Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 64 (2015) 1026 – 1034

Procedia Computer Science

Conference on ENTERprise Information Systems / International Conference on Project MANagement / Conference on Health and Social Care Information Systems and Technologies, CENTERIS / ProjMAN / HCist 2015 October 7-9, 2015

Measuring and querying process performance in supply chains: An approach for mining big-data cloud storages

Alejandro Vera-Baquero[a], Ricardo Colomo-Palacios[b,*], Owen Molloy [c]

[a]*Universidad Carlos III de Madrid, Av. Universidad 30, Leganes (Madrid) 28911, Spain*
[b]*Østfold University College, B R A Veien 4, Halden 1783, Norway*
[b]*National University of Ireland, Galway, Ireland*

Abstract

Survival in today's global environment means continuously improving processes, identifying and eliminating inefficiencies wherever they occur. With so many companies operating as part or all of complex distributed supply chain, gathering, collating and analyzing the necessary data to identify such improvement opportunities is extremely complex and costly. Although few solutions exist to correlate the data, it continues to be generated in vast quantities, rendering the use of highly scalable, cloud-based solutions for process analysis a necessity. In this paper we present an overview of an analytical framework for business activity monitoring and analysis, which has been realized using extremely scalable, cloud-based technologies. It provides a low-latency solution for entire supply chains or individual nodes in such chains to query process data stores in order to deliver business insight. A custom query language has been implemented which allows business analysts to design custom queries on processes and activities based on a standard set of process metrics. Ongoing developments are focused on testing and improving the scalability and latency of the system, as well as extending the query engine to increase its flexibility and performance.

Peer-review under responsibility of SciKA - Association for Promotion and Dissemination of Scientific Knowledge

Keywords: Business Process Analytics; Business Performance Management; Big Data.

* Corresponding author. Tel.: + 47 6921 5000; fax: + 47 6921 5002.
E-mail address: ricardo.colomo-palacios@hiof.no

Peer-review under responsibility of SciKA - Association for Promotion and Dissemination of Scientific Knowledge
doi:10.1016/j.procs.2015.08.623

1. Introduction

For companies to remain viable in the face of intense global competition, they must be able to continuously improve their processes in order to respond rapidly to changing business environments. Through the analysis of historical, and increasingly, real-time data, Decision Support Systems (DSS) and Business Intelligence (BI) tools can assist end-users in gaining an insight into business processes, key intelligence in initiatives aimed at measuring and improving overall business performance [1,2]. However, traditional BI platforms are normally business domain specific and are not sufficiently process-aware to support the needs of process improvement type activities [3], which may entail integrating, monitoring and analysing a vast number of distributed, unstructured event logs, produced on a variety of heterogeneous environments, in a timely manner.

The latest advances in technology have made it possible for organizations to co-operate with each other, necessitating the integration of diverse business information systems across large and complex supply chains in several domains [4,5]. In these scenarios, isolated optimisation within individual organizations is insufficient to optimize and improve end-to-end processes. Therefore "events need to be correlated across organizational boundaries"[6]. This leads to the management of non-trivial operational processes, where web services technology and cloud computing have become widely used, producing cross-functional event logs that are beyond company (and increasingly software) boundaries. This has promoted an incredible growth in event data in corporations that needs to be merged for analysis [6]. In addition, enterprises' business data are usually handled by heterogeneous systems which run on different technological platforms, and even use incompatible standards. Those systems are usually critical for corporations' efficiency, making them difficult and costly to replace them or re-engineer. Furthermore, the continuous execution of distributed business processes (BP) generates vast amounts of event data that cannot be efficiently managed by the use of traditional storage systems which are not designed to manage event data of the order of hundreds of millions of linked records [7].

