Process Data Store: A Real-time Data Store for Monitoring Business Processes

Josef Schiefer[1], Beate List[2], Robert M. Bruckner[2]

[1] IBM Watson Research Center
19 Skyline Drive
Hawthorne, NY 10532
josef.schiefer@us.ibm.com

[2] Institute of Software Technology
Vienna University of Technology
Favoritenstr. 9 / 188, A-1040 Vienna, Austria
{list,bruckner}@ifs.tuwien.ac.at

Abstract. With access to real-time information on critical performance indicators of business processes, managers and staff members can play a crucial role in improving the speed and effectiveness of an organization's business operations. While the investments in data warehouse technologies have resulted in considerable information processing efficiencies for the organizations, there is still a significant delay in the time required for mission critical information to be delivered in a form that is usable to managers and staff. In this paper we introduce an architecture for business process monitoring based on a process data store which is a data foundation for operational and tactical decision-making by providing real-time access to critical process performance indicators to improve the speed and effectiveness of workflows. The process data store allows to identify and react to exceptions or unusual events that happened in workflows by sending out notifications or by directly changing the current state of the workflow. Our proposed architecture allows to transform and integrate workflow events with minimal latency providing the data context against which the event data is used or analyzed.

1. Introduction

As businesses are being forced to become much more operationally efficient, enterprises are turning their attention toward implementing solutions for real-time business activity monitoring (BAM) [2]. These initiatives require enterprise data to be captured and analyzed in real-time from a wide variety of enterprise applications, operational data stores (ODS) and data warehouses. While traditional data integration approaches, built on top of core ETL (extraction, transformation, loading) solutions, are well suited for building historical data warehouses for strategic decision support initiatives, they do not go far enough toward handling the challenge of integrating data with minimal latency and implementing a closed loop for business processes.

Separated from operational systems, data warehouse and business intelligence applications are used for strategic planning and decision-making. As these applications have matured, it has become apparent that the information and analyses they provide are vital to tactical day-to-day decision-making, and many organizations can no longer operate their businesses effectively without them. Consequently, there is a trend toward integrating decision processing into the overall business process. Workflow audit trail information is a good example for this trend because it provides the most value to the users and process analysts if it is embedded into the decision-making processes of workflows and if it is available with minimal delays.

In this paper, we propose a process data store (PDS) which aims to support business processes with a near real-time business intelligence. The main data source is a workflow audit trail that is propagated and transformed with minimal delay into valuable business information. The PDS involves several challenges:

Real-time data propagation. Delays in propagating a workflow audit trail from workflow management systems (WFMSs) to the PDS can significantly decrease the value of the audit trail information for the users. When workflow audit trail information is available in near real-time in the PDS, it gives managers and the operational staff accurate and detailed information about current business situations. In addition, it allows them to identify weaknesses and bottlenecks in the process handling earlier.

Adding business process context. The calculation of workflow metrics often requires additional information from other data sources. For instance, in the case of an order process, the workflow events do not include detailed information about the corresponding orders and customers. Nevertheless, order and customer information might be needed for the calculation of workflow metrics about order transactions. Therefore, during data integration the workflow audit trail has to be merged with additional information from other data sources.

Automated response mechanisms. The monitoring or analysis of workflow audit trails often entails a direct or indirect feedback into WFMSs or operational systems. This response can be done manually or automatically and enhances the operational system with business intelligence. This is usually referred to as closed loop analysis. In the case where an automatic response is required, the ETL process or an active PDS has to evaluate workflow metrics and trigger business operations.

Adaptive architecture. Business processes can change over time. Therefore, it is very crucial that the PDS also adapts to these changes. Traditional data warehouse systems are built with the assumption that business transactions change seldom. The PDS cannot make this assumption because it has to stay in sync with the currently executed business processes. Therefore, the schema and data propagation components of the PDS must be very configurable and extendible.

The remainder of this paper is organized as follows. In section 2, we discuss the contribution of this paper and related work. In section 3, we present an architecture for integrating workflow audit trail into a data warehouse environment. In section 4, we discuss the real-time integrating of workflow audit trail information into the PDS. Section 5 discusses in detail the ETL container which is the core component for the integration of workflow events. In section 6, we discuss data management issues of the PDS. Section 7 presents a case study and examples for the PDS. Finally, in section 8 we present our conclusion.

