PROCEEDINGS OF SCIENCE

A Real-time Performance Monitoring Framework Based on ACC Model in Collaboration Business Process Environment

Junbao Zhang[1]
Computer Science and Technology of Donghua University
Shanghai, 201620, China
E-mail: junbaozb0451@163.com

Guohua Liu[2]
Computer Science and Technology of Donghua University
Shanghai,201620, China
E-mail: ghliu@dhu.edu.cn

Lili Huang
Computer Science and Technology of Donghua University
Shanghai,201620, China
E-mail: 11772117@qq.com

In order to improve the efficiency of building a real-time performance monitoring system in collaboration business process environment, a framework integrated with the collaboration business process system based on artifact-centric collaboration (ACC) model is prompted. With the view information of ACC model, a rule-based method is proposed to collect real-time data and the monitoring system is abstracted as a monitoring model. After the monitoring designer defines a monitoring model according to monitoring requirements, the monitoring model is automatically transformed to the technical specification of the monitoring system. With this framework, the monitor designer can focus on designing the monitoring model other than details of the technical specification. There is no need to negotiate with collaborative partner from design to deployment without risk of privacy leakage. This approach both increases the agility of the monitoring system and reduces the difficulty in building a monitoring system.

CENet2017
22-23 July, 2017
Shanghai, China

[1]Speaker
[2]Corresponding Author

PoS(CENet2017)058

1. Introduction

It is a mega trend for modern enterprises to fulfill the whole business service through collaboration of different organizations' private business processes. The collaboration businesses are ubiquitous. For the purpose of better deciding the process correctively, the enterprise manager needs to collect real-time business data distributed across different organizations to monitor some relevant KPIs [1,2].

Despite significant amount of work in terms of this topic, it is hard and laborious to build an effective and efficient performance monitoring system. At present, most monitoring system is built independently in a hard-coded manner; meanwhile, changes of either monitoring requirements or business process would result in rebuilding of the monitoring system and need a large amount of time and labor. Obviously, there is a privacy problem in the collaboration. To our knowledge, the monitoring system is built on the basis of mutual trust and the monitoring protocol is established through negotiation. Changes of the protocol need to rebuild the system.

To address these problems, we suggest a real-time performance monitoring framework based on ACC model [3,4]. ACC model is an artifact-centric view-based [5] collaboration business process model. The share artifact in ACC model defines the share business data that the participants will read or operate. It has an information model that describes the business data type and a lifecycle model that describes the key stages of the artifact while collaboration process is evolving. More importantly, the share artifact has unified universal semantics that all relevant process participants have agreed. The view information in ACC model imposes restrictions on previous artifact to confine which part a process participant can read or operate. This view can avoid spreading all business data across all process participants and be capable of protecting their privacy. Taking previous property into consideration, we take share artifact as monitoring contract between collaboration business process system and our monitoring framework, and use the view information to confine the real-time data that the framework can collect.

Our work has been greatly inspired by research of Liu R, et al [6]. Based on the framework, the monitor designer explicitly defines the monitoring model based on the share artifact information that he/she can reach according to the view information with respect to him/her. The monitoring model precisely defines the type of the required real-time data, the time that the real-time data should be collected and the way that these data will be computed according to definitions of metrics and KPIs. Differently, (1) the time of collecting real-time data is determined by observing changes of the share artifact instead of inspecting the process system events because the process participants don't know exactly process event in collaboration business process circumstances. (2) the monitoring model is defined based on view confined share artifact as oppose to the business artifact so that the privacy can be preserved. Compared to other works, our framework features below advantages: (1) the generated monitoring system has high level of agility; (2) the process partner is not involved in building the monitoring system; (3) the privacy problem is considered in building the monitoring system.

Generally, a performance monitoring system mainly contains four parts [7]: (1) a real-time data capture layer to collect all required real-time data; (2) a data processing layer to compute the metrics and KPIs and generate the monitoring system event; (3) an event dispatch layer to dispatch the generated event and; (4) a display layer to represent metrics and KPIs in a

dashboard. Our work focuses on the real-time data capture layer and part of data processing layer that compute the metrics and KPIs. The paper is organized as follows. Section 2 reviews the relate works, Section 3 introduces the method of capturing the real-time data in detail. In Section 4, we formally define the monitoring model. Section 5 presents the architecture of the monitoring framework. Section 6 presents discussions and evaluation of our approach. Section 7 show the experiments and Section 8 makes the conclusion.