Big-data technology can be used productively in managing supply chains [8]. Previous authors' work in the area of business process analytics (BPA) introduced in [7], [9], [10] overcome this problem by offering both an analytical framework for monitoring highly distributed business processes, and a methodology to guide BPI activities using the proposed IT system. This analytical framework provides a fully distributed solution that leverages emerging technologies, such as big-data and cloud-computing to support collaborative business process analytics in (or near) real-time. Big-data analytics require high-performance processors to achieve timely results, whereby cloud-computing infrastructures are ideal for meeting the computational and storage needs of BPA applications over big-data [11]. This is especially interesting in business scenarios that involve very complex and highly distributed processes like supply chain management. For instance, manufacturing processes produce massive amounts of data due to the continuous execution of process operation and control [12], thus, big-data technology has tremendous significance in this field. Likewise, the framework can support not only inter-departmental but also cross-organizational performance analysis in a distributed and collaborative fashion rather than at individual supply chain nodes, thus leveraging big-data technology for performing end-to-end analysis. In this paper, authors explore the real-time measurement capabilities of the system and we propose a built-in specific-purpose SQL-like language for mining large event data cloud repositories, implemented using big-data technology.

2. Analytical Framework

Real-time measurement and big-data analysis of the performance of managerial activities can provide tremendous competitive advantage to corporations [13] by allowing them to react quickly to competitors in an increasingly competitive environment. The monitoring of business process executions allows business users to detect defects and non-compliant business situations. This action must be performed promptly in order to respond to those inconsistencies quickly. In a well running process, it is desirable that work arrival (demand) and throughput rates are in balance. Processes or activities which do not have the capacity to work at the rate of demand will cause delays and bottlenecks, thereby starving proceeding activities of input. This may result in a loss of profit due to a waste of valuable resources that are underutilized, and in consequence, a quality degradation and loss of customer satisfaction and loyalty.

Vera-Baquero et al.[7] introduced a cloud-based platform with big-data support for providing timely business performance analysis on highly distributed environments. The analytical framework comprises a set of cloud-computing nodes that provides analytical services to third party applications, namely BASU (Business Analytics Service Unit) and GBAS (Global Business Analytical Service) modules. These service components have the capability for collecting data originating from distributed heterogeneous systems, unifying and correlating process instances, storing the enterprise data leveraging big-data underlying technology and inferring knowledge from the gathered information. These modules are designed monitor and analyse operational activities within both local and global contexts. In a local context, the processes reside within an organization where they can be analysed and optimized by the means of local BASU units. These units are used by individual organizations to manage their internal processes. In contrast, the GBAS component supports the monitoring, analysis and optimization of large and complex supply chain processes like manufacturing and retail distribution. The overall architecture (Fig. 1) has the ability to provide cloud-based services at very low latency response rates. These services can contribute to the continuous improvement of business processes through the provision of a rich informative environment that supports BPA and offers clear insights into the efficiency and effectiveness of organizational processes. The architectural concerns of the overall system in general, and the internals of the involved components in particular, are out of scope at the present paper. Further details can be found at [7].

