IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 12, 2018, accepted November 6, 2018, date of publication November 16, 2018,
date of current version December 31, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2881759

Business Process Analytics and Big Data Systems: A Roadmap to Bridge the Gap

SHERIF SAKR[1], (Senior Member, IEEE), ZAKARIA MAAMAR[2], AHMED AWAD[1,3], BOUALEM BENATALLAH[4], AND WIL M. P. VAN DER AALST[5]

[1]Institute of Computer Science, University of Tartu, 50409 Tartu, Estonia
[2]College of Technological Innovation, Zayed University, Dubai, UAE
[3]Cairo University, Giza, Egypt
[4]School of Computer Science and Engineering, University of New South Wales, Sydney, NSW 2015, Australia
[5]Department of Computer Science, RWTH Aachen University, D-52074 Aachen, Germany

Corresponding author: Sherif Sakr (sherif.sakr@ut.ee)

The work of S. Sakr and A. Awad was supported by the European Regional Development Funds via the Mobilitas Plus Programme under Grant MOBTT75.

ABSTRACT Business processes represent a cornerstone to the operation of any enterprise. They are the operational means for such organizations to fulfill their goals. Nowadays, enterprises are able to gather massive amounts of event data. These are generated as business processes are executed and stored in transaction logs, databases, e-mail correspondences, free form text on (enterprise) social media, and so on. Taping into these data, enterprises would like to weave data analytic techniques into their decision making capabilities. In recent years, the IT industry has witnessed significant advancements in the domain of Big Data analytics. Unfortunately, the business process management (BPM) community has not kept up to speed with such developments and often rely merely on traditional modeling-based approaches. New ways of effectively exploiting such data are not sufficiently used. In this paper, we advocate that a good understanding of the business process and Big Data worlds can play an effective role in improving the efficiency and the quality of various data-intensive business operations using a wide spectrum of emerging Big Data systems. Moreover, we coin the term *process footprint* as a wider notion of process data than that is currently perceived in the BPM community. A roadmap towards taking business process data intensive operations to the next level is shaped in this paper.

INDEX TERMS Business process analytics, Big Data systems, process data-intensive operations

I. INTRODUCTION

It is largely accepted that a Business Process (BP) is a set of coordinated activities that are executed so, that, specific business objectives are achieved. Examples of BPs are expense claim processing, customer order handling, and inpatient treatment billing. A Business Process Management System (BPMS) provides the necessary support for the design, enactment, and monitoring of BPs [1]. Over the years, BPMS had to cope with multiple technological advances like ubiquitous computing, cloud/edge computing, and Internet-of-Things (IoT). A recent advance that is putting pressure on any BPMS is the large volume of unstructured data (e.g., text, image, and audio) that originate from different (sometimes unknown) sources (e.g., sensors, smart phones, and social media) that BPs might have to be aware of so, that, better decisions can be made. Known as Big Data, this advance is changing the way enterprises are operating by being able to handle such large volumes of data and extract insights in a timely manner [2].

Usually, Big Data is understood as massive data stored in a distributed file system and Big Data processing hints to the MapReduce computational model [3]. Big Data systems can roughly be classified into: offline (batch) processing and online (stream) processing. For the former, based on the nature of data, other computational models than MapReduce can be applied. For instance, to process large graphs like social networks, the Bulk Synchronous Parallel (BSP) computational model is recommended in terms of speed and ease of producing results. For the latter, MapReduce cannot handle the limited time overhead to respond to data as they arrive. Thus, other computational models have to be used where data are processed on the spot and processing is split among different operators. We are witnessing the uptake of a new generation of Big Data systems like Spark, Flink, Storm,

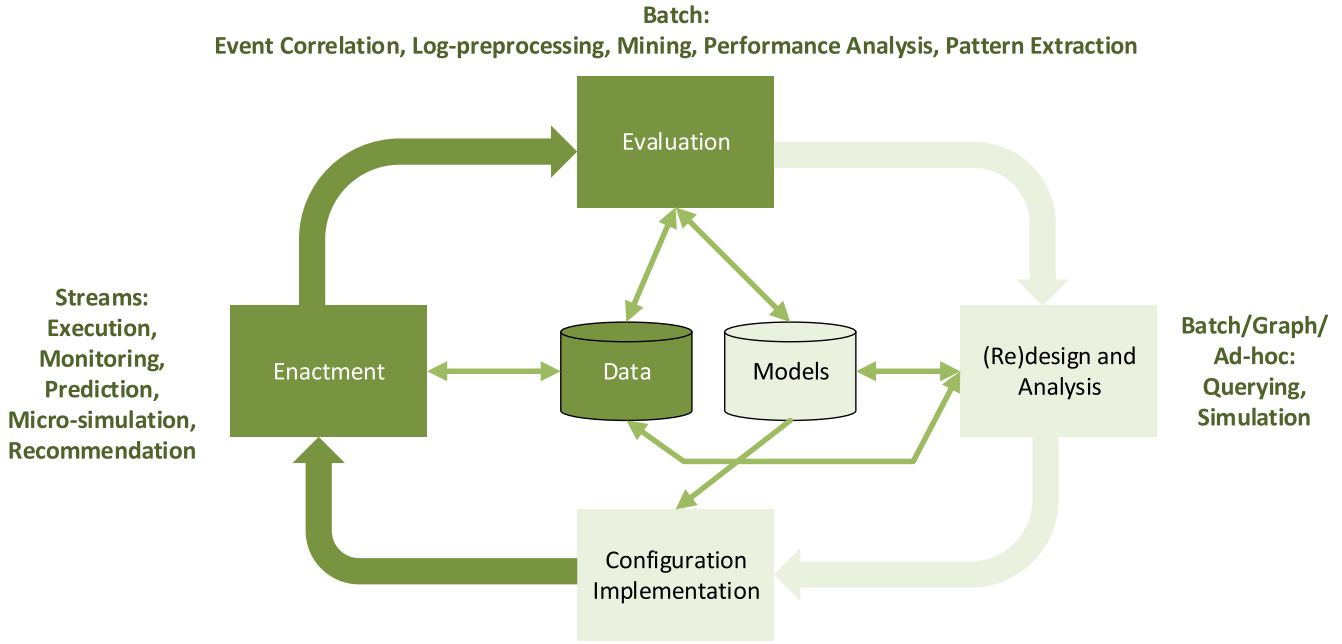

FIGURE 1. Business process life-cycle based on [1].

and Impala [4]. These systems provide technical solutions that can sustain the ability of drawing conclusions based on processing and analyzing *all data* [5].

BPM-related practice and research have not fully caught up, yet, with the benefits that Big Data processing can provide. The focus of this article is to highlight the existing gap between BPM and Big Data technologies and to present various insights towards filling this gap. To better articulate the potential benefits, we show how Big Data processing can be woven into the different phases of a BP's lifecycle.

The remainder of this article is organized as follows. Section II discusses process data generation. Section III discusses data intensive operations in BPM. Section IV provides an overview of Big Data processing systems. Section V discusses opportunities to connect the BP and big data worlds together. Section VI concludes the paper.

II. PROCESS (BIG) DATA GENERATION

To appreciate how BPM can benefit from Big Data processing, we start by describing the BP's lifecycle in terms of its main phases and the activities taking place in these phases (Figure 1). For each phase, we discuss what type of data is generated and how big it can be. Furthermore, we briefly describe Big Data processing techniques that can fit into that phase which we elaborate more on in Section V. We argue that the degree to which an organization follows this lifecycle is directly proportional to an organizations process awareness.