2. Related Work and Contribution

Although monitoring and analysis are considered as important tasks of the workflow management system (e.g. [4]), and the Workflow Management Coalition has already published a standard for workflow logs [9], little work has been done in integrating and analyzing the workflow audit trail information. Some approaches emphasize the need for integrating audit trail into data warehouse systems (e.g. the process data

warehouse in [7]), others are limited to a smaller set of workflow history that is managed within a workflow management system. To our knowledge there has been no work that thoroughly discusses an end-to-end solution for propagating, transforming and analyzing large amounts of workflow events in near real-time.

Sayal et al. present in [7] a set of integrated tools that support business and IT users in managing process execution quality. These tools are able to understand and process the workflow audit trail from HP Process Manager (HPPM), and can load them into the process data warehouse using a particular loader component. Sayal et al. provide a high-level architecture and a data model for the process data warehouse, but they do not discuss the problem of integrating and analyzing the workflow audit trail in near real-time. An approach for history management of audit trail data from a distributed workflow system is also discussed in [5]5. The paper describes the structure of the history objects determined according to the nature of the data and the processing needs, and the possible query processing strategies on these objects. These strategies show how to write queries for retrieving audit trail information. Unlike our approach, neither the transformation and aggregation of audit trail data, nor the analytical processing of this data is considered.

Geppert and Tombros [33.] introduce an approach for the logging and post-mortem analysis of workflow executions that uses active database technology. The post-mortem analysis is accomplished through querying the event history, which is stored in an active database system supporting Event-Condition-Action (ECA) rules. Various types of events (e.g., database transitions, time events, and external signals) in the event history can trigger the evaluation of a condition. If the condition evaluates to true, the action is executed.

Most of the existing WFMSs offer only basic monitoring and analytical capabilities, such as the retrieval of status information about process instances or summary information about cycle times. While this supports basic reporting, it requires considerable configuration effort and assumes the existence of comprehensive knowledge of the users to write reasonable queries. Furthermore, it does not provide active mechanisms for automated intelligent responses of currently running workflows. Our contribution in this paper is the characterization of the PDS, the identification of technologies that can support it, and the composition of these technologies in an overall architecture. We also show examples how the PDS can be applied to a real world scenario.

3. Architectural Overview

The main rationale of a PDS is to provide real-time information about business processes that can be used for monitoring purposes. It is conceptually equivalent to traditional operational data stores (ODSs) with the only difference being that it is used to store process and workflow data. The PDS can be combined with a process warehouse (PWH) [6] which is part of the enterprise data warehouse system and which is used for storing a rich set of historical process control data for the strategic decision support.7 The combination of the PDS and the PWH forms a process information factory, which is a data foundation for a process-driven decision support system to monitor and improve business processes continuously. It is a global process information repository, which enables process analysts and software agents to conveniently access comprehensive information about business processes very

quickly, at different aggregation levels, from different and multidimensional points of view, over a long period of time, using a huge historical data basis prepared for analyzing purposes to effectively support the management of business processes.