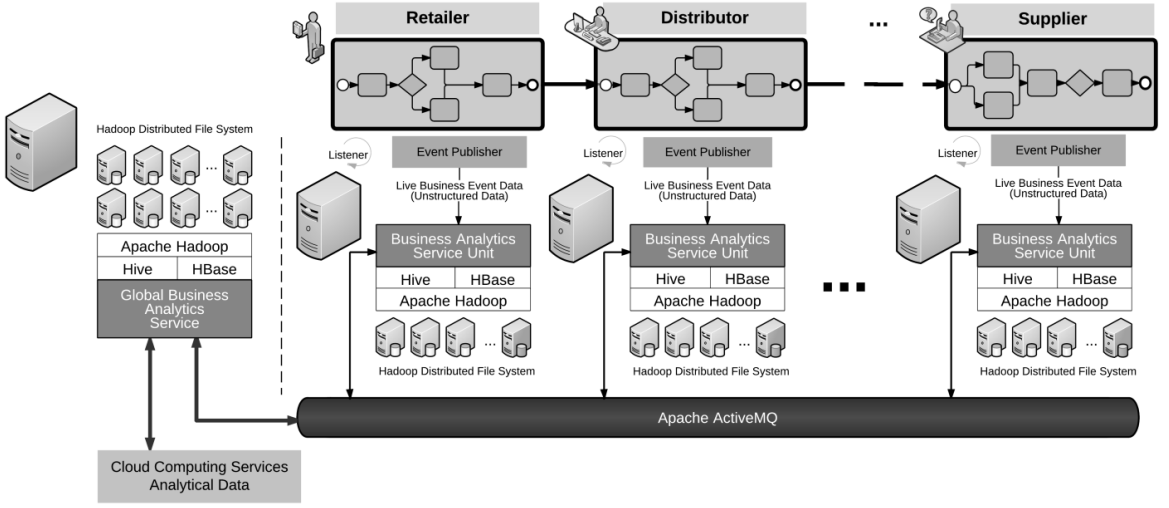

Fig. 1 - Overall Architecture

The following sections explore the big-data capabilities of the system to produce metrics and perform BAM activities in real-time. Likewise, it presents the latest advances of this research work by introducing a specific-purpose SQL-like language that leverages the BPAF model, and applies its fundamentals to the context of big-data with the aim at mining cloud storages.

2.1. Event-Based model

In order to enable analysts to infer knowledge about business performance we need to define an event model to provide the framework of a concrete understanding and representation of what needs to be monitored, measured and analysed [14]. The proposed event model is based on BPAF standard, and it provides the information required to enable the global system to perform analytical processes over them, as well as representing any derived measurement produced during the execution of any business process flow. BPAF supports the analysis of audit data across heterogeneous BPM systems [15], and it enables the delivery of basic frequency and timing information to decision makers, such as the cycle times of processes, wait time, etc. This permits host systems to determine what has occurred in the business operations by enabling the collection of audit data to be utilized in both analysis and the

derivation of status information [16]. This format has been extended to support the event correlation algorithm of distributed processes. The proper instance identification is essential for measuring process performance as these must be correlated before any analysis is to be attempted. Otherwise, it would not be possible to generate metrics per process instance or activity [14], thus preventing analysts from identifying exceptional situations or discovering business opportunities. The event correlation process must be performed at very low latency rates in order to enable the system to generate metrics in (or near) real-time. The details of the correlation algorithm are out of scope at the present paper, but it basically relies on fnding the associated process instance or activity in the entire event cloud repository. This is achieved by using the triplet - source, model and event correlation, where event correlation is given in the form of key-value pairs. This triplet must guarantee that one, and only one, instance can be associated to a determined process model, executed on a determined supply chain node, and run with a specific correlation data.

Whereas the event repository storage may raise to very high volumes of event information, the system leverages big-data technology to identify consecutive instances along the mass set of data. The event repository is a cloud data store that supports the event-based model described above and it is implemented using HBase and Apache Hadoop as big-data storage. This big-data repository is designed as a cloud-based solution in order to allow the system to scale up easily on readily available hardware, which is essential for dealing with the data-intensive processing demands of the correlation process. Consequently, it allows us to determine the trade-off between volume of data, KPI latency and hardware investments, and thereby we can easily adapt the analytical framework to multiple business domains with very specific business demands.

The very low latency of the correlation algorithm is achieved by using secondary indexes in HBase over the event correlation table. This enables the event correlation mechanism to have immediate access to events that meet the set of key-value pair conditions for a specific source and model. Since it is not possible to know or estimate beforehand the execution time of BPs, the event correlation process must have available all historical data in order to identify correlated instances. Depending on the business case, the volume of the event storage can rise to the order of TBs, PBs or even EBs of information. According to [7], the read operations over the row key in the event table are performed in the order of milliseconds on very small clusters (approx. 4 nodes) that handle hundreds of millions of event records. Since read operations present very-low latency based on the row keys, we can leverage this feature to enable real-time analysis. The idea behind this approach is to create an alternate HBase table that will be used as a secondary index for the event correlation table by using the triplet filter as a row key with the aim at speeding up the HBase scans. The row key is established as a byte stream sequence of strictly ordered key-value pairs for every event such as follows:

RowKey: Key1Value1Key2Value2KeyN…ValueNSourceModel cf: "event_correlation" {eventId}

The use of secondary indices requires data duplication but it provides an extraordinary response time on read operations, albeit to the detriment of writes. Achieving low rates on reads is essential for the correlation algorithm in order to rapidly identify process or activity instances, and is key for having metrics available on time.