2.Relate Works

As the globalization and conglomeration of large enterprises have resulted in greatly increased complexity of business process, it demands more effective methods of real-time performance monitoring in collaboration [8]. Many researchers have noticed this issue and carried out many valuable works in this regard.

B Wetzstein, et al proposed a process monitoring instrument to inspect the process system event and collect required real-time data [9]. The data type and the time of collecting real-time data were settled by monitoring the contract as signed by multipartities through negotiation. The collected data are delivered to the monitoring system and then transformed to relevant KPI. A Baouab, et al prompted a component named EFM (External Flow Monitoring) as deployed to every process partner [10]. EFM component was responsible for sending and receiving real-time data. The delivered real-time data were divided into three varieties: orchestration data, monitoring purpose data and exception data. The type and semantics of real-time data were settled by negotiation, either. But it didn't mention how to deal with these real-time data. M Comuzzi, et al used PBWD (Product-Based Workflow Design) methodology to define executable monitoring process [11,12]. To get an executable monitoring process, the monitoring designer defined PDM (Product Data Model), which described types of real-time data and operations of getting these data. Every PDM had three non-functional dimensions, expense, quality and availability respectively. One or more PDM instance can generate a new PDM instance. For target PDM instance, there may have multiple generation graphs. The most preferable one was determined by Ant Colony algorithm according to previous non-functional dimension constraints. Finally, with the determined graph as input, the executable workflow monitoring process based on web service was generated. Here, the real-time data were collected by monitoring service provided by the collaboration partner.

Similar to previous work [9-12], our work focuses on monitoring business related metrics and KPIs. Differently, previous work didn't explicitly simulate the monitoring model and interactions between the monitored side and the monitor side in hard-coded manner. Furthermore, previous work didn't take into consideration of the privacy safety. There are other works to monitor SLA (Service Layer Agreement) related metrics and KPIs [2,13], which are not our research point.

3. Real-time Data Collection

In real-time data capture layer, the majority previous researches applied "intrusive" way to collect the real-time data. And it is laborious and time consuming in implementation. Here we propose a "semi-intrusive" way to collect the real-time data: 1) a rule is proposed to describe what kind of data should be collected and when to collect these required data; 2) view information of ACC model is used to define and dispatch the rules so that negotiation is

escaped; 3) a monitoring instrument is added at the monitored side to collect and deliver the real-time data according to the rules, verify that the owner of rules may collect these real-time data according to the view information of ACC model so that interactions between two sides are "soft-coded". For the convenience of description, the real-time data collection is divided into following three parts.

3.1Requirement Definition Phase

In the phase of requirement definition, the type of real-time data and the time of collecting real-time data are defined by the event and the event rule respectively.

Definition 1 (Event Type). Event type E = (T, D), in which, T and D refer to the set of name-value pair attributes. T refers to the set of event related domain independent description attributes, for example, ID, timestamp, name etc. D refers to the set of domain dependent attributes, as defined by monitoring designer. There is an injection from D to the attribute set of share artifact.

Given an attribute att, which is called the simple attribute if the type of att is simple type (integer, string etc). att called composite attribute if the type of att is composite type which is the composition of simple type and composite type. $\forall d \in T$, d refers to the simple attribute and $\forall d \in D$, d refers to the simple attribute or composite attribute. Note that all events must conform to some defined event type. Event in this article contains both the description information and the data information. If not specifically pointed out, the event mentioned below is the newly introduced event. Here, the event type, the event instance and the instance of event instance i.e. concrete event are distinguished. If it is distinguishable from context, all of them are referred to as the event for short.

Definition 2 (Event Rule Type). The event rule type is a rule of form $P_o \times P_n \to E$, in which P_o and P_n refer to the multivariate quantifier free predicate formula over view confined attributes of share artifact, and E is the event type. P_o describes the state of share artifact before some services update the artifact and P_n describes the state of share artifact after the same services update the artifact. $P_o \times P_n$ is used to estimate the current state of a process instance and E is used to define the required real-time data type. The instance of an event rule type is an event rule.

3.2Requirement Dispatch Phase

The event rules defined by the monitoring designer should be dispatched to the proper monitoring instrument so that the required real-time data can be collected and delivered to the monitoring system. Algorithm 1 is presented to dispatch these event rules.