- The *(re)design* phase focuses on BP models that capture an enterprise's know how when processing users' requests. All activities in the model are identified along with the control flow and data dependencies. This phase can use the insights and analysis results that were generated from the evaluation phase as its input. This phase is relevant when there is a sort of process awareness in the organization.
- The *configuration/implementation* phase typically transforms the conceptual BP model into a running application. A major task in this phase is the selection of the target execution platform. Also, the configuration of the individual tasks, either automated or manual tasks, takes place in this phase. For instance, configuration parameters must be set if a task is executed via an API/Service invocation. Or, if an activity is performed via SQL script against a database, the SQL statement as well as database connection string must be set. In a less or a non-process-aware organization, the result of this phase can be software applications that hard-code the process logic.
- In the *enactment* phase, BP models are put into production where BP instances are created in response to triggers like receiving users' requests. In this phase, process instances are being monitored, while running, by the management to see if any changes are needed. During this phase too, large amounts of data can be generated. Besides instance evolution events that are emitted by a BPMS, other data can be collected for future analysis. Such data can be IoT related like sensor tracking the temperature of a medical package in a shipment process, free-form text from interactions among process participants, or from interactions with end-customer in a CRM process, etc. Again, the degree of process-awareness and

maturity contributes to the adoption of BPMS. This is called structured process execution. If, however, this is not the case, the process execution is unstructured.

- In the *evaluation* phase, several post execution analyses can take place. This ranges from event correlation and log preprocessing all the way to learning prediction models and suggestions of process improvements. This could lead to a new cycle where the BP model is enhanced. Usually, the main input artifact to this phase is an event log. However, as we discuss later in this section, this scope can be widened to include other data sources that can be collected either at enactment phase or by other data acquisition techniques [5].

BP models play a dominant role in the *(re)design* and *configuration/implementation* phases. They represent the blueprint (schema) for subsequent executable processes. Traditionally, the *design phase* is mainly concerned with modeling processes. These models mainly focus on arranging the control flow of the involved activities. In practice, even for large enterprises, the number of process models in their repositories would not exceed a few hundreds or thousands of models [6]. The disk footprint for such repositories would usually not exceed a few tens or hundreds of Megabytes in size. *Therefore, Big Data processing systems might appear to not have the potential to play a significant role in the context of the design and configuration/implementation phases due to the limited data scale needs.*

In the *enactment* phase, often, massive amounts of *process data* are generated. Traditionally, process data at runtime are just the events that describe the evolution of running cases, e.g., completion of a task within a case possibly with data items produced by that task. Considering this source alone, the generated data may not qualify as being big. However, if we consider the contextual data that co-occur with process evolution, then the data can be qualified as big (e.g., GBs, TBs, and PBs). Contextual data encompasses case evolution events, sensor data, social media interactions, customer interactions, etc. that are within the context of a running process instance. we coin the term *Process Footprint* to define process contextual data. A Process Footprint refers to a variety of process data. A Process Footprint can be at the level of individual instances or process models. A process instance footprint collects every data item that directly or indirectly relates to the instance. These are dark data whose relation to business processes is there but not studied. A Process Footprint may include, the process model (if any) used to run the case, event logs reflecting case evolution, database records that were affected by the case, emails exchanged as a form of interactions within the case, sensor data, customer interaction for feedback, intra-organizational interactions among process participants over enterprise social media, etc. A Process Footprint goes beyond process-enactment generated data.

Figure 2 shows an example Process Footprint which contains process data on three levels: 1) Process model level, 2) Process execution level, and 3) Systems and people level. At the process model level, the main source of data is the process model itself. When this model is configured and then enacted, process instances are created as shown in the middle tier of process execution level. At this level, in case of structured execution, an execution engine handles the orchestration of invoking the different tasks and generating events, that will eventually comprise execution logs, that reflect the evolution of the process instance. Moreover, the engine will handle the invocation of lower level tasks. As shown for the ''Check claim'' task, as an automatic task, the service that checks the claim and store it to the database, thus data-wise, this results in creating a new record to the database of claims. The next task ''Assess claim'' is to be carried out by a human. There might be several communications taking place, e.g. in the form of email exchanges, between the case assessor and some other colleagues. The emails exchanged are considered as part of the process instance footprint. Eventually, the claim record is updated before the task is completed. The ''Decide'' is also a human task. It involves a meeting to decide about the case. But, for this specific instance, the meeting was rescheduled due to a participant being unavailable. This causes delays in completing the task. Without this data, one cannot explain the excess time taken to decide about the claim in that specific case. We argue that all these data items are process related because unless for the process instance being executed, the actions generating this data would not have occurred.

The ideal setting to enact a BP is to have a dedicated execution engine that guarantees that a process instance strictly follows a *pre-defined* BP model, as shown in Figure 2. However, in many real-world scenarios, there is no such execution engine and the enactment phase becomes *unstructured* [7]. There are many reasons that the *unstructured* mode for enacting BP is commonly followed/exists. For example, many enterprises suffer from the lack of process awareness. In such organizations, the administration starting from top-level management down to the operational level is not aware of the notion of a process. Each role has only a local view of what they are doing without an end-to-end view. In addition, in other enterprises the maturity of the organization has not reached that level where all or most of the processes are well known and well configured. It is often that there is no complete or up-to-date documentation of the BP models. Furthermore, in many scenarios, it could be the case that enactment of various tasks or activities of a single BP are supported and recorded in the log files of multiple IT-Systems where the flow among these systems is hard-coded within applications. This *unstructured* mode for BP execution makes tracking process data at enactment time infeasible and the task of collecting process data moves to the evaluation phase.

During the *evaluation* phase, insights on the data generated at the enactment phase form another type of process data. These insight data are the result of various *data-intensive* operations. The generated insights at this phase can be used as feedback to both the design and enactment phases. Moreover, the quality of the derived insights is directly proportional to the amount of Process Footprint we can bring and make

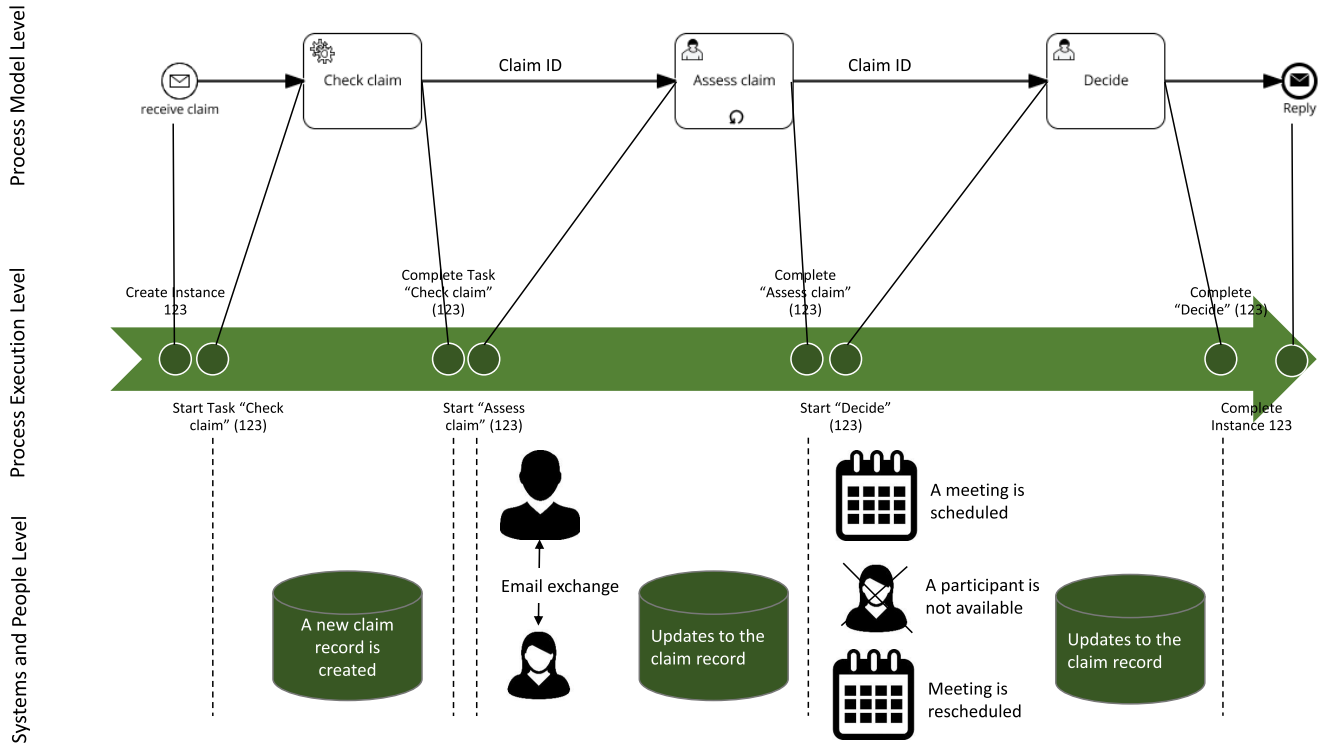

FIGURE 2. An example process footprint.