Once the instances are correlated on a timely fashion, we already have the event streams ready for analysis. Both event-data and process models together are essential to infer knowledge about process improvement. Process models by themselves represent the structural aspects of process instances, but a purely structural representation is not enough to construct a solid understanding of what needs to be improved. It is also required to capture and represent the behavioral state of these processes. Therefore the measurement of process performance is equally a critical factor, and thus the construction of metrics and KPIs must be accomplished at minimum latency.

2.2. Metrics and KPIs

The process metrics are generated by analysing the timestamps of a set of correlated events (event stream) which are associated with a specific process instance or activity. The analytical framework leverages the BPAF capabilities and adopts the zur Muehlen & Shapiro's work [17] for constructing performance metrics per process or activity. The system captures and records the timestamp of the events containing the time at which they occurred on the source system, not when they are packaged or delivered. This is essential for ensuring that the generation of metrics produces correct and precise output. Particularly for supply chains where multiple sources set timing information on the events, the timestamps must be perfectly synchronized across nodes. Otherwise the analytical data might become

untrustworthy. Based on the event timing information and the BPAF state model, it is possible to measure different behavioural aspects of business processes with the aim of analysing the behaviour of completed process instances, evaluating currently running process instances, or even predicting the behaviour of process instances in the future [16]. One key challenge in decision making is having access to all relevant information in order to undertake a performance and compliance assessment. As already stated, in order to provide Business Activity Monitoring functionality in (or near) real time, the construction of metrics and KPIs must be performed at minimum latency. To assist in this regard, we have incorporated an intermediate in-memory cache solution with event data eviction. This cache has been implemented using Infinispan and configured as a distributed cache with replication (owner nodes). As the event correlation identifies sequence of events in time as they arrive, those events are temporarily stored in a distributed cache, so that a large number of events co-exist during a variable period of time, in both permanent storage and cache. Once an event is correlated in the stream chain, this event is cached and associated with the event stream of its process instance. Additionally, there is an observer object which is continuously listening to events that transition towards a COMPLETED state. At that precise moment, the entire event stream is read from cache for that particular instance in place and forwarded to the data warehouse module for processing. Once the metrics processor is notified of a new incoming instance that has finished its execution, the event stream sequence is analysed and the metrics are produced according to the state transition changes based on zur Muehlen & Shapiro's model [17] (see Fig. 2). We leverage the state change records in the life cycle of the business process instance to determine the following metrics:

- **Turnaround**: Measures the gross execution time of a process instance or activity.
- **Wait Time**: Measures the elapsed time between the entrance of a process or activity in the system and the assignment of the process or activity to a user prior to start its execution.
- **Change-over Time**: Measures the elapsed time between the assignment of the process or activity to a user and the execution start of the process or activity.
- **Processing Time**: Measures the net execution time of a process instance or activity.
- **Suspend Time**: Measures the suspended time of a process or activity.

