Business Intelligence Revisited

Jianwen Su and Yan Tang
Department of Computer Science
UC Santa Barbara, USA
{su, yantang}@cs.ucsb.edu

***Abstract*—Combining big data with machine learning is a powerful tool for business intelligence (BI). Developed in the database community, The traditional ETL-data warehouse-OLAP approach to BI is effective to deal with multi-dimensional data (i.e. data cubes) but not suitable for flexible analytics such as exploration with ad hoc queries and process/data changes. Process mining techniques developed in the BPM community focus on activities and control flow but ignore data. In this paper, we propose a new framework for business analytics based on *workflow logs*. We introduce the key notions, illustrate querying logs as one useful aspect, and then discuss a range of interesting technical problems to be studied further.**

I. Introduction

According to a recent study [1], in almost every world's top economy, services (as opposed to products) contribute 60% to 80% GNP to the economy, and a majority of services are based on information (as opposed to material). Typically, services are provided through performing business operations, which in turn are accomplished by executing *business processes*. Business processes are also used in many aspects of enterprise operations such as management of products, inventory, and human resources. It becomes apparent that business process (or workflow) management (BPM) lies at the heart of all organizations including government agencies, healthcare institutions, and business enterprises.

Gartner places business process improvement as a top business strategy of CIOs in enterprises [12]. A precursor to business process improvement and workflow change is the analysis of past business process/workflow executions to learn about the characteristics of these executions. *Business intelligence* (or *BI*) is a collection of techniques for gathering, storing, accessing, and analyzing data to help business managers and stakeholders make better decisions for improving business operations and processes.

As a simple example, consider a permit approval process in a real estate management agency of some government. During the process execution, staff in the agency might want to know: the number of applications that have been lodged since the beginning of the year, the peak time of the application lodging, the applications that have passed "Preliminary Decision", and the applications that did not follow the defined process. Such information are often used in key performance indicators (KPIs) for business process improvement. Under the current design practice for business process/workflow management systems, the data required to answer the above mentioned questions are typically scattered across various places with different formats and sometimes conflicting semantics: process logs, activity logs, event logs, data stores, process models, execution engines, just to name a few.

Two relevant developments in the past two decades should be mentioned here. The database community invested a significant effort into data/process warehouse techniques to help enterprises for their analysis of business process/workflow executions. Data/process warehouse techniques first *Extract* data from various databases containing business process/workflow execution data, *Transform* them into suitable forms, e.g., relations, and *Load* up the data into data warehouses (databases). Once the 3-step ETL is completed, online analytic processing (OLAP) tools are then used for process analysis [9][32].

Arising from the BPM research community, process mining [29] is a research topic with growing interest that employs machine learning and other techniques to discover, monitor, and improve business processes through extracting knowledge from event logs. While some systems provide event logs directly, in other systems event data needs to be re-constructed from many tables or needs to be tapped off from subsystems exchanging messages. In such cases, event data exist but some efforts are needed to extract them. Based on the event logs, three types of process mining tasks can be performed: *Discovery* of process models with possibly related models for e.g., resources; *Conformance* of an existing process model with an event log of the same process; *Enhancement* of an existing process model, i.e., an extension or improvement using information about the actual process executions recorded in the log. For examples, process *repair* to modify the model to better reflect reality has been studied in recent years [11][19], *extension* is another type of enhancement to add a new perspective to the process model by cross-correlating it with the log ([5]).

In spite of the solid progress made in both ETL-OLAP and process mining fronts, there are still significant obstacles for these techniques to address the needs from the

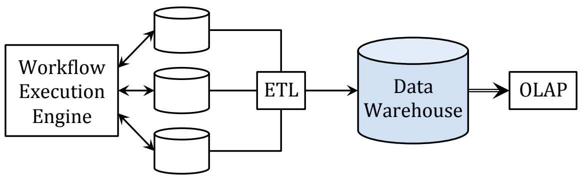

Fig. 1. Traditional Framework Using ETL/OLAP

present applications. Most of process mining techniques focus on activities (tasks, services) while ignoring data. Incorporating the data into the techniques is extremely hard, and will require new techniques. On the other hand, ETL-OLAP is only effective when the relevant data are extracted, and even though the information of activity sequencing is not readily available even if the data is extracted. For example, OLAP typically relies on SQL that is strong in summaries and aggregation but has no basic support for temporal reasoning (e.g., LTL style). Also, when processes evolve (new/modified data, new/modified activities), ETL needs to be manually changed/adjusted.