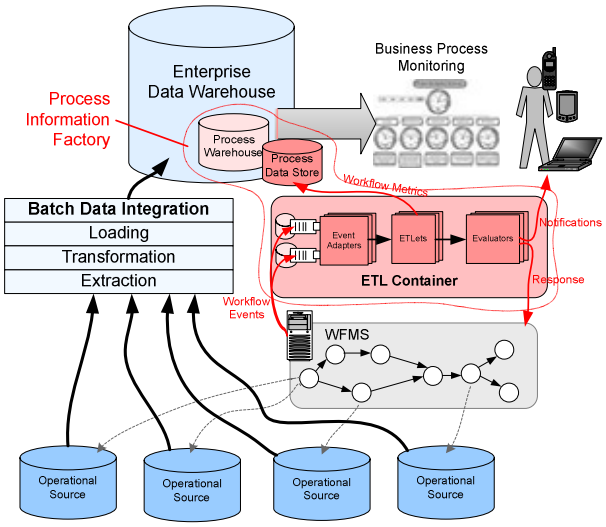

Figure 1: Process Information Factory with Process Data Store

The PDS is the component which includes very detailed up-to-date process data of currently running processes and also allows real-time access for the tactical and operational decision support. Figure 1 shows the PDS as part of the process information factory. In our architecture, we are using an ETL container for integrating event data continuously from a workflow management system. Arrows indicate a flow of data or control among these components. The components on the left highlight the extensions to a conventional (passive) data warehouse environment. The ETL container ultimately transforms on-the-fly workflow events into metrics that are stored in the PDS. Since the workflow events are continuously integrated, the data of the PDS can be used to create *automated intelligent responses* (e.g. sending out notifications to business people or triggering business operations) in near real-time. Therefore, the PDS addresses the need for collective, integrated operational, DSS/informational processing. A PDS is a hybrid structure that has equally strong elements of operational processing and DSS processing capabilities. This dual nature of the PDS easily makes it one of the most complex architectural structures in the process information factory.

A PDS is a collection of detailed data that satisfies the collective, integrated, and operational needs of the organization. Generally, these needs arise in the following situations: 1) as integrated operational monitoring and tactical decision support is needed for workflow management systems and also across multiple but related operational systems, 2) as strategic decisions are made using the PWH and action is required. A PDS is 1) subject-oriented, 2) integrated, 3) volatile, 4) current-valued, 5) detailed, and 6) adaptive to workflow changes. The PDS and the PWH are identical

when it comes to being *subject oriented* and *integrated*. There are no discernible differences between the two constructs with regard to those characteristics. However, when it comes to transaction support, level of integration with source systems, volatility, currency of information, history and detail, the PDS and the PWH differ significantly.

History of Data. In contrast to a PWH that is rich in history, a PDS generally does not maintain a rich history of process data, because it can be used within WFMSs and operational environments, and often has tight requirements for query response times. Consequently, a PDS is highly volatile in order to reflect the current status and information of business processes.

Data Propagation. For a PDS it is common that data records are updated. A PDS must stay in sync with the WFMSs to be able to respond to active workflows and to consistently operate within its environment. A PDS has predictable arrival rates for workflow event data and includes sync points or checkpoints for the ETL process.

Transaction Support. Data in a PDS is subject to change every time one of its underlying details changes. An advantage of the PDS is that it is integrated and that is can support both decision support and operational transaction processing. A PDS often requires a physical organization which is optimal for updates and flexible processing of data (e.g. "close" to a WFMS) while the PWH is not interwoven with the WFMS or other operational systems and requires a physical organization which is optimal for strategic analyses of processes.

Detailed Data. The PDS contains data that is very detailed, while a PWH contains a lot of summary data. The PDS also stores all workflow events with their attributes. The persisted event data is used to extract valuable business information which is stored in separated tables. Usually, the detailed event data is removed after a workflow instance has completed and all workflow metrics for this workflow instance have been calculated. Data in the PDS can also be summarized, but, because the summarized values are subject to immediate change (PDSs are highly volatile), it has a short effective lifetime.

Adaptive to Workflow Changes. Changes of the workflow or settings in the workflow management settings must not disrupt the interoperability with the PDS. The PDS must be able to adapt to such changes. Therefore, the schema and data propagation components for the PDS must stay in sync with the operational environment. This requires the PDS to be very adaptive to changes of the workflow models and settings in the workflow management system.

4. Real-Time Data Integration

Traditional data warehouse systems are refreshed, at best, on a daily basis and assume that data in the data warehouse can lag at least a day if not a week or a month behind the actual operational data. That was based on the assumption that business decisions do not require up-to-date information but very rich historical data. Existing ETL (Extraction, Transformation, Loading) tools often rely on this assumption and achieve high efficiency in loading large amounts of data periodically into the data warehouse system. While this approach works well for a wide range of data warehouse applications, the new desire for monitoring information about business processes in

real-time requires new mechanisms for propagating and processing data in near real-time. Workflow audit trail is a good example for data that provides the most value to the users and process analysts if it is available with minimal delays.