available for analysis. In the case of structured process execution, large parts of process data can be correlated and retrieved as an identifier that can be passed from one level to a lower level. For instance, as shown in Figure 2, with each newly enacted process instance, a process (case) identifier can be propagated by the execution engine to the invoked task. Such task can pass the identifier further down to its next level of execution etc. However, if the BP was enacted in an unstructured way, then an additional task needs to be done first. This additional task is required to integrate and correlate the scattered events and data of the executed BP. This task is not trivial and seldom fully automated. Assuming the availability of an enterprise data ''lake'', complicated data correlation operations need to take place in order to construct the Process Footprint.

Current efforts for data correlation are concerned just with constructing the event log [8]–[11], which represents the process execution level data in Figure 2. However, such data alone would not explain, for instance why the ''Decide'' task took longer in a specific case. The collection of a further level of data such as data on the systems level, require complicated analysis of data lakes and different forms of correlations to construct the footprint. Clearly, the amount of data to be analyzed and the number of different correlations and transformations to be applied on the data are beyond the capabilities of individual processing nodes and thus Big Data processing is needed.

III. DATA INTENSIVE OPERATIONS IN BUSINESS PROCESS ANALYTICS

As discussed in Section II, in case of a structured BP enactment, Process Footprint is updated continuously as new data are generated by process instances. Unfortunately, this is the exception rather than the norm. As such, this creates the need for several data intensive operations to construct the Process Footprint from scattered sources (event correlation) [9], e.g., into the form of event logs. After this has been completed, several analytics are conducted over the collected log, process discovery, conformance checking, process improvement, performance analysis, extraction of simulation parameters, etc. [5]. Figure 3 illustrates a general classification of these operations. In this section, we provide an overview of various data-intensive process analytics operations [12]. Prior to doing so, we introduce some preliminaries regarding process models and event logs.

Event logs serve as the starting point for process mining and other types of business process analytics. An event log is a *multiset of traces* [13]. Each trace describes the life-cycle of a particular case (aka process instance) in terms of the activities executed. A trace is a finite sequence of events. Each event represents an evolution of the lifecycle of one of the activities of the process. For instance, when a task is started, an event is generated that conveys that this specific activity of that specific process instance has been started with a timestamp indicating the event time. When the activity

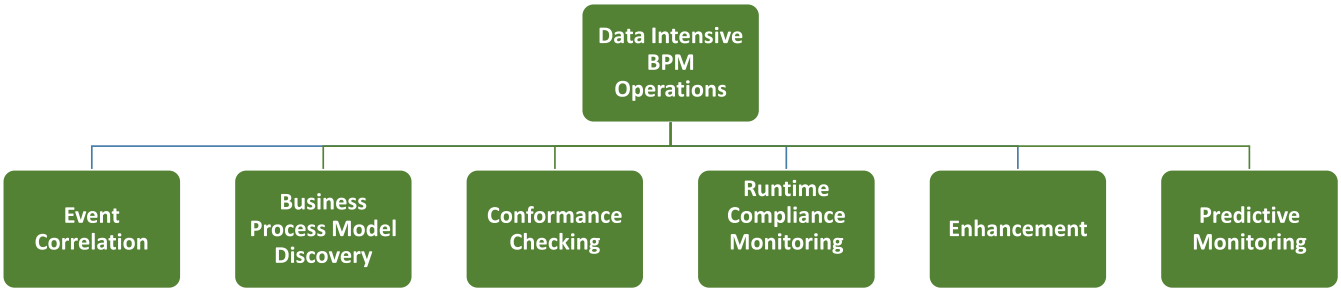

FIGURE 3. General classification of BPM data-intensive operations.

completes, a respective event is generated. Process models are represented as sets of traces.

Definition 1 (Possible Traces): $\mathcal{A}$ is the universe of *activities*, i.e., the set of all possible and relevant activities. Other activities cannot be observed (or are abstracted from). Elements of $\mathcal{A}$ may have *attributes*, e.g., costs, resource information, duration information, etc. A *trace* $\sigma \in \mathcal{A}^*$ is a sequence of activities found in an event log or corresponding to a run of a particular process model. $\mathcal{U} = \mathcal{A}^*$ is the universe of all possible traces over $\mathcal{A}$.

The simple definition abstracts from event and case *attributes* [5]. Events and cases can have any number of attributes. Events have timestamps such that they can be (partially) ordered in time. Next to the activity name and timestamp, typical examples of event attributes include resource, location, customer, type, and costs.

In a process model a specific trace $\sigma \in \mathcal{U}$ is either possible or not. Hence, a model can be characterized by its set of allowed traces.

Definition 2 (Process Model): A *process model* M is a non-empty collection of traces, i.e., $M \subseteq \mathcal{U}$ and $M \neq \emptyset$.

$A_M = \bigcup_{\sigma \in M} \{a \in \sigma\}$ is the set of activities possible in M.

An event log is a *multiset* of *sample* traces from a known or unknown process. The same trace can appear multiple times in the log. Moreover, the event log contains only example behavior. Often only a few of the possible traces are observed [5]. For example, in case of concurrency, we cannot expect to see all interleavings. Also, note that a loop in a process model describes an infinite set of possible traces.

Definition 3 (Event Log): An *event log* $L \in \mathcal{B}(\mathcal{U})$ is a multiset of observed traces. $L(\sigma)$ denotes the number of times that trace σ appears in L.

$A_L = \bigcup_{\sigma \in L} \{a \in \sigma\}$ is the set of activities occurring in L. For example, $L = [\langle a, b, d, e, g\rangle^5, \langle a, c, d, e, h\rangle^4, \langle a, b, d, e, f, c, d, e, g\rangle]$ is an event log containing 10 traces, e.g., $L(\langle a, b, d, e, g\rangle) = 5$, so 5 cases followed the path $\langle a, b, d, e, g\rangle$.

Defining event logs and process models as respectively multisets of observed behavior ($L \in \mathcal{B}(\mathcal{U})$) and sets of modeled behavior ($M \in \mathcal{P}(\mathcal{U})$) helps relate both. For example, process discovery can be viewed as the task to convert $L \in \mathcal{B}(\mathcal{U})$ into $M \in \mathcal{P}(\mathcal{U})$ where L only contains example behavior and may include outliers (i.e., behaviors one may want to abstract from). An observed trace $\sigma \in L$ such that $\sigma \notin M$ can be seen as a deviation from the model.

A. EVENT CORRELATION

This operation is of utmost importance in the context of unstructured process enactment. Current efforts are oriented towards grouping events into cases [8], [10]. With respect to collecting the Process Footprint, this represents a sub-task of constructing the event log. More challenging tasks are concerned with correlating data at the lower level, i.e., the system and people level as shown in Figure 2. This requires investigating a large space of potential correlation conditions between the attributes of the different data sets. Event correlation becomes more challenging if the BP executions span over several heterogeneous systems, services, or business functions [9]. In this case, details about BP instances can be scattered across multiple, potentially heterogeneous, data sources (i.e., logs). In addition, various information of the same objects in the different systems need to be identified, correlated, and integrated. Event correlation is foundational for the subsequent process analytics and takes place typically at the process evaluation phase.