In addition, in most enterprises, a great amount of information concerning their business processes does *not* exist in digital form. For example, the common term of "institutional memory" often refers to senior managers in the organization who are extremely knowledgeable about the history, rationals, and other possibly complicated details concerning the business operations. Quality of business analytics can be significantly enhanced by incorporating human knowledge into big data and machine learning. An initial idea is to support *explorations* of business data through, e.g., ad hoc queries. Thus, the ability to query about execution status, gather all traces of tasks, and find correlations of instances is a key element for business process analysis and improvement.

In this paper we formulate a new framework for business analytics based on workflow logs. A (logical) workflow log is a faithful recording of executions of process/workflow instances in a workflow management system. Specifically, the data accessed and modified by the instances are mostly reflected in the log records. Although there are no standards of logging proces/workflow executions, logging mechanisms in most workflow systems records the detailed effects of execution in several logs: event log, activity log, instance log, etc. More importantly, workflow logs cleanly separate analytics problems from irrelevant system details and allow BI techniques more generally applicable. We illustrate ad hoc log queries as an example use.

This paper is organized as follows. Section 2 briefly reviews ETL-OLAP techniques and their weaknesses. Section 3 introduces the concept of workflow log. Section 4 discusses a range of research problems and challenges with the new log-based framework.

II. ETL AND OLAP

In this section we provide a very brief overview the ETL-OLAP approach to BI and discuss its deficiencies. The discussions provides a basis for the new BI framework based on workflow logs. Relevance to process mining will be mentioned at the end of the section.

A workflow management system is a software system for managing executions of tasks (activities) in business processes. Fig. 1 illustrates the traditional framework [8]. The workflow engine schedules and manages executions of process/workflow instances (enactments). The execution of instances may result in the effects or data changes stored in multiple databases.

Traditionally, a main research focus of BI is on data warehousing and OLAP [8][7][2]. Since 1990s, this field has been focused on integration of data from multiple sources using Extract-Transform-Load (ETL) tools, effective/efficient data store, and support of multidimensional data analysis using OLAP.

As the first step in a typical BI architecture (Fig. 1), ETL tools are used to extract data from multiple sources (databases), then transform the source data schemas into target data schema, and finally integrate the data in a data warehouse. For *extraction* phase, minimum overheads should be incurred to the source system when extraction happens and the extraction tool should be installed on source side with minimum effect to source's configuration [17][16]. For *transformation* phase, different applications should have different ETL processes designed to convert the source data representation into target data representation. This phase is always done in an ad-hoc manner in data warehouse and optimization of transform process is of interest [21]. The third phase *loading* takes place in a periodic fashion to refresh the data warehouse. The data that arrives at this phase should be loaded to appropriate destination tables. Then there would be issues about updating dimension tables and fact tables [31]. At the same time, indexes and materialized views must also be maintained [20][22][13]. To support front-end applications, OLAP [28] provides operations such as filtering, aggregation, pivoting, rollup and drill-down on the multidimensional view of relational data. Other analytic techniques studied include streaming event processing [33], data mining, text analytic processing, etc. Traditional data warehousing has been mainly focused on structured data management and analysis. However, it barely considers business processes during the design and execution of analytic process.

These technical advancements have helped enterprises tremendously in business process analytics. However, the current process data warehousing techniques present some interesting challenges: (1) Since the data is tightly coupled with the process model and its execution engine, it is hard to provide *generic* solutions for warehousing process data for different processes. (2) Data warehousing approach is not efficient for runtime execution monitoring and analysis (e.g, [6]). (3) Process warehousing gets the data but misses the process information. As a result, when process/workflow models change, ETL mechanisms for the warehouse often need be changed.