The processing steps for continuously integrating workflow data are not equivalent to the traditional ETL systems because the underlying assumptions for the data processing and the latency requirements are different. Traditional ETL tools often take for granted that they are operating during a batch window and that they do not affect or disrupt active user sessions. If data is integrated continuously, a permanent stream of data must be integrated into the data warehouse environment while users are using it. The data needs to be trickle-fed continuously from the source systems and therefore differs from traditional batch load data integration. The data integration process has to use regular database transactions (i.e. generating *inserts* and *updates* on-the-fly), because in general database systems do not support block operations on tables while user queries simultaneously access these tables. In addition, the data rate of continuously integrated workflow data is usually low with an emphasis on minimized latency for the integration. In contrast, bulk-loading processes buffer datasets in order to generate larger blocks, and therefore, the average latency for data integration with bulk loading increases. Data integration based on frequent bulk loads can exceed the data latency requirements.

Since data in the PDS changes in near real-time, the data integration process and the PDS themselves can monitor the data and the corresponding metrics, automatically detect conditions that are likely to interest various users, and proactively communicate these conditions to the business users. This is usually referred to as *active* decision-making [8]. The detection of situations and exceptions in the PDS often indicates a condition that requires action. With this fact in mind, users should not only receive alerts, but should also have the ability to respond to an existing business situation. The purpose of alerting could also be merely to ensure that appropriate personnel are notified about an important situation or exception. In this case, the required action in the workflow could be the simple acknowledgement of the alert. Other actions could include forwarding the information to someone else, flagging as unimportant, or increasing the threshold before future alerts are triggered. In choosing these follow-up options, users can help the system learn over time to improve the accuracy of detecting future situations and exceptions.

Workflow rules can also be enabled by the PDS. For example, if an alert is not acted upon within a certain timeframe, it may automatically escalate to another event or change the priority of a workflow instance. Or, if recipients reply that they will take action on an alert, they may need to follow up with more details on the status of the resolution within a certain period of time.

5. Real-Time Data Integration with an ETL Container

We built an ETL container for the integration of workflow events which provides a robust, scalable, and high-performance data staging environment, and which is able to handle a large number of workflow events in near real-time. In [1] we proposed the architecture for an ETL container, which provides services for the execution and monitoring of event processing tasks. The core services are responsible for creating, initializing, executing and destroying the managed components that are used for the event data extraction, data processing and evaluation. In this paper, we want to focus our discussion on integrating data into the PDS and the advantages of using an ETL

container for this task. Therefore, we only briefly discuss the components and services provided by the ETL container. A detailed discussion of the ETL container can be found in [1].

An ETL container is part of a Java application server and manages the lifecycle of ETL components. There are three component types that are managed by the container: 1) event adapters, 2) ETLets, and 3) evaluators. Event adapters are used to extract or receive data from a WFMS (e.g. utilizing a JMS-based interface), and they unify the extracted event data in a standard XML format. ETLets use the extracted XML event data as input and perform the data processing tasks. ETLets also publish workflow metrics that can be evaluated by evaluator components. Figure 2 shows how these ETL components work together.

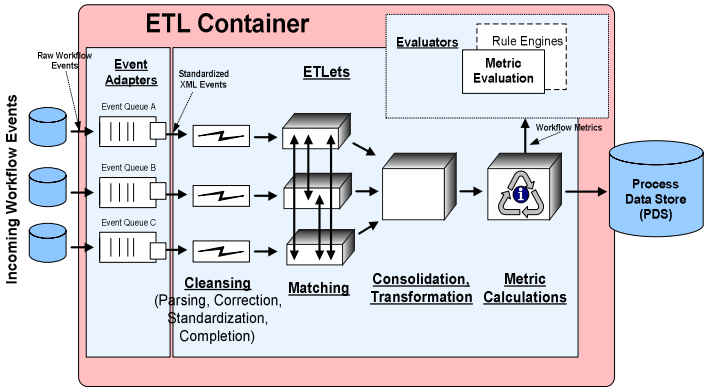

Figure 2: ETL Container – Architecture [1]

The ETL container handles incoming events with a lightweight Java thread, rather than a heavyweight operating system process. The event processing can be performed without using intermediate storage. When new events arrive, the ETL container decides which ETL components are called, in what sequence these components are executed, and which business metrics are evaluated. The ETL container is a crucial component for feeding data into the PDS and providing active mechanisms for automated responses based on the integrated event data. The main advantages for using the ETL container are summarized as follows:

- The separation of the extraction logic (event adapters), transformation logic (ETLets) and evaluation logic (evaluators) is very critical for the extendibility and configurability of the ETL components. It facilitates the maintenance and synchronization of the PDS with the workflow management system.
- ETLet components are used for the event data processing that can include complex calculations and aggregations of workflow metrics. During the data processing, ETLets can collect supplementary data from various data sources (e.g. ERP or legacy systems) that can be used for the calculation of workflow metrics.
- The scalability of the ETL container allows the processing of a large number of workflow events concurrently in near real-time.
- Mechanisms for evaluating fresh calculated and changed workflow metrics can be used for sending notifications or triggering business operations with minimal latency. The ability of directly responding to incoming workflow events can be

utilized for modifying currently running workflows (e.g. trigger an activity, increase the priority). The PDS contains all details about currently running workflow instances in order to enable automated actions and responses.

6. Data Management

The management of data that is integrated in near real-time raises some challenging data modeling issues. For instance, if the PDS includes aggregated workflow metrics at various levels based on a time dimension, the aggregates may be out of sync with the real-time data. Also, some metrics such as month-to-date and week-to-date metrics are permanently updated during the data integration.

The process data store includes two types of data: 1) very detailed event data, which is stored as workflow events streams into the PDS via the ETL container, and 2) detailed up-to-date process data at various granularity levels. For storing the workflow events in the PDS, the ETL container uses ETLet components to map all attributes of the workflow events in one or more database tables. Various XML technologies are available to perform this mapping (e.g. DB2 XML Extender, Oracle XML SQL Utility for Java). The detailed event data normally only resides in the PDS as long as it is needed for future metrics calculations. The time to live for event data is usually quite short since it has only limited use as direct input for the data analysis.

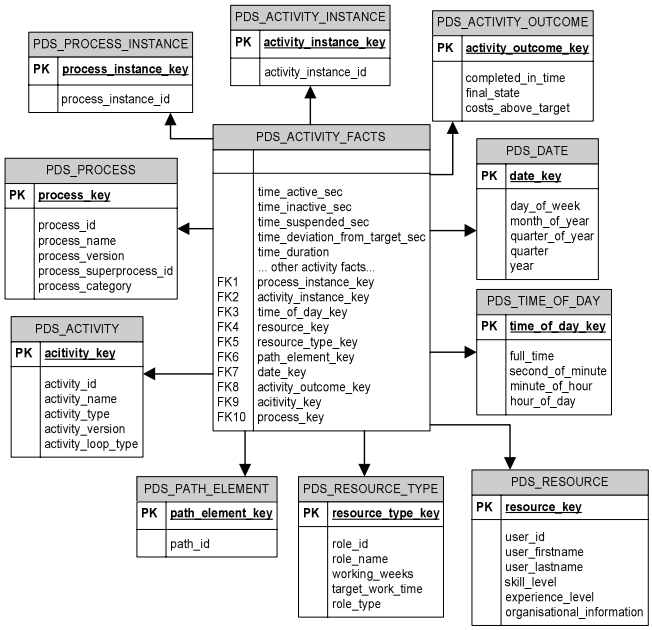

Figure 3: Activity-Level Facts of the PDS

A process is a set of one or more linked process activities, which may result in several work items. Processes and activities have hierarchical dependencies and therefore a lot of attributes in common such as workflow participants, but differ in

other aspects like instance states. Therefore, the PDS has to maintain separate process, and activity fact tables. Those dimensions, which are needed by all fact tables, enable a drill-across between the fact tables. The fact tables of the PDS look similar to the fact tables of the PWH with the only difference that they capture values of current running process instances. The PDS also uses accumulated fact tables for process instances, whose records are updated if there is a status change in the business process. Furthermore, the PDS fact tables often include a status dimension which allows to easy filtering or sorting of the process or activity instances by their current state.

Figure 3 shows a detailed logical data model for a star schema that supports the analysis of activity durations of currently running workflows. The model includes various dimensions that form the context of activity instances. The dimensions are used to show the activity facts from various perspectives. The schema illustrated in Figure 3 can be used to find out the status of a workflow instance by querying the last performed activity of a workflow instance. Furthermore, analysts can ask questions like what are the workflow instances that completed certain steps in the business process (e.g. an approval step).