Algorithm 1. Event Rule Dispatch

Input：view confined ACC model share(m), set of event rules ER.

Initialize: $S_1, S_2, \ldots, S_n = \varnothing$.

1. For each event rule in ER

2. If there is a service, belonging to the role l_i (1≤i≤n) in share(m), operate on some attribute available in the event rule.

3. Add the event rule to S_i.

4. Send none empty S_i ($1 \le i \le n$) to the monitoring instrument MI_i.

In algorithm 1, MI_i denotes the monitoring instrument deployed to the local ACP system of l_i. If there is a service belonging to role l_i, operate on some attribute appears in the event rule, the event rule may be evaluated to be true, thus the event rule should be delivered to MI_i.

3.3 Real-time Data Capture Phase

When the event rules have been delivered to the monitoring instrument, the monitoring instrument does two kinds of work in real-time data capture phase: 1) to verify the legality of the received event rules; 2) to collect and deliver the real-time data.

Verify the legality of the received event rules. For every received event rule, if there's some attribute in the event rule that the monitor designer has no right to read according to the view information, this event rule is illegal and a warn should be reported. In this way, the hostile attack can be detected and the private information can be preserved.

Collecting and delivering real-time data. The monitoring instrument inspects all updates of share artifact and evaluates the two predicates of the event rule. If all predicates are satisfied, the required real-time data are collected and packaged as the event and delivered to the monitoring system.

4.Monitoring Model

A monitoring model formally depicts all required elements of building an executable monitoring system. It includes the description information of monitoring model, the set of events, the set of event rules, the metrics and KPIs. Intuitively, the model precisely defines the type of the required real-time data, the time that the real-time data should be collected and the way that these data should be computed. The formally defined monitoring model is presented below.

Definition 3 (Monitoring Model Type). A monitoring model type MM is a 6-tuple, MM= (ID, name, EVENT, EVENTRULE, METRIC, K), where (1) ID is the identifier. (2) name is the name of MM. (3) EVENT is set of event instance. (4) EVNETRULE is set of event rules. There is a one-one mapping from EVENT to EVENTRULE. (5) METRIC is set of metric. The type of metric Metric = (ID, name, D_{Metric}, mp, e, f_{Metric}) is a 6-tuple. ID is the identifier; name is the name of the Metric; D_{Metric} is the set of attributes, $\forall d \in D_{Metric}$, d is the simple attribute; mp is the value of the Metric, mp is a simple numerical attribute; e is an event instance, $e \in EVENT$; f_{Metric} is a compute function from e to Metric. An instance of Metric is metric [14]. (6) K is the set of KPIs. The KPI type Kpi = (ID, name, D_{Kpi}, kp, Metric, f_{Kpi}) is a 6-tuple. ID is the identifier, name is the name of Kpi, D_{Kpi} is the set of dimension attributes, $D_{Kpi} \subseteq Metric.D_{Metric}$, kp is the value of the Kpi that is numerical, Metric is a metric type and f_{Kpi} is an aggregate function over metrics of type Metric and compute the value of kp, the instance of Kpi is KPI. The instance of MM is the monitoring model.

5. Architecture of Framework

The architecture of our framework is depicted as Fig. 1.