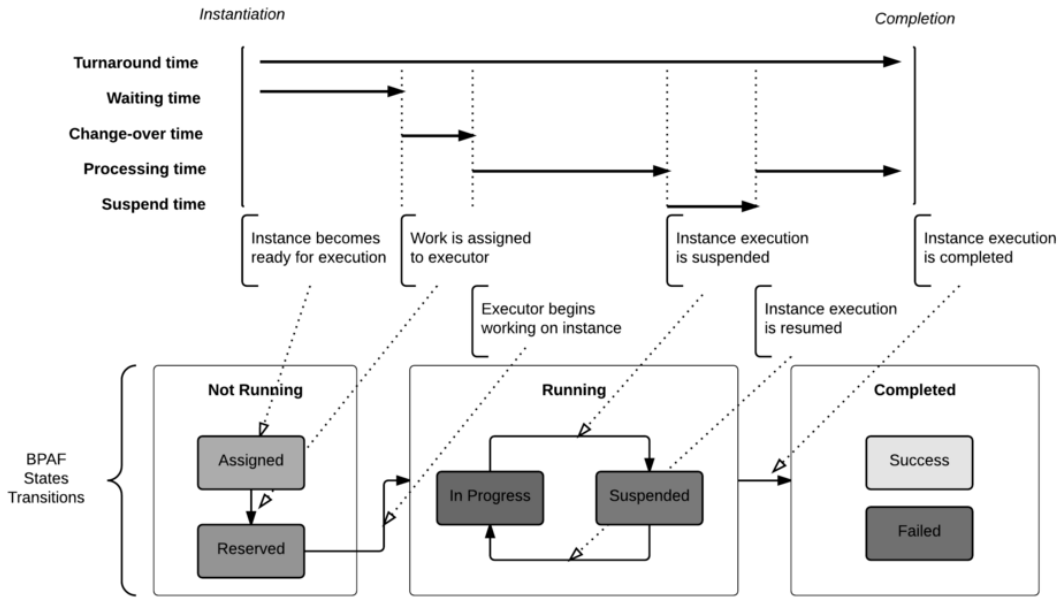

Fig. 2 - Instance metrics (adapted from [17]).

Once the metrics are generated, these are stored back in cache and persisted in HBase. The data warehouse solution represents a star schema but with the difference that the fact table has been denormalized to improve the

speed of BPEQL queries (see next section) over HBase. The reasons why metrics are stored in cache for a fixed period of time with data eviction backed by HBase, is twofold: 1) enabling low-cost time access to metrics of the latest instances with the aim at supporting real-time BAM capabilities, and 2) giving the framework real-time support for retrieving metrics of newly processed instances through API calls.

3. Business Process Execution Query Language

So far we have shown how the framework gives analysts an insight into what is happening on the business operations on real-time, but not how to access historical data of old instances in order to know what happened in the past. Commonly, analysts need to know how the business behaved during a certain period of time, learn from the errors experienced in the past or see the evolution of business operations over time. One key challenge in decision making is having access to all relevant information in order to undertake a performance and compliance assessment. Such information is normally distributed on diverse heterogeneous systems belonging to different organisational units. In such cases, not only the gathering, unification and correlation of event data are required, but also the ability to query the event repository and display the data thereof.

There exist some approaches for performing ah-doc queries on event logs which have been proposed in [18] for retrieving BP execution data (behavioural aspects of business processes) and [19] for BP model data (structural aspects). To the best of the authors' knowledge, as of yet there does not exist a standard for querying business processes over big-data deployments, and thus we propose a BPEQL (Business Process Execution Query Language) to enable systems to access structural and behavioural properties of business processes over big-data cloud storages. This is achieved by implementing a BPEQL engine in different scopes, one in every BASU component for retrieving intra-organisational performance data, and the other in the GBAS component for analysing the performance of distributed processes across supply chain nodes. The internals of the BPEQL module are similar on both components. It provides a query engine that processes query statements formulated in the BPEQL language and works with a big-data background for processing the translated queries. The query engine works as a translator by parsing and converting BPEQL query statements into HiveQL. Once the queries are translated, these are forwarded to the data warehouse component which performs the query over the big–data tables and returns the result back to the query engine. The HiveQL serves as a suitable intermediary layer for accessing the metrics stored in the data warehouse. The resulting data is then transformed into entity beans which are bound to a specific relational domain model. The transformation between the big-data storage and the relational world is performed by the DataNucleus framework, which enables binding big-data tables in HBase with a relational view of the data via a JPA implementation. This analytical information is exposed throughout cloud computing services and is available to third-party applications.