B. PROCESS MODEL DISCOVERY

In the context of unstructured enactment of BPs, *discovery* operations are applied to automatically generate BP models by analyzing the log, which is generated during the execution of BP instances [5]. In particular, these techniques receive the event log as input and produce a discovered business process model without using any a-priori information. The resulting model depends on the discovery algorithm as well as the quality and coverage of the log to different scenarios [5]. There is no one silver bullet discovery approach. Rather, different algorithms are run and results are discussed with stakeholders until an acceptance is reached. Process model discovery is typically run at the process evaluation phase of the process lifecycle.

C. CONFORMANCE CHECKING AND COMPLIANCE MONITORING

Conformance checking is a business process analysis technique that compares a pre-defined BP model to its event log with the objective of ensuring whether the real-world process

execution conforms to the defined model [5]. In particular, *conformance checking* associates events in the log with activities in the BP model with the aim of identifying the discrepancies and commonalities among the modeled and executed behaviors. Therefore, conformance checking serves as a sort of *auditing* that is performed *offline* for diagnostic purposes and commonly conducted on a large number of carried out process instances, following a *retrospective* (after-the-fact) mechanism. Another operation that attempts to achieve the same objective, however, in an *online* mode (i.e. at enactment) is *runtime compliance monitoring* [14]. In this scenario, process events arrive in streams and are analyzed for compliance with prescribed rules. The key value here is to detect or even predict the possibility of a violation as soon as it occurs. Delays or non-detection can impose harmful penalties on the organization. In principle, these are quite vital operations as, practically, only a subset of compliance rules can be checked during design-time due to the lack of necessary contextual information (e.g., time, actors, roles, and order). Compliance monitoring assumes a structured process execution where correlated events are generated by a process execution engine.

D. BUSINESS PROCESS ENHANCEMENT

Enhancement is another technique that revises, improves, or extends an already defined BP model using the recorded events about the actual BP instances in the system logs in order to better reflect reality [5]. In principle, enhancement techniques that focus on analyzing the event logs with the aim of understanding the scope and execution flow of a BP is a challenging undertaking. In addition, it can turn to be a highly subjective process that depends on the perspective of the person looking at BP execution traces. For example, for one type of enhancement techniques, *repair*, focuses on modifying the business process model to better reflect reality. For instance, if 2 activities are modeled sequentially but in reality it can happen in any order, then the model may be corrected to reflect this. Another type of enhancement technique is *extension* which focuses on adding a new perspective to the process model by cross-correlating it with the log. In practice, enhancement techniques usually rely on process-centric abstractions and query languages that enable the querying, exploration, and visualization of integrated views for the process event relationships and execution data [15]. The suggested enhancements can be discovered *offline* by analyzing the event logs.

E. PREDICTIVE MONITORING

Predictive business process monitoring techniques are designed to focus on forecasting potential problems for running process instances by finding a similarity or an alignment with previously completed cases. For this purpose, predictive monitoring has an offline part that analyzes the log of completed cases to extract the knowledge about the monitoring objective. For example, learning which process instances failed to complete or completed but was overdue, etc. This knowledge is then used online to monitor running cases by finding similarities of the process instance execution path with historical cases as early as possible in order to handle potential problems *proactively* [16]. In this context, proactive behavior means that once violations are detected, appropriate recovery action(s) are automatically planned to mitigate their impacts. For instance, if a delay in delivery time is predicted during the execution of a freight transport BP, a faster means of transport or alternative transport routes could be scheduled proactively before the delay occurs [16].

Usually, machine learning approaches (e.g., Bayesian networks and neural networks) are used to offline build the knowledge and the model of process behavior. To this extent, the input used in training may go beyond process logs. For instance, feedback from customers as well as process participants' feedback can be added to gain more insight. This data are usually unstructured as it can refer to natural language text that need to be processed and linked to the respective process instance before running machine learning tasks [17].

We would like to note that the classification shown in Figure 3 shows the main areas of data-intensive BPM operations. It is not meant to be comprehensive nor complete, in the sense that we do not enumerate all operations in each area. In our context, we have been mainly focusing on some of the data-intensive process mining operations. However, there are various process mining techniques such as those which are used for predicting the remaining processing time of running BP instances [5] or others which focus on learning factors that influence decisions and the discovery of roles are integral parts of process mining. For a comprehensive overview for process mining techniques, we refer readers to [5].

IV. BIG DATA PROCESSING SYSTEMS

Big Data refers to the explosive and tremendous increase of digital data generated from various sources. This phenomenon has invited the research communities to revisit their scientific methods and techniques. Experiments, study of theorems and laws, and simulation were in a chronological way the first approaches. In 2007, Jim Gray, the Turing Award winner, separated data-intensive science from computational science. Gray believed that the fourth paradigm is not only a change in the way of scientific research, but, also a change in the way that people think [18].

In practice, the Hadoop framework has pioneered the domain of general purpose data analytics systems. For about a decade, Hadoop has been recognized as the defacto standard of Big Data analytics and was popularly employed as an effective solution that can harness the resources and power of large computing clusters in various application domains [19]. However, recently, both the research and industrial communities identified various limitations in Hadoop [19] and, thus, Hadoop cannot be the *one-size-fits-all* solution for the various Big Data analytics challenges. In particular, one of the main performance limitations of Hadoop is that it materializes the intermediate results of each Map or Reduce step, of its parallel execution, on Hadoop Distributed File System (HDFS) before starting the next one. This undermines

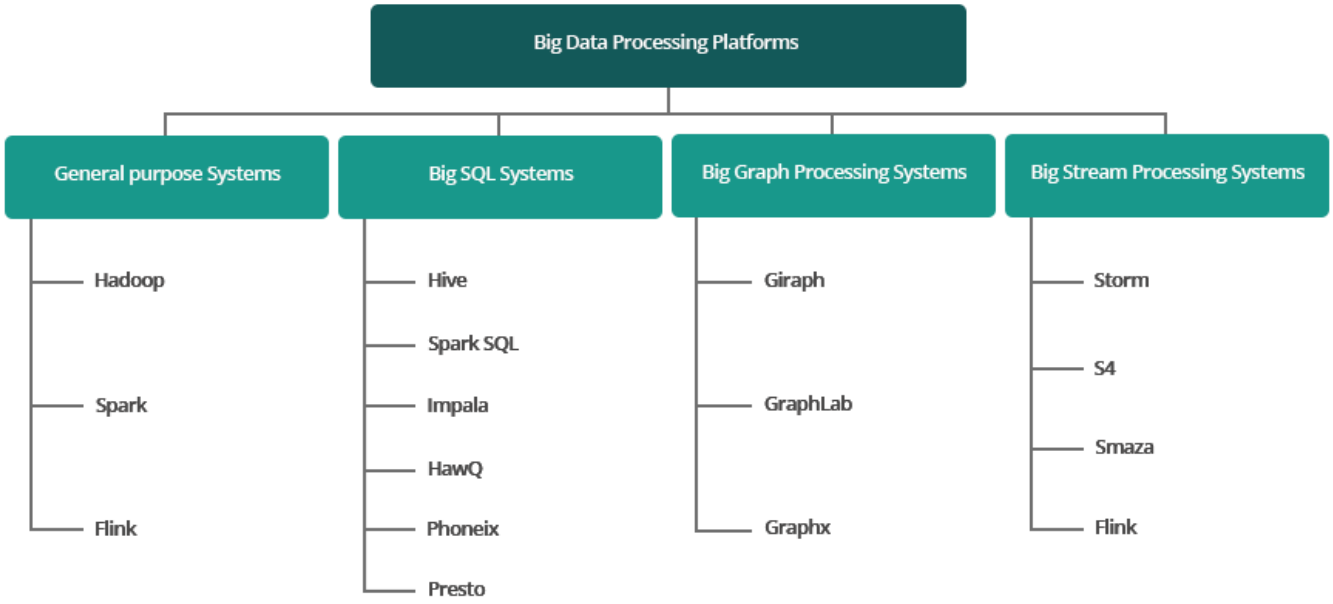

FIGURE 4. General classification of Big Data processing systems.

the performance of data processing that is iterative in nature (e.g., mining operations and graph processing) or require near real-time processing capabilities (e.g., data streams). Therefore, in the last few years, Hadoop has been slowly replaced by a collection of engines, which are dedicated to specific verticals and referred to as Big Data 2.0 processing systems [4]. Figure 4 illustrates a general classification for the Big Data processing systems.