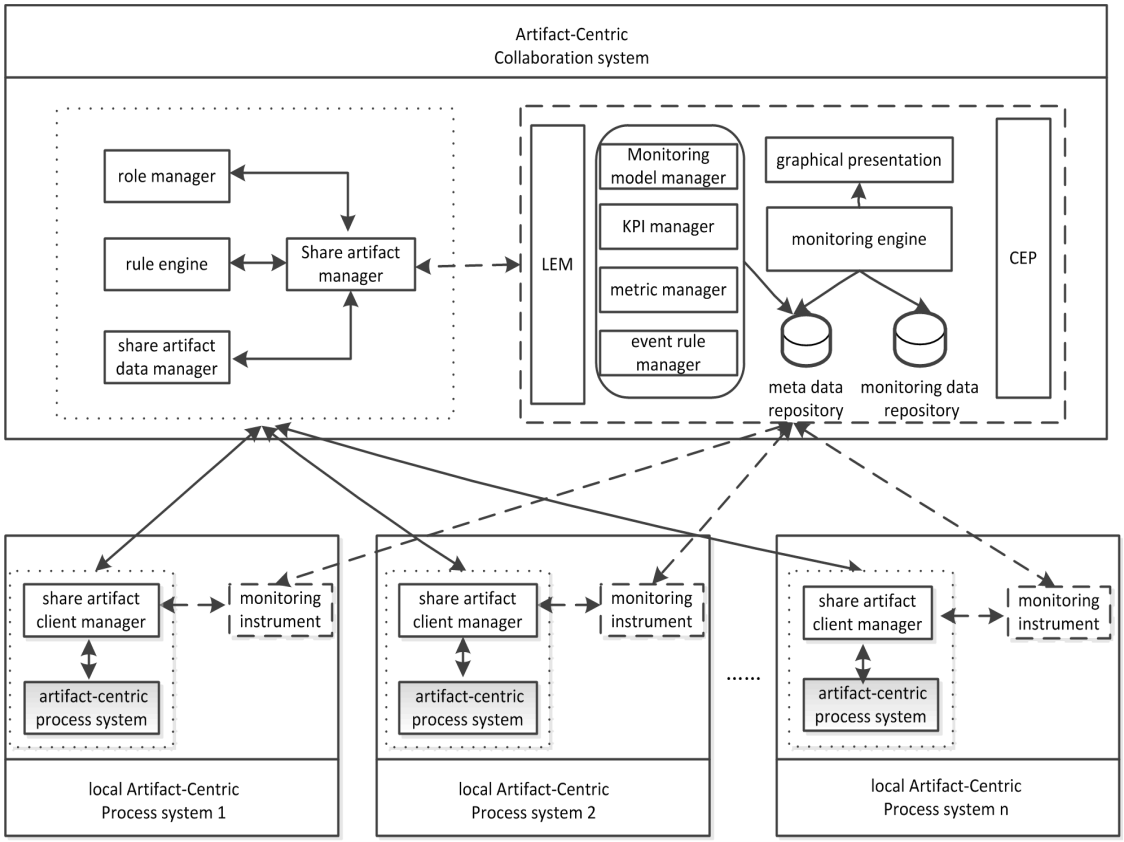

Figure 1:the Architecture of the Framework

In Fig.1, the dashed rectangle wrapped components are proposed in our framework, and the dotted rectangle wrapped components are proposed in ACC system; the arrows lined with the solid line present interactions between components for process management purpose; and the arrows lined with the dashed line present interactions between components for performance monitoring purpose. Among the new introduced components, LEM (Lightweight Event Module) is responsible for dispatching the defined event rules; the monitoring model manager, KPI manager, metric manager and event rule manager is used to manage the monitoring model and its related elements; the monitoring engine is the heart of the framework which is responsible for coordinating every component's work, initiating and deploying the monitoring models i.e. read monitoring meta data from meta data repository, call LEM to dispatch the event rules by using algorithm 1, transform the monitoring model to executable technical specification and receive the real-time data and call proper processing function, similar to Liu's work, the transformation details is not presented; the meta data repository and monitoring data repository storage monitoring model related meta data and monitoring data respectively; the graphical presentation display metrics and KPIs in graphic. CEP (Complex Event Processing [15]) is responsible for handling exceptions and feedback control, which is not introduced in this article; monitoring instrument is introduced in Section 3.3. The process of building a monitoring system is depicted as Fig. 2.