The query engine uses the ANTLR runtime for parsing and translating the queries into HiveQL. ANTLR (ANother Tool for Language Recognition) is an open source product that “provides a framework for constructing recognizers, interpreters, compilers and translators from grammatical descriptors” [20]. The BPEQL grammar is an SQL-like language based upon of a subset of the ANSI-SQL standard. The advantage of the languages based on an SQL-like syntax is that SQL is an industry standard that is widely used in business environments, and many non-technical people are familiar with it [21]. An SQL-based language is intuitive, and easier to learn for business users who are familiar with the notion of entities and queries for reporting purposes. The proposed grammar has been drastically reduced with respect to ANSI-SQL and some other features have been incorporated to adapt the language to the business process domain. The specification of the BPEQL grammar is as follows:

```
SELECT [AGGREGATE] ((*) | (id | name | source | start_time
                | end_time | successful | failed
                | aborted | turn_around | wait
                | change_over | processing | suspend))
  FROM (ACTIVITY | PROCESS | MODEL)
[WHERE condition]
```

The *select* clause specifies the attributes that will be presented in the output. Such attributes correspond to the metrics described above and they are self-explanatory. The optional *AGGREGATE* clause groups the result in just

one row and applies the average function to the metric attributes. This clause should be used with caution, as this operation may take some time depending on the volume of data stored in the cloud storage. The *FROM* clause specifies the context domain of data where the information will be retrieved from. It can specify indistinctly an ACTIVITY, PROCESS or MODEL. The optional *WHERE* clause may specify alternatively an ID or a NAME or both. The next figure illustrates how the BPEQL translator works internally by generating the parse-tree accordingly to the code snippet outlined above. Furthermore, it evaluates the semantic rules at the parse tree nodes against the input statement. The translation is carried out on the following input query:

BPEQL	*HiveQL*
SELECT id, start_time, end_time *FROM PROCESS* *WHERE name = 'ProcessTripOrder'*	*SELECT i.id, start_time, end_time* *FROM event_fact f* *JOIN process_instance i on (f.process = i.id)* *JOIN process_model m on (m.id = i.id)* *WHERE f.activity is null* *AND name = 'ProcessTripOrder'*

Figure 3 shows how the parse-tree generates the HiveQL statement in pieces while processing their nodes.