As a specific aspect, while effective ETL is seriously challenged as enterprises demand more flexible process analytics. The ETL approach typically focuses on/supports specific types of analysis queries centered around summaries over data cubes [8]. It is effective only when the relevant data are extracted. For examples, if the timestamps are not extracted, analysis on activity durations is not possible, or if drug prescriptions are not available in the data warehouse, it is impossible to discover medical fraud that patients sell over prescribed drugs in the underground market, or if procedure codes "left-chest-tube" and "right-chest-tube" are merged into "chest-tube" during ETL, it is impossible to discover the correlation between left chest tube and patients on beds with windows on the left. These are samples of analytics that enterprises nowadays hope to perform, which we call *flexible* (or *ad hoc*) *process analytics*.

Process enactments (executions) are temporal in nature. ETL turns enactments into essentially relational databases, often resulting in loss of temporal relationships. For example, in detecting medicare fraud, the amount of prescription medicine covered would depend on the diagnostics and treatment procedures. It may not be possible to identify fraudulent cases from incomplete temporal relationships.

OLAP does an effective job on dealing with multi-dimensional data using data cubes. However, there are many analysis tasks in BI that do not focus on multi-dimensional data and data cubes. SQL is often not an effective tool for exploring process enactments.

Different business processes need to engage with each other to achieve competitiveness. Enabling collaboration remains a fundamental challenge, Collaboration needs to address two fundamental issues: (1) different model transformation, and (2) runtime process execution status and behavior analysis. The former can smooth the communication, while the latter is critical for execution analysis, monitoring and management. Web service standards such as WSDL, BPEL, WSCDL have provided basic interoperation support specified in terms of flow of activities, messages to be exchanged, roles and relationships. But they do not provide a satisfactory support in runtime analysis, monitoring and process change. ETL-OLAP is also weak in dealing with enactments of collaborating processes.

Process mining is the task of re-constructing the workflow model from workflow logs [29]. We omit the details here since existing work mostly ignored data contents. Incorporating data requires new data modeling/abstraction/manipulation techniques that are not quite compatible with the current graph/network-oriented process mining techniques.

A key observation from the above discussions is that, in spite of significant recent progress in process modeling and enactment, there is a lack of integrated conceptual models and support tools that can capture a sufficient semantics of business processes for runtime execution queries, business analysis, and process improvement. There is a clear need of new techniques for data modeling and manipulation for business process/workflow enactments to help both flexible process analytics and process mining. This paper focuses on this important issue.

III. Workflow Logs

We argue that an effective framework to support general purpose, flexible business analytics can be built around the key notion of a workflow log. In this section we first briefly discuss the evolution of incorporating data into business process/workflow model and management, define the notion of a workflow log, and then illustrate as one concrete aspect how languages can be developed to support ad hoc queries on logs.

Earlier process modeling languages emphasize on activities and control flow, and pay almost no attention to modeling data that are used in processes. The case for incorporating data into business process/workflow modeling was convincingly made in the seminal paper by Nigam and Caswell in 2003 [18]. This work inspired development of concrete technical modeling languages, e.g., [3], that use data-centricity techniques for business process/workflow modeling, these technical models have been studied and extended [15].

The trend of combining data and process is now evident. In the past decade, the BPM research and development communities have made the transition from focussing primarily on activities/tasks and control flow in business process/workflow modeling to elevating data to the same importance as activities/tasks and control flow [3][15].

A recent article [25] classified data used in business processes/workflows into five classes: (1) *Specification of process/workflow models* that serve as road maps for execution, (2) *Business data* (e.g., about the applicant, total cost), (3) *Execution status* (e.g., the initial review is completed and an auxiliary background check process instance is initiated), (4) *Correlation* relations of the current instance and its collaborating instances, and (5) *Resouce* information *and status* (e.g., credit report request is made but the report has not arrived).