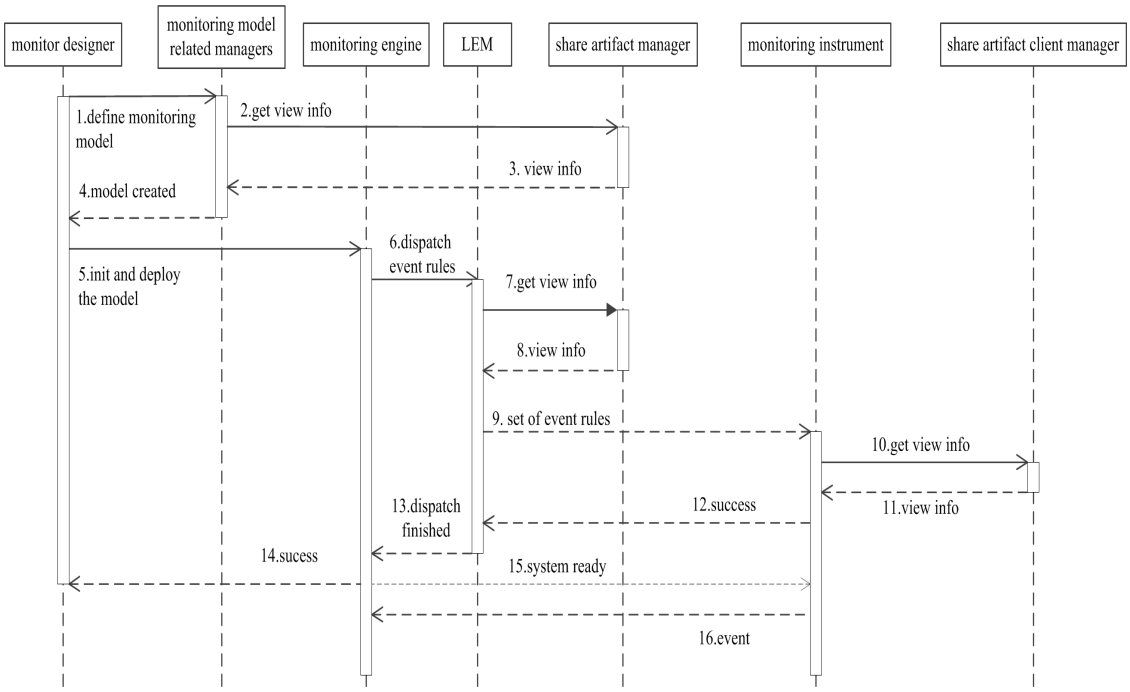

Figure 2: Process of Building A Monitoring System

Note that there are three requests of view information. The first is used to define the event rules in the monitoring model; the second is used to dispatch the event rules; the third is used to confirm that the legality of the event rules. In the process, we can see that building a monitoring system is simplified as defining a monitoring model. If the monitoring system needs to change derives from collaboration or the monitoring requirement change, just modify the monitoring model and refresh the process.

6. Discussions and Evaluation

Feasibility analysis: the prompted framework in this article adopts a model-driven paradigm to build an executable monitoring system. It has been proved that it is feasible to transform a formally defined monitoring model to executable technical specification [6]. Here we analyze the feasibility of our real-time data collection approach. Firstly, updates of the share artifact can be inspected by the database layer listening or the application layer listening technology. In this sense, the real-time data can be absorbed at local ACP system side by the monitoring instrument. Secondly, we applied the event rules in real artifact-centric business process system to collect the real-time data, from the experimental results, the required real-time data can be collected correctly. Finally, by using the network data transmission technology, like JMS, real-time data can be delivered to the generated monitoring system. By conclusion, the prompted framework is feasible.

In the monitoring environment of collaboration business process, the majroity works didn't concern the modeling monitoring model explicitly and technical specification of the model is manually specified while featuring high skill and time consumption. Like Liu's work, explicit model monitoring model has the following benefits.

(1) Simplicity. Key elements of monitoring model simply rely on share artifact. D part of event, variables of event rules, attributes of metric and KPI are all based on the share artifact.

Furthermore, the share artifact has universal unified semantics, and leverages the understandability of metrics and KPIs for monitoring designer.

(2) Configurable. The share artifact with proper granularity makes metrics very flexible and configurable for KPIs in the monitoring model. When the change happens in either the monitoring requirement or the collaboration business process, the metrics and KPIs are re-configured and redeployed easily.

Differently, Liu's work has to capture the process system event to collect real-time data, which become complex in the collaboration business process environment and likely to result in privacy leakage. We use view information ACC model and utilize event rule to collect real-time data. This approach has following benefits.

(1) Security. In ACC model, different roles have different views. With the view information, there is no need of negotiation to collect required real-time data without privacy leakage risk. Supposing that the monitoring designer has some role authority and has restricted view, the designer can't read private information. As the designer is working for some organization that takes part in the collaboration business process, it's reasonable to propose that the designer has some role authority. Furthermore, the security property makes the monitoring designer to enjoy the freedom of modifying the monitoring models and the agility of the monitoring system is leveraged.