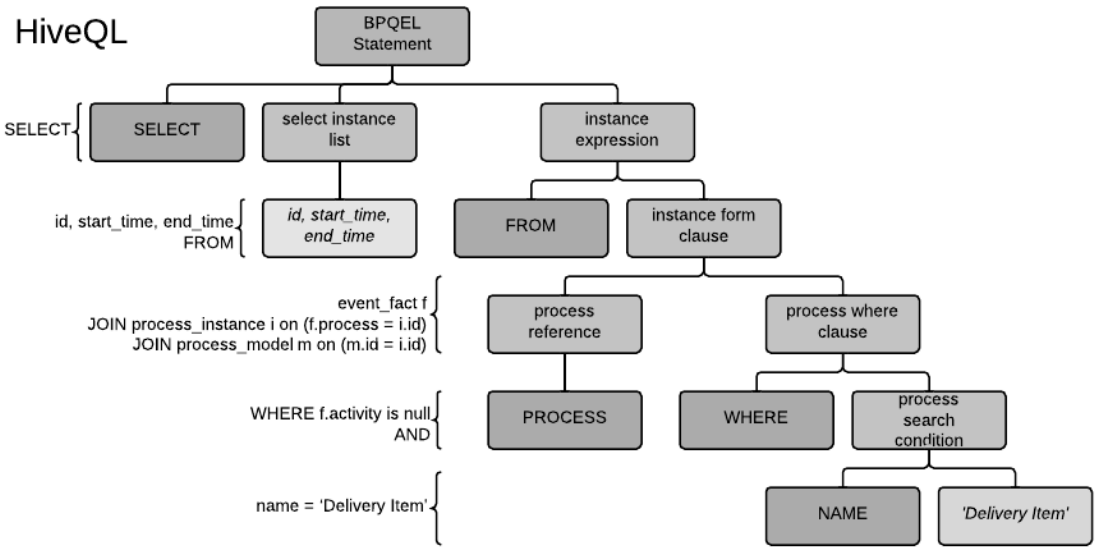

Fig. 3 - BPEQL syntactic tree

4. Evaluation

The evaluation has been carried out on a test environment using a 4-nodes cluster for the big-data solution plus a number of nodes for the functional deployment. The infrastructure of the analytical framework (see Fig. 1) was deployed on a variety of servers using the Google cloud platform, and we generated a vast amount of data that has risen to 200 hundred millions of records to date, and it continues to grow. The experiment is still ongoing, and so far we have achieved outstanding results on the correlation algorithm, which manages to link consecutive instances in between 0 and 3 milliseconds for such volume of data. Whereas the cluster size was very small, the correlation process performed at very low latency rates, thus exceeding the authors' expectations. As expected, the performance of the metrics generation process was equally excellent as the entire operation is done in-memory thanks to the built-in cache system. Future endeavours will be devoted to gradually increasing the rate of the incoming events per second, and analysing the impact of data overflow on cache. This could easily be solved extending the cache size by adding new nodes to the distributed cache, but this might affect the performance of our solution. It is worthwhile for the authors to analyse the frameworks behaviour in these edge-cases. ....It is worthwhile for the authors to analyse the frameworks behaviour in these edge-cases. This evaluation has been purely focused on a performance

perspective, but a more considered analysis of the functional framework and its applicability has been extensively covered in different case studies published in [7] and [8].

In relation to the BPEQL language, it is worth pointing out that the big-data infrastructure used was built upon a Cloudera (CDH 4.7.1) configuration using a cluster of 4 nodes. We tested BPEQL queries against both Apache Hive and Cloudera Impala. In general, neither of them was suitable for performing real-time queries using the given configuration, but they were very good for batch processing. Therefore the BPEQL solution is still far from getting real-time outcomes, however this is greatly dependent on the cluster configuration and the star schema modelled. Future work will be focused on study the inclusion of secondary indexes in the data warehouse solution as well as increasing the cluster size with the aim at finding the balance between data volumes and response time for historical data retrieval.

5. Conclusions and Future Work

A cloud-computing solution to support distributed business process analysis based on big-data technology has been reviewed in this paper. The proposed solution aims to provide integrated cloud services for monitoring and analysing business process performance over highly distributed environments. We have focused on the internals of the systems for supporting real-time BAM functionality by addressing the challenges involved in the generation of metrics with ultra-low latencies. This feature has enabled the system to access structural and behavioural properties of business processes in real-time, thus enabling third party applications to access online performance information. In order to provide support for performing ad-hoc queries over historical data, a BPEQL query language has been presented with the aim of mining big-data cloud repositories. The query engine provides HiveQL support within the context of BASU and GBAS modules. Queries can be either executed in isolation within each local node, or globally via the GBAS node in order to get instant access to inter-organisational performance data. Even though inter-organizational analytical data is available on GBAS, it may not have access to certain data located within each BASU node due to security reasons or due to a high nested level for a specific task which is unreachable from the GBAS node itself. Those cases require drilling down into a greater level of detail for a specific task within a particular BASU node. Hence, the extension of the query engine is also part of future work for fulfilling the objectives presented in this paper in terms of distributed processing and collaborative environments.

In an ideal scenario, business process analytical techniques are intended to be performed over very large amounts of data, whereby the scalability of the system, in terms of volume of data and workload on business queries, gains an enormous significance in the system evaluation. This is an important area to explore in further research work as part of that already mentioned in the evaluation section.