Based on the ability in modeling and managing data, Process modeling approaches are also divided into the following 5 groups [25]. *Data agnostic models* essentially ignore all data other than class 1 but focus on activities possibly with arrangement of activities using swimlanes and pools; execution status data is only partially available and implicit through the execution semantics. Petri nets, workflow nets, UML activity diagrams, and BPMN are typical examples. *Data aware models* use variables to hold a small subset of business data (class 2) and are capable of specifying detailed control flow logic for processes. This group includes BPEL and YAWL [30]. *Storage aware models* (e.g., jBPM, UML with both class and activity diagrams) include the concept of persistent data stores and modeling of enterprise data. *Artifact-centric models* (e.g., DEZ-Flow [34], GSM [14]) include conceptual data modeling for each process. However, in data/storage aware and artifact-centric models, mappings between persistent data in the enterprise database and process data are defined at the implementation level through SQL expressions (jBPM) or Java Hibernate. *Semantic process models* (e.g., SeGA [24][25]) can capture data of all 5 classes and have the ability to specify mappings between all data in business processes and the enterprise database [23].

It is interesting to note that the timing for the development of data warehouse and ETL-OLAP techniques in the database community coincided very well with the use of data agnostic models in business process/workflow research and development.

While the BPM communities are embracing data wholeheartedly in business process/workflow modeling, what advancements can be made in the data management communities on process analytics? Clearly, data warehouse and ETL technology needs a rethinking. There is an urgent need to fully embrace *process-centricity* in business process analytics. This concept means to keep *all* process execution information and make it directly available for analytics. The term "process-centric BI" was invented in a 2009 paper [4] by Bucher, Gericke, and Sigg who adopted the ETL approach but did not develop concrete techniques for keeping process information in the general manner.

While the trend of embracing data in business process modeling is going strong, we expect process-centricity to be a new breed of business process analytics. Recently, a new framework for BI was developed in [27] in the work on a query language that supports ad hoc queries. Fig. 2 illustrates their new approach for flexible process analytics that center around "workflow logs". While the workflow engine advances process/workflow instances, it also records the key actions into a workflow log in addition to the relevant databases. The log faithfully records the execution traces for all active instances in their execution order.

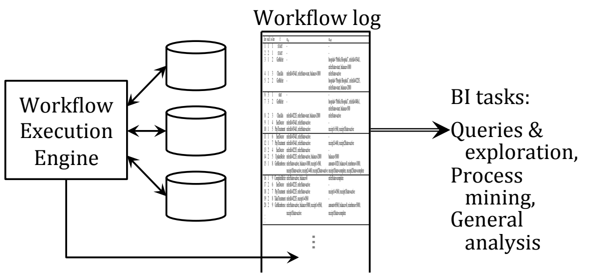

Fig. 2. A Log-Based Framework for Flexible Process Analytics

A *workflow (execution) log* is a chronicle sequence of log records generated by all active process/workflow instances. Specifically, when a new process instance starts a START log record is written on the log which contains a *log sequence number* (lsn), an *identifier* for the instance, an *instance-specific (is-)lsn* that should be 1 (first log record for the instance), the current *timestamp*, and other relevant data such as the event and the contents that causes the launch of the instance, data generated or modified, etc. When an instance ends, an END log record is written and similarly it contains lsn, instance identifier, is-lsn, timestamp, modified data, and event(s) generated by completing the instance. During the execution of an instance, for each activity execution, choice and other control flow decision, there should be at least one log record containing lsn, instance identifier, is-lsn, timestamp, the name of the activity or control flow decision, and possibly data read (input) and written (output).

While the structure of such a workflow log might be new, the contents of the log are available in most workflow systems. For example, Activiti spreads the contents of the log in several relations in a relational database, jBPM does this in a similar way. Collecting the data from difference sources and assembling them into a workflow log as defined in the above is straightforward.

Workflow log has several advantages. For example, classic algorithms for process mining can easily run on traces that are extracted from the log by simply ignoring all other data except for activity name and instance identifier. The work reported in [10] cleverly utilizes the "redo" log in Oracle DBMS to generate traces to feed into process mining algorithms. Workflow log easily removes the need for crafting ad hoc workflow/database systems to produce the trace data.

