example, in the case of risk management, users may want to view event patterns with lower KPI values. On the contrary, if managers intend to improve performance, they are also interested in event occurrences with higher KPI values. Figures 9 (b) and (c) show sub-lattices which have been extracted from the whole lattice according to thresholds of monitoring scope. Such extraction of sub-lattices can help managers to recognize relations of events in a more comprehensible manner.



(a) whole reachability lattice



(b) risk lattice ($c<c^{ub}=72$)



(c) Success lattice ($c>c^{lb}=70$)

Figure 9: Visualization of reachability lattices: (a) whole reachability lattice, (b) risk lattice and (c) success lattices.

During process execution, the ongoing process instance is observed periodically during monitoring instants. Suppose that e_2 and e_3 were generated at monitoring instant t. Based on the event occurrence, real-time process monitoring is implemented using a part of the reachability lattice as shown in Figure 10. The figure shows all possible probable routes with reachable states for the observed ongoing process instance through completion.

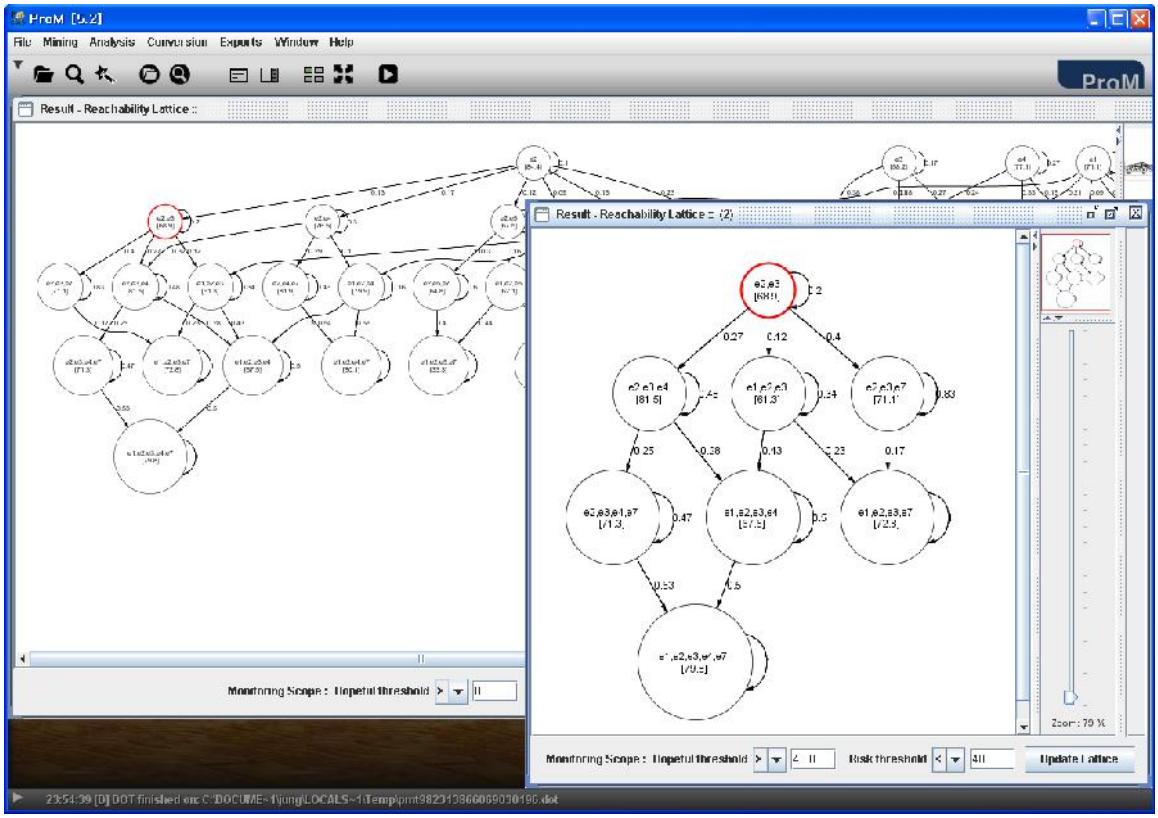


Figure 10: Visualization of possible states with predicted performances for a given event occurrence

According to the given event occurrence s_t, a reachable state set is retrieved as $S_{t+\Delta t}$={{e_2,e_3}, {e_2,e_3,e_4}, {e_1,e_2,e_3}, {e_2,e_3,e_7}}, since it is composed of s_t itself and states satisfying reachable orders; e_2 and e_3 could be generated earlier than could e_1, e_4 or e_7 in the process model. The next states of $S_{t+\Delta t}$ have average KPI values of 68.9, 81.5, 61.3, and 71.1, respectively. In the example scenario shown in Figure 7, the manager of service provisioning will consider performance improvements. Possible actions should be taken to trigger events of 'urgent processing' (e_1) or 'rapid confirm' (e_7), which can increase average performances up to 81.5 and 71.1, respectively.