A. GENERAL-PURPOSE SYSTEMS

Apache Spark project[1] has been introduced as a general-purpose Big Data processing engine that takes the concepts of Hadoop to the next level by loading the data in distributed memory and relying on less expensive shuffles during the data processing. In particular, Spark adopts the Resilient Distributed Datasets (RDD) abstraction that is an in-memory data structure that enables user-defined jobs to exploit the loaded data into a cluster's memory and query it repeatedly [20]. Users can also explicitly cache an RDD in-memory across machines and reuse it in multiple parallel operations. It provides various packages with higher-level libraries including support for SQL queries [21], streaming data, machine learning [22], statistical programming, and graph processing [23]. These libraries increase developer productivity and can be seamlessly combined to create complex workflows. In addition, Spark provides APIs for various programming languages including Scala, Java, Python, and R. In the 2014 annual Daytona Gray Sort Challenge,[2] which benchmarks the speed of data analysis systems, Spark has strongly outperformed Hadoop and was able to sort through 100 terabytes of records within 23 minutes while Hadoop took about 72 minutes, which is over three times as long to execute the same task. Currently, Spark has over 500 contributors from more than 200 organizations making it the most active project in the Apache Software Foundation and among other Big Data open source projects. Popular distributors of the Hadoop ecosystem (e.g., Cloudera, Hortonworks and MapR) are currently including Spark in their releases.

Apache Flink[3] is another distributed in-memory data processing engine, which represents a flexible alternative for Hadoop that supports both batch and realtime processing [24]. Instead of Hadoop's map and reduce abstractions, Flink uses a directed graph that leverages in-memory storage for improving the performance of the runtime execution. Flink is, also, equipped with *Flink Streaming*[4] API as an extension of the core Flink API for high-throughput and low-latency data stream processing. The system can connect to and process data streams from various data sources where data streams can be transformed and modified using high-level functions that are similar to those provided by the batch processing API. In addition, recently, the Flink open source community has developed libraries for machine learning (FlinkML)[5] and graph processing (Flink Gelly).[6]

B. BIG SQL ENGINES

In the context of large-scale processing of structured data, various studies reported on Hadoop's limitations. Indeed, it is inadequate for interactive queries that target response times of

[1] http://spark.apache.org/
[2] http://sortbenchmark.org/
[3] https://flink.apache.org/
[4] http://ci.apache.org/projects/flink/flink-docs-release-0.7/streaming_guide.html
[5] http://ci.apache.org/projects/flink/flink-docs-master/libs/ml/
[6] https://cwiki.apache.org/confluence/display/FLINK/Flink+Gelly/

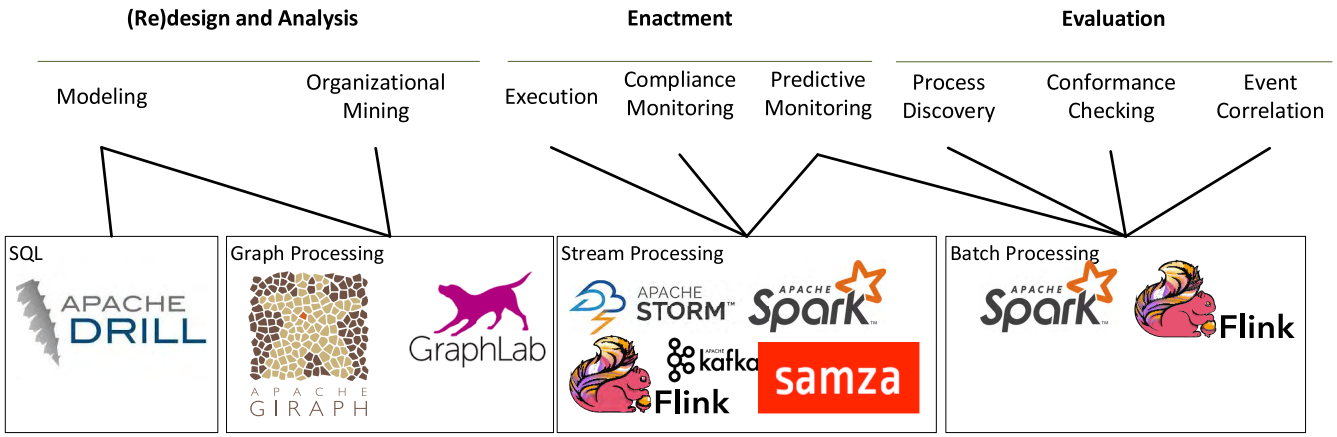

FIGURE 5. BPM operations meet big data systems.

milliseconds or a few seconds [25]. In addition, many programmers could be unfamiliar with the MapReduce framework and they prefer to use SQL (in which they are more proficient) as a high level declarative language to express their task while leaving all of the execution optimization details to the backend engine. Therefore, several systems (e.g., **Apache Hive** [26], **Google Big Query**, **Cloudera Impala** [27], **Facebook Presto**, and **Spark SQL** [21]) have been introduced to provide SQL flavor on top of the Big Data processing platforms and support interactive querying and analysis of large scale structured data.

C. BIG GRAPH PROCESSING ENGINES

Graph is a natural, neat and flexible structure to model the complex relationships, interactions, and interdependencies between objects. In particular, each graph consists of nodes (or vertices) that represent objects and edges (or links) that represent the relationships among the graph nodes. In principle, graph analytics is an important and effective Big Data discovery tool. For instance, it enables identifying influential persons in a social network, inspecting fraud operations in a complex interaction network and recognizing product affinities by analyzing community buying patterns. However, with the enormous growth in graph sizes, huge amounts of computational power would be required to analyze such massive graph datasets. In practice, the Hadoop framework has been discarded within the context of large-scale graph processing [19]. Therefore, in 2010, Google introduced the **Pregel** system [28], open-sourced by Apache Giraph,[7] that uses a Bulk Synchronous Parallel (BSP) based programming model for efficient and scalable processing of massive graphs on distributed clusters of commodity machines. Various other systems (e.g., **GraphX** [29], **GraphLab** [30], and **Trinity** [31]) were developed to efficiently tackle the problem of large-scale graph analytics.

D. BIG STREAM PROCESSING ENGINES

Stream computing is a new paradigm that responds to a new generation of data-intensive systems, such as ubiquity of mobile devices, location services, and sensor pervasiveness. In particular, as the world gets more instrumented and connected, there is flood of digital data that is getting generated from several hardware (e.g., sensors) or software in the format of flowing stream of data. Examples of this phenomenon are crucial for several applications and domains including financial markets, surveillance systems, manufacturing, smart cities and scalable monitoring infrastructure. In all these applications and domains, there is a crucial requirement to collect, process and analyze big streams of data in order to extract valuable information, discover new insights in real-time. Therefore, in order to tackle the challenge of large-scale processing of streaming data, Twitter released the **Apache Storm** system[8] that supports a distributed and fault-tolerant platform for implementing continuous and real-time processing applications of streamed data. Other systems that have been introduced in this domain include **Apache S4**, **Apache Samza**, **Spark streaming** and **Flink streaming** [24].