(2) Effectiveness. The real-time data are collected by requirement and only the required data are obtained and delivered so that the expense of bandwidth is reduced. When there are multiple monitoring models, equivalent event rules can be detected and merged. In this way, the pressure of process server and monitoring instrument can be lowered.

In addition to these priorities, there is a drawback in our framework. As our framework needs to read the information of ACC model and tightly coupled with the ACC model.

7. Experiments

The model-driven paradigm of transforming a monitoring model to executable technical specification has been proved to be feasible [6]. We hereby verify the time effectiveness of collecting real-time data. As the network delay is inevitable and many optimization methods have been prompted, we focus on the time expense of evaluating the event rules. We thus implemented two realizations: the sequence traverse evaluation and the index traverse evaluation. The index traverse evaluation takes attribute of the event rules as index. The event rule is evaluated if there're some attribute values as changed. The environment of the experiment is windows7 64 bit operating system, core i5-4460 CPU, 12 GB main memory and 1 TB hard disk.

We simulated the update operation at random frequency between 0ms to 30ms and 0~N attributes are changed. The duration time from data generation time to data capture time is recorded while evaluating different M event rules as randomly generated. In order to reduce random error, final results are the average value of multiple experiments.

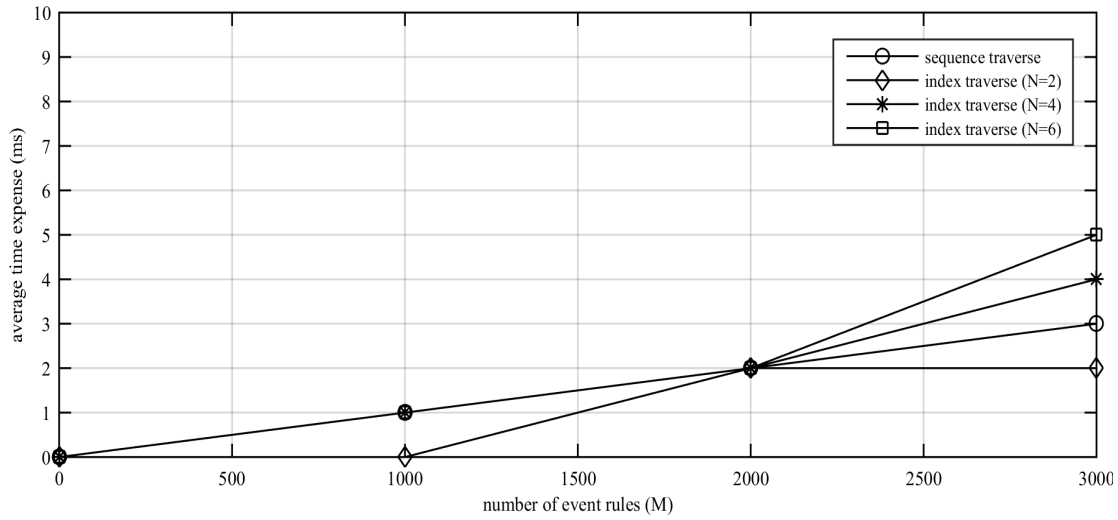

Figure 3: Average Time Expense

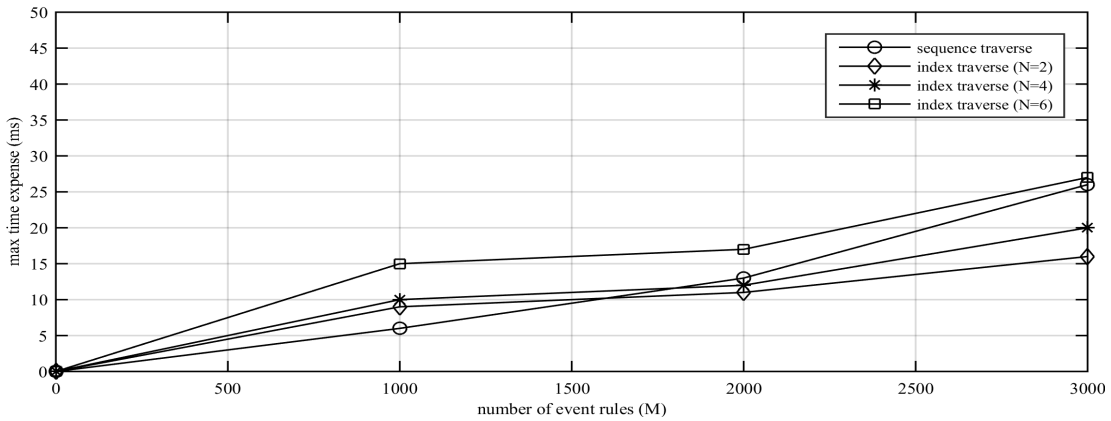

Figure 4: Max Time Expense

From Fig. 3 and Fig. 4, we can see that average expense time and the max expense time are growing while M increases and there are direct ratio relationships from the number of event rules to average time expense and from the number of event rules to the max time expense. Both the realizations are acceptable from the experimental result for both of them are applicable to human use.

8. Conclusion

In this paper, we proposed a real-time performance monitoring framework based on ACC model in collaboration business process environment. By using the view information of ACC model, the event rule is defined to collect real-time data and the process partner doesn't need to take part in the process of building a monitoring system without risk of privacy leakage. This framework adopts a model-driven paradigm to transform a monitoring model to executable monitoring system. With this framework, the procedure of building a monitoring system is dramatically simplified and the agility of the generated monitoring system is greatly increased. The prompted framework is only the preliminary research and more works are demanded to put this framework in a real application.