In the remainder of the section, we briefly illustrate how workflow logs can be queried through several examples. Since there is no prior data filtering, querying the log allows the user to formulate a much richer set of queries. Log queries were initially studied in [27] with an algebraic language. Our examples below will use a query language currently being developed [26].

Consider the Emergency Department (ED) of some hospital. During flu season, many patients who visited ED for flu-like symptoms later evolved to develop pneumonia. The staff at ED are especially interested in find cases of patients who did blood tests for pneumonia. This query can be expressed as the following.

```
FOR     INSTANCE L IN ED-Log
SELECT  X.wid
FROM    CheckIn@L X, BloodTest@L Y
WHERE   X << Y
```

Here the FOR clause specifies that the query intents to find (workflow) instances in the log named "ED-Log": the keyword "INSTANCE" indicates that the variable "L" represents a workflow instance. The SELECT clause specifies what the query answer should be. In the above query, variable X holds a log record and "X.wid" means the workflow instance id of log record held in X. The SELECT component plays a similar role as the Select clause in SQL. The FROM clause indicates which log records in the instances are used to answer the query (including both the query condition and answer). In this example, "CheckIn@L" and "BloodTest@L" specify that X and Y (resp.) should hold log records for activities *CheckIn* and *BloodTest* (resp.) in the instance L. Note that both records must belong to the same instance. Finally, the WHERE clause specifies the condition to be satisfied by the query answer. In this example, the WHERE clause contains a single *incident* expression "X << Y" to mean that log record X occurs before log record Y, i.e., the activity *CheckIn* occurs before *BloodTest*. Notation "<<" denotes a sequential operator, the language also includes consecutive, choice, and parallel operators.

The hospital is located in an area with many retirement communities. Among the symptoms of pneumonia is low body temperature for older adults. A resident doctor is conducting a research trying to find cases of young adults (ages 18 to 25) with low body temperature and diagnosed as having pneumonia. To express this query, data contents of log records have to be examined. The following is the complete query expression.

```
FOR     INSTANCE L IN ED-Log
SELECT  X.wid, X.out.systolic
FROM    CheckIn@L X, Diagnosis@L Y
WHERE   X[bodyTemp<36 AND 18⩽age⩽25]
        << Y[cause="J18.9"]
```

This query also prints a blood pressure reading (systolic) for the patient. The keyword "out" indicates the value is in the "out-map" of the log record, i.e., at the completion of activity X. The query condition checks if the patient's age is in the range between 18 and 25, patient's body temperature is low, and the diagnosis is pneumonia (coded as J18.9 in ICD-10). Note that these conditions are enclosed by square-brackets "[ ]" following X or Y, which indicates that the data values to be examined are the values when the activities CheckIn, Diagnosis complete (i.e., in their out-maps).

Details of the language(s) for workflow log query can be found in [27][26]. Clearly, directly querying workflow logs is new and interesting. The approach extends the traditional data warehouse and OLAP approach [9][32] by allowing richer set of ad hoc queries for business users.

IV. Problems and Challenges

As hinted in the previous section, workflow logs provide a logical structure that can hide the application dependent system details from business analytics tasks. This separation empowers BPM practitioners and researchers to focus on their respective tasks, a widely adopted divide-and-conquer strategy. For example, the use of workflow logs can avoid getting into the details of DBMS redo logs [10]. The concept of logical workflow logs also brings several research problems and challenges to the community. In the following we briefly discuss a few interesting ones.

Defining and (Re-)Construction of Workflow Logs

There are no standards for workflow logs. Unlike logging in DBMSs that is transparent to the user, workflow logs provide a powerful means to business analytics and stake-holders, and are clearly not transparent. An initial problem is to develop languages to define logical workflow logs to suit specific application contexts. Consequently, logs with the defined structure should be generated, either by the workflow management system directly or constructed from the various existing (event, activity, etc.) logs. Practical systems (e.g., jBPM, Activiti) all support logging; while these systems log process/workflow execution in multiple places with different formats, the information captured can essentially be used to construct workflow logs.