Five probable routes from s_t are derived and summarized with the expected values of KPI in Table II. If an allowable threshold is defined as 75%, three $Route_{partial\text{-}risk}$ and two $Route_{risk}$ should be observed closely to assure higher customer satisfaction.

Table II. Route prediction with reachable orders and expected values of target KPI

Route category	Reachable order	Expected values of target KPI (%, * means lower than threshold)
$Route_{partial\text{-}risk}$	$\{e_2,e_3\} \rightarrow \{e_2,e_3,e_4\} \rightarrow \{e_2,e_3,e_4,e_7\} \rightarrow \{e_1,e_2,e_3,e_4,e_7\}$	68.9* → 81.5 → 71.3* → 79.8
	$\{e_2,e_3\} \rightarrow \{e_2,e_3,e_4\} \rightarrow \{e_1,e_2,e_3,e_4\} \rightarrow \{e_1,e_2,e_3,e_4,e_7\}$	68.9* → 81.5 → 67.5* → 79.8
	$\{e_2,e_3\} \rightarrow \{e_1,e_2,e_3\} \rightarrow \{e_1,e_2,e_3,e_4\} \rightarrow \{e_1,e_2,e_3,e_4,e_7\}$	68.9* → 61.3* → 67.5* → 79.8
$Route_{risk}$	$\{e_2,e_3\} \rightarrow \{e_2,e_3,e_7\} \rightarrow \{e_1,e_2,e_3,e_7\}$	68.9* → 71.1* → 72.8*
	$\{e_2,e_3\} \rightarrow \{e_1,e_2,e_3\} \rightarrow \{e_1,e_2,e_3,e_7\}$	68.9* → 61.3* → 72.8*

7. Comparison to other monitoring techniques

To realize business process monitoring, so-called business activity monitoring (BAM) practitioners and researchers have introduced a variety of techniques such as statistics, optimization and artificial intelligence. Despite those efforts, existing approaches have not satisfied industrial managers as effective decision support tools. One of the reasons is that most of the techniques have focused on discovering the best solutions, rather than presenting easy and comprehensible visualization of processes and outcomes. In BAM, visualization of probable situations has great significance in final decisions, because business environments are quite risky and dynamic. Moreover, if it is possible to illustrate probable situations to business managers based on the real-time progress of key business processes and historical data, such comprehensive analysis is helpful for making decisions on business problems.

In addition to FCA, there is another graphical and mathematical technique for business monitoring known as Petri nets. In the research domain of business process modeling and analysis, Petri nets such as WF-nets are often used to design behavior models of business processes (van der Aalst and van Hee, 2004) and to analyze business log data for business process mining and monitoring (van der Aalst *et al.*, 2007). Although both Petri nets and FCA are graphical tools that provide mathematical formalisms, they each have a slightly different focus. Petri nets propose effective mathematical analytics to design and analyze behavior models of business cases, while FCA supports the simplification and visualization of relations among cases in hierarchical graphical models. For this reason, Petri nets are often used to extract business process models from historical business cases (Alves de Medeiros *et al*., 2007); however, they have actually been applied to business monitoring in relatively few studies. For instance, Du *et al*. (2009) proposed a graphical model for defining and monitoring the behavior of e-commerce workflow. Cicirelli *et al*. (2010) suggested the dynamic reconfiguration of service processes according to real-time progress. A real-time scheduling method was proposed by Julia *et al*. (2008), which aimed to allocate resources to obtain acceptable scenarios according to specific sequences of activities.

Different from business process miners who aim at discovering behavioral models, our research assumes known behavioral models, which may be represented by Petri nets, EPC (Event-driven Process Chain) models, or UML activity diagrams. As a result, the goal of our research is to provide the simplified and comprehensible visualization of probable situations along with the expected values of target performance indicators, rather than to analyze business process models. For these purposes, FCA is a strong tool compared to other graphical tools and their many applications, as introduced in Section 2.2.