V. WHERE CAN PROCESS MANAGEMENT AND BIG DATA SYSTEMS MEET UP?

In this section we elaborate on how BPM can benefit from Big Data techniques, see Figure 5. We divide this discussion based on the different process lifecycle phases presented in Figure 1 as well as the process analytics operations discussed in Section III.

A. PROCESS (RE) DESIGN

The process *(re-) design* phase can benefit from process data and insights gained from other analytics to improve process models. Improvement can be applied to the process model structure, e.g., adding, removing or moving process tasks or their reordering, to simulation runs of the

[7] https://giraph.apache.org/

[8] https://storm.apache.org/

process based on more accurate measures for performance elements like tasks durations, or to validation experiments based on input from actual execution insights. Most of these improvements are built on analytics results from the *evaluation* phase. However, there are analytics techniques that are applied on process artifacts at the design phase. These analytics are related to *process querying* techniques [32]. One of the first works on large-scale process querying is presented Yang *et al.* [33] utilize the parallel processing nature provided by cloud computing and big data systems to construct so-called path indexes over process models to enhance efficient retrieval. Technologies like Apache Drill [9] seem promising to support the notion of ad-hoc process querying.

Other use cases at process design time are related to recommendation of tasks while modeling. Recommendation depends on other aspects on top of the control flow structure of similar processes [34]. This includes, semantics of the task labels, meta information, e.g., performance indicators, discussions from previous collaborations on process modeling, feedback from users, i.e. parts of the Process Footprint. All such data when combined can be classified as Big Data considering the size and its degree of (un-)structuredness. Thus, process modeling recommender systems can benefit a lot from analytics run on such sources for a better-informed recommendation. Such analytics are typically run in a batch mode using either the MapReduce computational model or the Bulk-Synchronous-Parallel model [35]. The choice of which depends on how the data to be analyzed is represented. In the former case, data are arranged in a variant of key-value pairs. In the latter, the data are arranged in graphs where connections represent relations among the vertices. Examples of vertices could be tasks, process participants, roles, organizations etc. This use case overlaps with the business process model enhancement operation discussed in Section III.

B. ENACTMENT

The need for a distributed execution environment of business processes has been acknowledged recently [36]. This is further fueled by the growing support and integration of IoT applications by business processes. This challenges currently available execution engines scalability. There are performance benchmarking studies of open source centralized execution engines [37], [38]. Evidently, the scale discussed here is beyond the scope of central execution engines. Cloud Process Execution Engine (CPEE) [39] is an open source process execution engine that claims to be ''cloud-ready'' and fully event-based. Unfortunately, neither scientific papers presenting CPEE nor its documentation elaborate on its architecture or how events are processed. Being cloud-ready appears to be meant to provide a set of Restful APIs to communicate with the engine. The installation guidelines do not describe any distributed settings or cluster-based deployment of CPEE. Yet, it is considered as a first step towards distributed process execution.

Big stream processing technologies like Apache Storm, Flink, Spark streams, etc., along with message queuing systems like Apache Kafka can form a basis for a distributed event-based process execution environment. Process models' control and data flow can be translated into stream processing applications on these systems, a related discussion can be found in [40]. All types of process enactment data, see Section II, can be collected and processed as they are generated with scalability as an out-of-the-box feature of the underlying processing systems. Moreover, this helps streamline the construction and maintenance of the Process Footprint, as data will be correlated at its generation time. Having the process execution engine built as an event-driven system provides seamless integration of different extensions and applications like compliance monitoring, e.g. [14], [41], and [42], different prediction scenarios [36], [43], online conformance checking [44], [45] or support for advanced process constructs that continuously consume events [46], [47]. We acknowledge this topic as an open research area.

C. EVALUATION

The evaluation phase, see Figure 1, of the business process lifecycle is probably the most benefiting phase from Big Data technology. Most of the data intensive operations discussed in Section III are typically run in this phase. In the case of unstructured process execution, event correlation to construct the Process Footprint is one major task that can benefit from big data technology. As the data should already be stored in a data lake, batch processing techniques are the candidates to implement data correlation pipelines. These can be tackled on modern systems like Spark or Flink. Sophisticated multi-level correlation pipelines are needed as data has to be partitioned across the different nodes of a computing cluster.

Business process mining techniques have been one of the main topics on the annual programme of the primary scientific conference of the Business Process Management (BPM) research community over many years. Moreover, A variety of process mining tools (e.g., ProM, Disco, Celonis, and minit) and packages are made available for BP engineers. For instance, ProM [48] is a popular business process mining framework whose development started in 2003 and now has more than 1500 plugins implementing various process mining techniques. However, the majority of these techniques use conventional computation models and do not apply novel scalable and distributed techniques [49].

Distributed implementations of process mining techniques necessarily require partitioning the event log among the nodes of the computing cluster so that each node becomes responsible for processing a part of the event data. In principle, there are two main approaches for partitioning the event log [5], [50]:

1) *Case-based partitioning* (vertical partitioning) which distributes events based on the case of which they belong so that each case is assigned to one node.

[9] https://drill.apache.org/

2) *Activity-based partitioning* (horizontal partitioning) which distributes events based on the activities to which they refer so that cases are split up into smaller traces working on subsets of activities. Therefore, each compute node participates on processing all cases.

Some approaches are taking advantage of Big Data technologies for process mining that do not explicitly consider or require log partitioning described above. These approaches include: Reguieg *et al.* [51] who introduced a MapReduce-based algorithm for discovering event correlations from big event logs, Evermann [52] who presented a MapReduce-based implementations of two popular process mining algorithms, Hernández *et al.* [53] who proposed an integration framework between ProM and Hadoop, Meddah *et al.* [54], who provided a MapReduce-based pattern mining from large event logs. The common factor among these approaches is that they are Hadoop-based. In practice, with the recent advancements on Big Data processing systems, Hadoop is not the only representative state-of-the-art Big Data system, nowadays. In addition, different limitations in the architecture of the Hadoop framework might make it an inadequate choice for the *iterative*, *real time* or *performance* requirements of the various BPM data intensive operations. In particular, several other Big Data processing engines have been introduced to advance this domain and can be effectively exploited in various means. For instance, in the MapReduce-based implementation of mining algorithms in [52], the map step may produce several identical tuples, representing the ordering relation between pairs of activities, that would not affect the mined model but will result in extra communication and materialization overheads between mapper and reducer nodes. If, however, the algorithm would have been implemented in a modern Big Data processing system, e.g. Spark, the approach would have benefited from stateful map operators and only unique tuples are produced.

Recently, approaches that make use of state-of-the-art batch processing systems, e.g. Spark, have been presented. Cheng *et al.* [55] presented an efficient approach for event correlation. The source of efficiency is two-fold. Firstly, the use of Spark guarantees better runtime performance over Hadoop. Secondly, the algorithm follows a notion of filter-and-verification where non-interesting correlations are identified early enough and abandoned. The key point is that the algorithm design is aware of the cost of transferring data over the network and thus minimizes it. Another recent approach that considers explicit log partitioning is presented in [56]. The approach addresses online conformance checking of events as they occur. The approach depends on Apache Spark, thus is able to support both stream and batch conformance checking.