References

[1] García-Moreno C, Hernández-González Y, Rodríguez-García MÁ, Miñarro-Giménez JA, Valencia-García R, Almela A. A Semantic based Platform for Research and Development Projects Management in the ICT Domain. J Univers Comput Sci 2013;19:1914–39.
[2] García-Sánchez F, Valencia-García R, Martínez-Béjar R. An integrated approach for developing e-commerce applications. Expert Syst Appl 2005;28:223–35. doi:10.1016/j.eswa.2004.10.004.
[3] Van Der Aalst W. Using Process Mining to Bridge the Gap between BI and BPM. Computer 2011;44:77–80. doi:10.1109/MC.2011.384.
[4] Colomo-Palacios R, Stantchev V, Rodríguez-González A. Special issue on exploiting semantic technologies with particularization on linked data over grid and cloud architectures. Future Gener Comput Syst 2014;32:260–2. doi:10.1016/j.future.2013.10.021.
[5] Stantchev V, Colomo-Palacios R, Niedermayer M. Cloud Computing Based Systems for Healthcare. Sci World J 2014;2014:e692619. doi:10.1155/2014/692619.
[6] Van der Aalst W. Process Mining: Making Knowledge Discovery Process Centric. SIGKDD Explor Newsl 2012;13:45–9. doi:10.1145/2207243.2207251.
[7] Vera-Baquero A, Colomo-Palacios R, Molloy O. Business Process Analytics Using a Big Data Approach. IT Prof 2013;15:29–35. doi:10.1109/MITP.2013.60.
[8] Milliken AL. Transforming Big Data into Supply Chain Analytics. J Bus Forecast 2014;33:23–7.
[9] Vera-Baquero A, Colomo-Palacios R, Molloy O. Towards a Process to Guide Big Data Based Decision Support Systems for Business Processes. Procedia Technol 2014;16:11–21. doi:10.1016/j.protcy.2014.10.063.
[10] Vera-Baquero A, Colomo-Palacios R, Molloy O, Elbattah M. Business process improvement by means of Big Data based Decision Support Systems: a case study on Call Centers. Int J Inf Syst Proj Manag 2015;3:5–26.
[11] Talia D. Clouds for Scalable Big Data Analytics. Computer 2013;46:98–101. doi:10.1109/MC.2013.162.
[12] Qin SJ. Process data analytics in the era of big data. AIChE J 2014;60:3092–100. doi:10.1002/aic.14523.

[13] Chan JO. An Architecture for Big Data Analytics. Commun IIMA 2013;13:1–13.
[14] Molloy O, Sheridan C. A Framework for the use of Business Activity Monitoring in Process Improvement. E-Strateg. Resour. Manag. Syst. Plan. Implement. Alkhalifa, IGI Global; 2010.
[15] WfMC. Workflow Management Coalition - Business Process Analytics Format Specification 2012. http://www.wfmc.org/Download-document/Business-Process-Analytics-Format-R1.html.
[16] Muehlen M zur, Shapiro R. Business Process Analytics. In: Brocke J vom, Rosemann M, editors. Handb. Bus. Process Manag. 2, Springer Berlin Heidelberg; 2015, p. 243–63.
[17] Zur Muehlen M, Shapiro R. Business Process Analytics. In: vom Brocke J, Rosemann M, editors. Handb. Bus. Process Manag. 2, Berlin, Heidelberg: Springer Berlin Heidelberg; 2015, p. 243–63.
[18] Beheshti S-M-R, Benatallah B, Motahari-Nezhad HR, Sakr S. A Query Language for Analyzing Business Processes Execution. In: Rinderle-Ma S, Toumani F, Wolf K, editors. Bus. Process Manag., Springer Berlin Heidelberg; 2011, p. 281–97.
[19] Deutch D, Milo T. A structural/temporal query language for Business Processes. J Comput Syst Sci 2012;78:583–609. doi:10.1016/j.jcss.2011.09.004.
[20] Parr T. ANTLR - ANother Tool for Language Recognition 2014. http://www.antlr3.org.
[21] Rozsnyai S, Schiefer J, Roth H. SARI-SQL: Event Query Language for Event Analysis. IEEE Conf. Commer. Enterp. Comput. 2009 CEC 09, 2009, p. 24–32. doi:10.1109/CEC.2009.14.