To analyze historical business data and support decision-making for the purpose of improving the performance of business processes, stochastic, optimization, and data mining techniques have also been adopted in business process monitoring. For example, some researchers have proposed stochastic prediction techniques for decision-making and optimization of resource allocation to ensure high performances of business processes (Ha *et al.*, 2006; Zhou and Chen, 2003). These have aimed at determining optimal or heuristic solutions, and they suggest predicted or simulated outcomes. Such optimization-based approaches are more helpful for business process improvements and decision support on numerical analysis. Nevertheless, to integrate them into real-time monitoring environments, optimization should be conducted iteratively at subdivided monitoring intervals, even though the dynamic environment of business process execution can be modeled in mathematical

programming. In contrast, our FCA-based proposal focuses on indirectly supporting the decision-making of industrial managers by visualizing all probable scenarios in progress, rather than finding optimal solutions. Meanwhile, data mining techniques like hierarchical clustering can also be used to generate hierarchy-representing knowledge in large cases. Nevertheless, FCA is more helpful than other hierarchical agglomerative clustering methods because of its traceability (Cimiano *et al*., 2005). FCA not only generates specific clusters and routes, but it also provides an intentional description and thereby contributes to improved intuition among users.

8. Conclusions

In this paper, we proposed a novel approach for real-time monitoring of business processes through the application of FCA. FCA was introduced to investigate historical process logs and to visualize relations among event patterns. Furthermore, a concept lattice, which is often used in FCA, was extended into a reachability lattice to overcome limitations in existing process monitoring methods: the conventional concept lattice does not consider the possible flows in process models, nor does it contain the notion of probabilities of transitions between event states. Thus the construction of reachability lattices and their application to real-time process monitoring were formulated. A prototype system was implemented on the ProM framework and used to illustrate a simplified real-world example scenario.

One of the main contributions of this paper lies in the development of a real-time monitoring approach of ongoing processes. While existing monitoring methods evaluate final outcomes upon process completion, the proposed method can predict final results during process execution. By monitoring an event as the occurrence of a significant incident during process execution, effective prediction of process outcomes can be achieved. We also provide a new knowledge representation scheme, called a reachability lattice, to better visualize probable routes in event co-occurrences during progress of business processes. A reachability lattice is differentiated from a conventional concept lattice as follows. First, reachability lattices represent the constraints of event occurrences, by removing unallowable sequences of event occurrences of a process model. The second difference is the practical utilization of a reachability lattice. Whereas the conventional use of concept lattices focuses on matching similar cases, reachability lattices can be used to predict expected values of KPI after considering historical cases with event occurrences in ongoing business processes. In other words, the reachability lattice determines the hierarchical structure of event occurrences to represent knowledge extracted from historical logs, and hence the reachability lattice enhances the efficient investigation of historical logs by visualizing them and focusing only on probable event states in real-time progress. Overall, the proposed method is significant for visualizing highly probable routes of generated events and for predicting expected performances of ongoing business processes in order to support real-time decision-making.

There remain several issues to be addressed in further research. The effective selection of events is an important issue, as the performance of real-time monitoring is greatly dependent on the events in a reachability lattice. Another issue is how to handle any unobserved event occurrences. Some significant incidents may occur even though they cannot be identified in the historical process logs on which our proposed prediction method is based.

References

Abad-Grau, M.M. and Arias-Aranda, D. (2006), "Operations strategy and flexibility: modeling with Bayesian classifiers", *Industrial Management & Data Systems*, Vol. 106 No. 4, pp. 460-484.

Alves de Medeiros, A.K., Pedrinaci, C., van der Aalst, W.M.P., Domingue, J., Song, M., Rozinat, A, Norton, B. and Cabral, L. (2007), "An outlook on semantic business process mining and monitoring", *Lecture Notes in Computer Science*, Vol. 4806, pp. 1244–1255.

Arévalo, G., Ducasse, S., Gordillo, S. and Nierstrasz, O. (2010), "Generating a catalog of unanticipated schemas in class hierarchies using Formal Concept Analysis", *Information and Software Technology*, Vol. 52 No. 11, pp. 1167-1187.