In conclusion, with the wide spectra on both sides of data intensive operations of BPM and the Big Data processing systems, it has become quite crucial to make the right decisions on exploiting the adequate Big Data system for effectively tackling and providing a scalable implementation of a specific process data-intensive operations. However, general guidelines in this context can be recommended as follows:

- Most of the approaches that have been presented for exploiting Big Data processing engines in the context of business process analytics have been mainly relying on the Hadoop framework. However, as stated earlier, one of the main limitations of Hadoop is its inefficiency for performing iterative tasks. Hence, due to the iterative nature of the *process discovery*, *event correlation*, and *enhancement* techniques, it would be more natural to rely on other main memory-based Big Data processing engines with intuitive support for iterative processing such as Spark and its machine learning extension, MLlib.[10] Apache Mahout,[11] SystemML[12] and bigml[13] represent other examples of new emerging systems that have been designed for providing scalable machine learning solutions on top of the various Big Data processing engines and can be also effectively utilized in this context.
- Due to the natural graph-based structure of business process models, large scale graph processing systems (e.g., Giraph, GraphLab and GraphX) represent important solutions that can be effectively exploited for implementing scalable graph mining and graph analytics techniques [57]–[59] for discovering BP models from the event logs. These systems can be also effectively utilized to analyze large graphs of organizational social networks [5] that connect the resources of the organization with the organizational entities (e.g., departments, groups, roles). Utilizing the known fact that event logs contains references to the human resource that have performed a certain activity within a case, the interconnectivity among those resources/activities can be represented as graphs where vertices are the resources and activities and edges connecting a resource vertex to an activity vertex represent the *performed by* relationship whereas edges from a resource to another resource might represent the *co-working* relationship between resources. Such edges can have attributes reflecting the frequency of such co-working relationship. Several interesting analytics use cases over such graph have emerged. For instance, finding (sub) teams among performers of parts of the process can help to further refine performance analysis conducted on the logs and might also help to reveal why some of the individual resource perform differently even when doing the same activity.
- *Conformance checking* techniques mainly focus on querying the event logs, which are usually structured in nature, with the aim of detecting any existing deviations during the process instance execution. Therefore, in this context, exploiting processing engines for large scale

[10]https://spark.apache.org/docs/latest/mllib-frequent-pattern-mining.html

[11]http://mahout.apache.org/

[12]http://researcher.watson.ibm.com/researcher/view_group.php?id=3174

[13]https://bigml.com/

structured data (e.g., Hive, Impala, and Spark SQL) represent an adequate approach for effectively tackling this challenge.

- While the *process model discovery*, *event correlation* and *enhancement* techniques are *offline* and can be performed by batch processing engines (e.g, Hadoop and Spark), other techniques such as *runtime compliance monitoring* and *proactive business process monitoring* are *online* and *realtime*(cf. [60], [61]). Therefore, for these techniques, large scale stream processing engines (e.g., Flink, Storm) would provide the adequate support for implementing scalable and efficient solutions that can tackle these problems. In particular, these systems would provide the ability of online processing for the generated stream of events during the business process execution.
- Most of the process-analytics approaches are concerned with a subset of process data that are structured by nature, i.e. event logs. The notion of Process Footprint as described in Section II is still beyond the scope of current analytics approaches. Yet, available Big Data processing systems provide the technical capabilities to help design algorithms that collect such footprint data from their heterogeneous sources and further process them for more insights.

VI. CONCLUSION

In this article, we highlighted several business process data intensive operations along the process lifecycle. Moreover, we discussed Big Data systems-based solutions to a subset of those analytics operations. We acknowledge that very recently, Big Data systems-based solutions have started to appear. We discussed further improvements for such approaches. Many of the outlined use cases qualify as Big Data problems with no scalable solutions, yet.

We have coined up the notion of *Process Footprint* and discussed how big data systems can help collect that footprint and serve as an infrastructure to analyze the collected data with the possibility of being able to derive insights beyond the capability of existing state-of-the-art process analytics techniques. This opens new challenges for process analytics and forms an important research direction. Finally, we sketched a roadmap and provided some recommendations for improving the state-of-the-art on harnessing the power of Big Data technologies in the business process analytics domain. We hope that this will inspire the reader to bring business process analytics to the next level.

ACKNOWLEDGMENT

The work of Sherif Sakr and Ahmed Awad is funded by the European Regional Development Funds via the Mobilitas Plus programme (grant MOBTT75).

REFERENCES

[1] M. Dumas, M. L. Rosa, J. Mendling, and H. A. Reijers, *Fundamentals of Business Process Management*, 2nd ed. Cham, Switzerland: Springer, 2018.
[2] J. Manyika *et al.*, "Big data: The next frontier for innovation, competition, and productivity," Tech. Rep., 2011. [Online]. Available: https://www.mckinsey.com/business-functions/digital-mckinsey/our-insights/big-data-the-next-frontier-for-innovation
[3] J. Dean and S. Ghemawa, "MapReduce: Simplified data processing on large clusters," in *Proc. OSDI*, 2004, pp. 107–113.
[4] S. Sakr, *Big Data 2.0 Processing Systems: A Survey*. Cham, Switzerland: Springer, 2016.
[5] W. M. P. van der Aalst, *Process Mining—Data Science in Action*. Cham, Switzerland: Springer, 2016.
[6] M. Remco Dijkman, M. L. Rosa, and H. A. Reijers, "Managing large collections of business process models—Current techniques and challenges," *Comput. Ind.*, vol. 63, no. 2, pp. 91–97, 2012.
[7] U. Dayal, M. Hsu, and R. Ladin, "Business process coordination: State of the art, trends, and open issues," *VLDB J.*, vol. 1, pp. 3–13, Sep. 2001.
[8] D. Bayomie, A. Awad, and E. Ezat, "Correlating unlabeled events from cyclic business processes execution," in *Proc. 28th Int. Conf. Adv. Inf. Syst. Eng. (CAiSE)* in Lecture Notes in Computer Science, Ljubljana, Slovenia, vol. 9694. Cham, Switzerland: Springer, Jun. 2016, pp. 274–289.
[9] H. R. Motahari-Nezhad, R. Saint-Paul, F. Casati, and B. Benatallah, "Event correlation for process discovery from Web service interaction logs," *VLDB J.*, vol. 20, no. 3, pp. 417–444, 2011.
[10] S. Pourmirza, R. Dijkman, and P. Grefen, "Correlation miner: Mining business process models and event correlations without case identifiers," *Int. J. Cooperat. Inf. Syst.*, vol. 26, no. 2, pp. 1–32, 2017.
[11] A. A. Andaloussi, A. Burattin, and B. Weber, "Toward an automated labeling of event log attributes," in *Enterprise, Business-Process and Information Systems Modeling*. Cham, Switzerland: Springer, 2018, pp. 82–96.
[12] S.-M.-R. Beheshti *et al.*, *Process Analytics: Concepts and Techniques for Querying and Analyzing Process Data*. Cham, Switzerland: Springer, 2016.
[13] M. de Leoni, W. M. P. van der Aalst, and M. Dees, "A general process mining framework for correlating, predicting and clustering dynamic behavior based on event logs," *Inf. Syst.*, vol. 56, pp. 235–257, Mar. 2016.
[14] L. T. Ly, F. M. Maggi, M. Montali, S. Rinderle-Ma, and W. M. P. van der Aalst, "Compliance monitoring in business processes: Functionalities, application, and tool-support," *Inf. Syst.*, vol. 54, pp. 209–234, Dec. 2015.
[15] S.-M.-R. Beheshti, B. Benatallah, H. R. Motahari-Nezhad, and A. S. Sakr, "A query language for analyzing business processes execution," in *Business Process Management*. Cham, Switzerland: Springer, 2011, pp. 281–297.
[16] A. Metzger *et al.*, "Comparing and combining predictive business process monitoring techniques," *IEEE Trans. Syst., Man, Cybern. Syst.*, vol. 45, no. 2, pp. 276–290, Feb. 2015.
[17] A. Ayoub and A. Elgammal, "Utilizing Twitter data for identifying and resolving runtime business process disruptions," in *CoopIS* in Lecture Notes in Computer Science. Cham, Switzerland: Springer, 2018.
[18] T. Hey, S. Tansley, and K. Tolle, Eds., *The Fourth Paradigm: Data-Intensive Scientific Discovery*. New York, NY, USA: Microsoft Research, Oct. 2009.
[19] S. Sakr, A. Liu, and A. G. Fayoumi, "The family of mapreduce and large-scale data processing systems," *ACM Comput. Surv.*, vol. 46, no. 1, p. 11, 2013.
[20] M. Zaharia, M. Chowdhury, M. J. Franklin, S. Shenker, and I. Stoica, "Spark: Cluster computing with working sets," in *Proc. 2nd USENIX Workshop Hot Topics Cloud Comput. (HotCloud)*, Boston, MA, USA, Jun. 2010, p. 10.
[21] M. Armbrust *et al.*, "Spark SQL: Relational data processing in spark," in *SIGMOD*, 2015, pp. 1383–1394.
[22] E. R. Sparks *et al.*, "MLI: An API for distributed machine learning," in *Proc. ICDM*, 2013, pp. 1187–1192.
[23] S. Venkataraman *et al.*, "SparkR: Scaling R programs with spark," in *Proc. Int. Conf. Manage. Data, SIGMOD Conf.*, San Francisco, CA, USA, Jun./Jul. 2016, pp. 1099–1104.
[24] P. Carbone, A. Katsifodimos, S. Ewen, V. Markl, S. Haridi, and A. K. Tzoumas, "Apache Flink: Stream and batch processing in a single engine," *IEEE Data Eng. Bull.*, vol. 38, no. 4, pp. 28–38, Dec. 2015.
[25] A. Pavlo *et al.*, "A comparison of approaches to large-scale data analysis," in *Proc. SIGMOD*, 2009, pp. 165–178.
[26] A. Thusoo *et al.*, "Data warehousing and analytics infrastructure at Facebook," in *Proc. SIGMOD*, 2010, pp. 1013–1020.
[27] M. Kornacker *et al.*, "Impala: A modern, open-source SQL engine for Hadoop," in *Proc. CIDR*, 2015.