References

[1] C White. *Now is the right time for real-time BI*[J]. Information Management, 2004, 14(9): 47.

[2] A Keller, H Ludwig. *The WSLA framework: Specifying and monitoring service level agreements for web services*[J]. Journal of Network and Systems Management, 2003, 11(1): 57-81.

[3] S Yongchareon, K Ngamakeur, CF Liu, S Chaisiri, J Yu . *A workflow execution platform for collaborative artifact-centric business processes*[C]// Confederated International Conferences On the Move to Meaningful Internet Systems. Springer Berlin Heidelberg, 2014: 639-643.

[4] S Yongchareon, CF Liu, XH Zhao. *An artifact-centric view-based approach to modeling inter-organizational business processes*[C]//International Conference on Web Information Systems Engineering. Springer Berlin Heidelberg, 2011: 273-281.

[5] S Yongchareon, CF Liu. *A process view framework for artifact-centric business processes*[C]// Confederated International Conferences On the Move to Meaningful Internet Systems. Springer Berlin Heidelberg, 2010: 26-43.

[6] R Liu, R Vaculín, Z Shan, A Nigam, F Wu. *Business artifact-centric modeling for real-time performance monitoring*[C]//International Conference on Business Process Management. Springer Berlin Heidelberg, 2011: 265-280.

[7] B Gassman. *How the pieces in a BAM architecture work*. Technical report. Gartner Research, 2004.

[8] M Daneva, R Wieringa. *A requirements engineering framework for cross-organizational ERP systems* [J]. Requirements Engineering, 2006, 11(3):194-204.

[9] B Wetzstein, D Karastoyanova, O Kopp, F Leymann, D Zwink. *Cross-organizational process monitoring based on service choreographies*[C]// Proceedings of 25th ACM Symposium on Applied Computing. ACM New York, 2010:2485-2490.

[10] A Baouab, W Fdhila, O Perrin, C Godart. *Towards decentralized monitoring of supply chains*[C]// 20th International Conference on Web Services. IEEE, 2012: 600-607.

[11] M Comuzzi, I Vanderfeesten. *Product-based workflow design for monitoring of collaborative business processes*[C]// International Conference on Advanced Information Systems Engineering. Springer-Verlag, 2011:154-168.

[12] M Comuzzi, I Vanderfeesten, T Wang. *Optimized cross-organizational business process monitoring: design and enactment*[J]. Information Sciences, 2013, 244(7):107-118.

[13] A Bertolino, A Calabro, G Angelis. *Adaptive SLA monitoring of service choreographies enacted on the cloud*[C]//7th International Symposium on the Maintenance and Evolution of Service-Oriented and Cloud-Based Systems. IEEE, 2013:92-101.

[14] DW Hubbard. *How to measure anything: finding the value of intangibles in business* [M]. Jon Wiley & Sons Hoboken, 2007.

[15] A Bertolino, A Calabrò, F Lonetti, AD Marco, A Sabetta. *Towards a model-driven infrastructure for runtime monitoring*[C]//3th International Workshop on the Software Engineering for Resilient Systems, Geneva, Switzerland , 2011:130-144.