7. Case Study and Examples

In this section, an example business process for a mobile phone company is presented. The business process demonstrates how the PDS supports the competitiveness of the entire organization. The business processes to be analyzed is the mobile phone activation process of a telecommunication company. Organizations or private individuals can be customers of the mobile phone activation process. The deliverable of the process is an activated mobile phone connection for a new customer. The mobile phone activation process includes the following three activities: (1) registration, (2) debt and liability examination, and (3) clearing and connecting or refusal. The activity 'registration' includes the registration of the customer's personal data and terms and conditions. The data is submitted to the organization via fax or e-mail. Within the activity 'debt and liability examination', the customer is examined in terms of his/her credit-worthiness. Therefore, the Debt Collection Order System, a system for credit assessment is used. The activity 'clearing and connecting or refusal' includes the clearing of the request and the connection to the mobile phone system or the refusal of the phone service.

The mobile phone activation process is carried out about 1000 times a day and the number of personnel engaged in this process is about 30. The business process is a so-called core business process and generates value to the company. The overall goal of the mobile phone activation process is *low process duration*. The customer is assured that thirty minutes after registration at the latest, the mobile phone connection is established. This is an objective with highest priority. The entire business process is carried out by a WFMS, and workflow events are captured by the PDS. The PDS is used for monitoring the current status of the activation process. If a mobile phone activation process instance is running out of time, an event is sent to the workflow engine and the priority of the process instance is increased. As a result, the process instance has a higher priority and it is therefore moved to the top of the process performer's worklist, ready for processing in time. Further, the process owner of the business process is also notified. Due to an unexpected peak, additional process performers might be required.

As demonstrated, the PDSs main area of application is a highly competitive and fast-paced market with low margins like the Telecom or the Grocery sector. The PDS approach is especially suited for production workflows, because they generate a high business value, have a high degree of repetition, are customer focused and often time critical. Very tight task supervision that can be observed, reminds on the era of scientific management methods. The number of measures in the PDS is very low compared with the PWH. Measures are either time or frequency based and have a very low level of granularity as well as a short-term focus. Basically, measures can be characterized as quantitative and unbalanced. However, process cycle times, working times and waiting times are not sufficient key figures to assure long-term process quality. But, nevertheless they can be described as the most important performance drivers, which support long-term performance results. The analysis of these short-term performance drivers are the main objectives of the PDS.

8. Conclusion

A traditional data warehouse focuses on strategic decision support. A well-developed strategy is vital, but its ultimate value to an organization is only as good as its execution. As a result, deployment of data warehouse solutions for *operational* and *tactical* decision-making is becoming an increasingly important use of information assets. In this paper we have discussed the process data store, which is a part of the solution for the challenge of continuously integrating workflow event data and enables access to workflow metrics with minimal latency. A process data store is well integrated with the operational environment for business processes and is a data foundation that supports these processes with near real-time business intelligence.

References

1. Bruckner, R. M., Jeng, J. J., Schiefer, J.: *Real-time Workflow Audit Data Integration into Data Warehouse System.* ECIS, Naples, 2003.
2. Dresner, H.: *Business Activity Monitoring: New Age BI?*, Gartner Research LE-15-8377, April 2002.
3. Geppert, A., Tombros, D.: *Logging and Post-Mortem Analysis of Workflow Executions based on Event Histories.* Proc. 3rd Intl. Conf. on Rules in Database Systems (RIDS), Springer LNCS 1312, pp. 67-82, Skövde, Sweden, June 1997.
4. Jablonski, S., Bussler, C.: *Workflow Management. Modeling Concepts, Architecture, and Implementation.* Thomson Computer Press, London, 1996.
5. Koksal, P., Alpinar, S. N., Dogac, A.: *Workflow History Management*. ACM SIGMOD Record Vol. 27(1), pp. 67-75, 1998.
6. List, B., et al., The Process Warehouse – A Data Warehouse Approach for Business Process Management, MiCT 1999, Copenhagen, Denmark, 1999.
7. Sayal, M., Casati, F., Dayal, U., Shan M.: *Business Process Cockpit*. Proc. of VLDB 2002, Peking, 2002.
8. Thalhammer, T., Schrefl. M.: *Realizing Active Data Warehouses with Off-the-shelf Database Technology*. In Software - Practice and Experience, Vol. 32(12), pp. 1193-1222, 2002.
9. Workflow Management Coalition Audit Data Specification, Document Number WFMC-TC-1015, 1998.