Log Query Languages

As demonstrated in [27][26], query languages for logs need to take into consideration of process model or at least control flow elements and they can remain easy to use. Optimization techniques are necessary since logs can be of large size. Database query optimization techniques may help, new techniques must also be developed. For example, what would be an effective index to find all pneumonia patients who initially had flu? The ability to query about multiple correlated instances, about instances from different logs is also very useful in practice. Last but now least, support for summary and aggregations remains very relevant. These features might be provided with an extension of the language in [26] borrowing features from XQuery.

Process Mining with Data

Workflow logs effectively define a clean input for the process mining problem. With the data naturally included in log records, it will be natural to augment the process mining problem with data. A seemingly fruitful hint is that expressions in query languages capture semantics of process model fragments with data. It appears possible to combine query expressions with traditional activity-focused process mining techniques. For example, if a query expression captures all executions that did not follow a process model, perhaps a suitable model repair could be to combine the expression into the original process model in a "succinct" manner. From the other side, one could also "remove" a query expression from an existing process model. Developments along this direction will be extremely useful for process evolution.

Tools & Techniques for Applications

Finally, many application problems can be easily formulated based on workflow logs. Although some of the problems are not new, adapting them to workflow logs elevates the algorithms and solutions to an abstract level that will be more generally useful. For example, for effective resource management one hope to predict the needs for the future. Fraud detection is useful in many applications (such as medicare, financial), the current techniques mostly depend on ad hoc contexts. In some application domains, knowlege workers are highly sought after. A business enterprise usually trains new employees (for 3-6 months) and often trained new employees switched jobs resulting a loss to the enterprise. It is possible but interesting to see how workflow logs can be used to shorten the training period, a practical help to such enterprises.

References

[1] U. Apte, U. Karmarkar, and H. Nath. The U.S. information economy: Value, employment, industry structure, and trade. *Found. and Trends in Tech., Info. & Operations Manag.*, 6(1), 2012.

[2] A. Berson and S. Smith. *Data warehousing, data mining, and OLAP*. McGraw-Hill, 1997.

[3] K. Bhattacharya, C. Gerede, R. Hull, R. Liu, and J. Su. Towards formal analysis of artifact-centric business process models. In *Proc. Int. Conf. on Business Process Management (BPM)*, 2007.

[4] T. Bucher, A. Gericke, and S. Sigg. Process-centric business intelligence. *Business Process Management Journal*, 15(3):408–429, 2009.

[5] J. Buijs, M. L. Rosa, H. Reijers, B. van Dongen, and W. van der Aalst. Improving business process models using observed behavior. In *Proc. Int. Symp. on Data-Driven Process Discovery and Analysis*, pages 44–59, 2012.

[6] M. Castellanos, U. Dayal, T. Pedersen, and N. Tatbul, editors. *Enabling Real-Time Business Intelligence – Int. Workshops (BIRTE 2013) and (BIRTE 2014), Revised Selected Papers*, volume 206 of *Lecture Notes in Business Info. Processing*. Springer, 2015.

[7] S. Chaudhuri and U. Dayal. An overview of data warehousing and OLAP technology. *SIGMOD Record*, 26(1):65–74, 1997.

[8] S. Chaudhuri, U. Dayal, and V. Narasayya. An overview of business intelligence technology. *Comm. of the ACM*, 54(8):88–98, 2011.

[9] U. Dayal, M. Castellanos, A. Simitsis, and K. Wilkinson. Data integration flows for business intelligence. In *Proc. Int. Conf. on Extending Database Technology (EDBT)*, 2009.

[10] E. de Murillas, W. van der Aalst, and H. Reijers. Process mining on databases: Unearthing historical data from redo logs. In *Proc. 13th Int. Conf. Business Process Management (BPM)*, 2015.

[11] D. Fahland and W. van der Aalst. Model repair – aligning process models to reality. *Information Systems*, 47:220–243, 2015.

[12] G. Group. Gartner Newsroom. http://www.gartner.com/it/page.jsp?id=1740414, 2011.

[13] A. Gupta and I. Mumick. Incremental maintenance of aggregate and outerjoin expressions. *Info. Systems*, 31(6):435–464, 2006.