Beckett, A.J., Wainwright, C.E.R. and Bance, D. (2000), "Implementing an industrial continuous improvement system: a knowledge management case study", *Industrial Management & Data Systems*, Vol. 100 No. 7, pp. 330-338.

Beydoun, G. (2009), "Formal concept analysis for an e-learning semantic web", *Expert Systems with Applications*, Vol. 36 No. 8, pp. 10952-10961.

Bose, R. (2006), "Understanding management data systems for enterprise performance management", *Industrial Management & Data Systems*, Vol. 106 No. 1, pp. 43-59.

Boucher-Ryan, du P. and Bridge, D. (2006), "Collaborative Recommending using Formal Concept Analysis", *Knowledge-Based Systems*, Vol. 19 No. 5, pp. 309–315.

Buytendijk, F. and Flint, D. (2002), "How BAM can turn a business into a real-time enterprise", *Gartner Research*, AV-15-4650.

Carpineto, C. and Romano, G. (2004), *Concept Data Analysis: Theory and Applications*, Wiley, Hoboken, NJ.

Cicirelli, F., Furfaro, A. and Nigro, L. (2010), "A service-based architecture for dynamically reconfigurable workflows", *The Journal of Systems and Software*, Vol. 83 No. 7, pp. 1148-1164.

Cimiano, P., Hotho, A. and Staab, S. (2005), "Learning concept hierarchies from text corpora using formal concept analysis", *Journal of Artificial Intelligence Research*, Vol. 24 No. 1, pp. 305-339.

Curtis, B., Seshagiri, G.V., Reifer, D., Hirmanpour, I. and Keeni, G. (2008), "The cases for quantitative process management", *IEEE Software*, Vol. 25 No. 3, pp. 24-28.

Díaz-Agudo, B. and González-Calero, P.A. (2001), "Formal concept analysis as a support technique for CBR", *Knowledge-Based Systems*, Vol. 14 No. 3(4), pp. 163-171.

Du, Y.Y., Jiang, C.J., Zhou, M.C. and Fu, Y. (2009), "Modeling and monitoring of E-commerce workflows", *Information Sciences*, Vol. 179 No. 7, pp. 995-1006.

Dumas, M., van der Aalst, W.M.P., and ter Hofstede, A.H.M. (2005) *Process-Aware Information Systems: Bridging People and Software Through Process Technology*, Wiley & Son, Hoboken, NJ.

anter B. and Wille, R. (1999), *Formal Concept Analysis: Mathematical Foundations*, Springer, Berlin, Germany, 1999.

Grigori, D., Casati, F., Dayal, U. and Shan, M-C. (2001), "Improving business process quality through exception understanding, prediction, and prevention", in *Proceedings of the 27th International Conferences on Very Large Data Bases, in Roma, Italy, 2001*, Morgan Kaufmann Publishers Inc., San Francisco, pp.159-168.

Grigori, D., Casati, F., Castellanos, M., Dayal, U., Sayal, M. and Shan, M-C. (2004), "Business Process Intelligence", *Computers in Industry*, Vol. 53 No. 3, pp. 321-343.

Ha, B.-H., Bae, J., Park, Y,-T. and Kang, S.-H. (2006), "Development of process execution rules for workload balancing on agents", Data and Knowledge Engineering, Vol. 56, No. 1, pp. 64-84.

Hammer, M. and Champy, J. (1994), *Reengineering the corporation – a manifesto for business revolution*, Nicholas Brealey Publishing, London, UK.

Hernández, C., Prieto, F., Laguna, M.A. and Crespo, Y. (2002), "Formal Concept Analysis Support for Conceptual Abstraction in Database Reengineering", paper presented at the *Database Maintenance and Reengineering Workshop, Montreal, Canada, 2002*.

Ho, G.T.S., Lau, H.C.W., Chung, S.H., Fung, R.Y.K., Chan, T.M. and Lee, C.K.M. (2008), "Fuzzy rule sets for enhancing performance in a supply chain network", *Industrial Management & Data Systems*, Vol. 108 No. 7, pp. 947-972.

Hsieh, T.-C. and Wang, T.-I (2010), "A mining-based approach on discovering courses pattern for constructing suitable learning path", *Expert Systems with Applications*, Vol. 37 No. 6, pp. 4156-4167.

Jiang, G., Ogasawara, K., Endoh, A. and Sakurai, T. (2003), "Context-based ontology building support in clinical domains using formal concept analysis", *International Journal of Medical Informatics*, Vol. 71 No. 1, pp. 71-81.

Julia, S., Oliveira, F.F. and Valette, R. (2008), "Real time scheduling of Workflow Management Systems based on a p-time Petri net model with hybrid resources", *Simulation Modelling Practice and Theory*, Vol. 16 No. 4, pp. 462-482.

Kang, B., Cho, N.W. and Kang, S.-H. (2009a), "Real-time risk measurement for business activity monitoring (BAM)", *International Journal of Innovative Computing, Information and Control*, Vol. 5 No. 11(A), pp. 3647-3657.

Kang, B., Lee, S.K., Min, Y., Kang, S.-H. and Cho, N.W. (2009b), "Real-time Process Quality Control for Business Activity Monitoring", in *Proceedings of the 2009 International Conference on Computational Science and Its Applications, in Yongin, Korea, 2009*, IEEE Computer Society, pp. 237-242.

Keung, P. and Kawalek, P. (1997), "Goal-based business process models: Creation and evaluation", *Business Process Management Journal*, Vol. 3 No. 1, pp. 17–38.

Lam, C.Y., Ip, W.H. and Lau, C.W. (2009), "A business process activity model and performance measurement using a time series ARIMA intervention analysis", *Expert Systems with Applications*, Vol. 36 No. 3(2), pp. 6986-6994.

Luckham, D.C. (2002), *The Power of Events*, Person Education, Boston.

Lundberg, A. (2006), "Leverage Complex Event Processing to Improve Operational Performance", *Business Intelligence*, Vol. 11 No. 1, pp. 55-65.

Mendling, J., Verbeek, H.M.W., van Dongen, B.F., Van der Aalst, W.M.P. and Neumann, G. (2008) "Detection and prediction of errors in EPCs of the SAP reference model", *Data & Knowledge Engineering*, Vol. 64 No. 1, pp. 312-329.

OMG (2008), *Business Process Modeling Notation (BPMN) Version 1.1*, OMG Specification, Document Number: formal/2008-01-17, available at: http://www.omg.org/spec/BPMN/1.1/PDF.

Park, Y. (2000), “Software retrieval by samples using concept analysis”, *The Journal of Systems and Software*, Vol. 54 No. 3, pp. 179-183.

Peddabachigari, S., Abraham, A., Grosan C. and Thomas, J. (2007), “Modeling intrusion detection system using hybrid intelligent systems”, *Journal of Network and Computer Applications*, Vol.30 No.1, pp.114-132.

Priss, U. (2006), “Formal concept analysis in information science”, in Cronin, B. (Ed.), *Annual Review of Information Science and Technology*, Vol. 40, American Society for Information Science and Technology, Silver Spring, MD, pp. 521-43.

Solesvik, M.Z., Encheva S. (2010), “Partner selection for interfirm collaboration in ship design”, *Industrial Management & Data Systems*, Vol. 110 No. 5, pp. 701-717.

van der Aalst, W. and van Hee, K. (2004), Workflow Management: Models, Methods, and Systems, MIT Press, Cambridge, MA.

van der Aalst, W.M.P., Reijers, H.A., Weijters, A.J.M.M., van Dongen, B.F., Alves de Medeiros, A.K., Song, M., and Verbeek, H.M.W. (2007), "Business Process Mining: An Industrial Application", *Information Systems*, Vol. 32, No. 5, pp. 713-732.

Wang, D. and Romagnoli, J.A. (2005), “Robust multi-scale principal components analysis with applications to process monitoring”, *Journal of Process Control*, Vol. 15 No. 8, pp. 869-882.

Widodo, A. and Yang, B-S. (2007), “Support vector machine in machine condition monitoring and fault diagnosis”, *Mechanical Systems and Signal Processing*, Vol. 21 No. 6, pp. 2560-2574.

Wille, R. (1982), “Restructuring lattice theory: an approach based on hierarchies of concepts”, in Rival, I. (Ed.), *Ordered Sets*, Reidel, Dordrecht, pp. 445-70.

Zhou, Y. and Chen, Y. (2003), "Project-oriented business process performance optimization," in *Proceedings of Systems, Man and Cybernetics, 2003. IEEE International Conference on*, Vol. 5, pp. 4079- 4084.