[14] R. Hull, E. Damaggio, R. D. Masellis, F. Fournier, et al. Business artifacts with guard-stage-milestone lifecycles: Managing artifact interactions with conditions and events. In *Proc. ACM Int. Conf. on Distributed Event-Based Systems (DEBS)*, 2011.

[15] R. Hull, J. Su, and R. Vaculin. Data management perspectives on business process management: tutorial overview. In *Proc. ACM Int. Conf. on Management of Data (SIGMOD)*, 2013.

[16] W. Labio and H. Garcia-Molina. Efficient snapshot differential algorithms for data warehousing. In *Proc. Int. Conf. on Very Large Data Bases (VLDB)*, pages 63–74, 1996.

[17] B. Lindsay, L. Haas, C. Mohan, H. Pirahesh, and P. Wilms. A snapshot differential refresh algorithm. In *Proc. ACM Int. Conf. on Management of Data (SIGMOD)*, pages 53–60, 1986.

[18] A. Nigam and N. S. Caswell. Business artifacts: An approach to operational specification. *IBM Systems Journal*, 42(3):428–445, 2003.

[19] A. Polyvyanyy, W. van der Aalst, A. ter Hofstede, and M. Wynn. Impact-driven process model repair. *ACM Trans. on Software Eng. and Methodology*, 25(4), 2016.

[20] N. Roussopoulos, Y. Kotidis, and M. Roussopoulos. Cubetree: Organization of and bulk updates on the data cube. In *Proc. ACM Int. Conf. on Management of Data (SIGMOD)*, 1997.

[21] A. Simitsis, P. Vassiliadis, and T. Sellis. Optimizing ETL processes in data warehouses. In *Proc. Int. Conf. on Data Engineering (ICDE)*, pages 564–575, 2005.

[22] Y. Sismanis, A. Deligiannakis, N. Roussopoulos, and Y. Kotidis. Dwarf: Shrinking the petacube. In *Proc. ACM Int. Conf. on Management of Data (SIGMOD)*, pages 464–475, 2002.

[23] Y. Sun, J. Su, B. Wu, and J. Yang. Modeling data for business processes. In *Proc. Int. Conf. on Data Engineering (ICDE)*, 2014.

[24] Y. Sun, J. Su, and J. Yang. Separating execution and data management: A key to business-process-as-a-service (BPaaS). In *Proc. Int. Conf. on Business Process Management (BPM)*, 2014.

[25] Y. Sun, J. Su, and J. Yang. Universal artifacts: A new approach to business process management (BPM) systems. *ACM Trans. Management Information Systems*, 7(1), 2016.

[26] Y. Tang and J. Su. Querying business process enactments, 2017. (In preparation).

[27] Y. Tang and J. Su. Querying workflow logs. In *Proc. 1st Int. Workshop on Integrating Process-oriented, Event-based, and Data-driven Systems (ICDCS-PED)*, 2017.

[28] E. Thomsen. *OLAP solutions: building multidimensional information systems*. John Wiley & Sons, 2002.

[29] W. van der Aalst. *Process Mining: Data Science in Action*. Springer, 2016.

[30] W. van der Aalst and A. ter Hofstede. YAWL: yet another workflow language. *Information Systems*, 30(4):245–275, 2005.

[31] P. Vassiliadis and A. Simitsis. Extraction, transformation, and loading. In *Encyclopedia of Database Systems*, pages 1095–1101, 2009.

[32] R. Wrembel and C. Koncilia. *Data Warehouse and OLAP: Concepts, Architectures and Solutions*. IRM Press, 2007.

[33] E. Wu, Y. Diao, and S. Rizvi. High-performance complex event processing over streams. In *Proc. ACM Int. Conf. on Management of Data (SIGMOD)*, pages 407–418, 2006.

[34] W. Xu, J. Su, Z. Yan, J. Yang, and L. Zhang. An artifact-centric approach to dynamic modification of workflow execution. In *Proc. Int. Conf. on Cooperative Information Systems (CoopIS)*, pages 256–273. Springer, 2011.