[28] G. Malewicz *et al.*, "Pregel: A system for large-scale graph processing," in *Proc. SIGMOD*, 2010, pp. 135–146.
[29] J. E. Gonzalez, R. S. Xin, A. Dave, D. Crankshaw, M. J. Franklin, and I. Stoica, "GraphX: Graph processing in a distributed dataflow framework," in *Proc. OSDI*, 2014, pp. 599–613.
[30] Y. Low, D. Bickson, J. Gonzalez, C. Guestrin, A. Kyrola, and J. M. Hellerstein, "Distributed GraphLab: A framework for machine learning and data mining in the cloud," *VLDB Endowment*, vol. 5, no. 8, pp. 716–727, 2012.
[31] B. Shao, H. Wang, and Y. Li, "Trinity: A distributed graph engine on a memory cloud," in *Proc. SIGMOD*, 2013, pp. 505–516.
[32] A. Polyvyanyy, C. Ouyang, A. Barros, and W. M. P. van der Aalst, "Process querying: Enabling business intelligence through query-based process analytics," *Decis. Support Syst.*, vol. 100, pp. 41–56, Aug. 2017.
[33] L. Yang, S. Li, W. Chen, K. Lei, and T. Wang, "PBI: A path-based bitmap index for efficient process analysis in cloud computing environment," in *Proc. IEEE 3rd Int. Conf. Big Data Secur. Cloud (BigDataSecurity), IEEE Int. Conf. High Perform. Smart Comput. (HPSC), IEEE Int. Conf. Intell. Data Secur. (IDS)*, May 2017, pp. 57–62.
[34] M. Fellmann, N. Zarvic, D. Metzger, and A. Koschmider, "Requirements catalog for business process modeling recommender systems," in *Proc. Int. Tagung Wirtschaftsinformatik (WI) Smart Enterprise Eng.*, Osnabrück, Germany, Mar. 2015, pp. 393–407.
[35] D. Wu, S. Sakr, and L. Zhu, "Big data programming models," in *Handbook of Big Data Technologies*. Cham, Switzerland: Springer, 2017, pp. 31–63.
[36] M. Borkowski, W. Fdhila, M. Nardelli, S. Rinderle-Ma, and A. S. Schulte, "Event-based failure prediction in distributed business processes," *Inf. Syst.*, to be published. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0306437917300030#!
[37] V. Ferme, A. Ivanchikj, and C. Pautasso, "A framework for benchmarking BPMN 2.0 workflow management systems," in *Business Process Management*. Cham, Switzerland: Springer, 2015, pp. 251–259.
[38] S. Harrer, *Effective and Efficient Process Engine Evaluation*, vol. 25. Bamberg, Germany: Univ. Bamberg Press, 2017.
[39] J. Mangler and S. Rinderle-Ma, "CPEE—Cloud process execution engine," in *Proc. BPM Demo Sessions Co-Located 12th Int. Conf. Bus. Process Manage. (BPM)*, Eindhoven, The Netherlands, Sep. 2014, p. 51. CEUR-WS.org, 2014.
[40] P. Soffer *et al.*, "From event streams to process models and back: Challenges and opportunities," *Inf. Syst.*, to be published. [Online]. Available: https://www.sciencedirect.com/science/article/abs/pii/S0306437917300145
[41] A. Barnawi, A. Awad, A. Elgammal, R. E. Shawi, A. Almalaise, and A. S. Sakr, "Runtime self-monitoring approach of business process compliance in cloud environments," *Cluster Comput.*, vol. 18, no. 4, pp. 1503–1526, 2015.
[42] A. Awad, A. Barnawi, A. Elgammal, R. Shawi, A. Almalaise, and A. S. Sakr, "Runtime detection of business process compliance violations: An approach based on anti patterns," in *Proc. 30th Annu. ACM Symp. Appl. Comput.*, Salamanca, Spain, Apr. 2015, pp. 1203–1210.
[43] I. Teinemaa, M. Dumas, F. M. Maggi, and C. D. Francescomarino, "Predictive business process monitoring with structured and unstructured data," in *Proc. 14th Int. Conf. Bus. Process Manage. (BPM)* in Lecture Notes in Computer Science, vol. 9850. Rio de Janeiro, Brazil: Springer, Sep. 2016, pp. 401–417.
[44] M. A. Iman Helal, A. Awad, and A. E. Bastawissi, "Runtime deduction of case ID for unlabeled business process execution events," in *Proc. 12th IEEE/ACS Int. Conf. Comput. Syst. Appl. (AICCSA)*, Marrakech, Morocco, Nov. 2015, pp. 1–8.
[45] S. J. van Zelst, A. Bolt, M. Hassani, B. F. van Dongen, and W. M. P. van der Aalst, "Online conformance checking: Relating event streams to process models using prefix-alignments," *Int. J. Data Sci. Analytics*. Cham, Switzerland: Springer, Oct. 2017, pp. 1–16.
[46] S. Appel, P. Kleber, S. Frischbier, T. Freudenreich, and A. Buchmann, "Modeling and execution of event stream processing in business processes," *Inf. Syst.*, vol. 46, pp. 140–156, Dec. 2014.
[47] S. Mandal, M. Hewelt, and M. Weske, "A framework for integrating real-world events and business processes in an IoT environment," in *Proc. OTM*, in Lecture Notes in Computer Science, vol. 10573. Cham, Switzerland: Springer, 2017, pp. 194–212.
[48] A. Rozinat and W. M. P. van der Aalst, "Decision mining in ProM," in *Proc. 4th Int. Conf. Bus. Process Manage. (BPM)*, Vienna, Austria, Sep. 2006, pp. 420–425.
[49] W. M. P. van der Aalst and E. Damiani, "Processes meet big data: Connecting data science with process science," *IEEE Trans. Services Comput.*, vol. 8, no. 6, pp. 810–819, Nov./Dec. 2015.
[50] F. M. Maggi, C. D. Ciccio, C. D. Francescomarino, and T. Kala, "Parallel algorithms for the automated discovery of declarative process models," *Inf. Syst.*, vol. 74, pp. 136–152, May 2017.
[51] H. Reguieg, B. Benatallah, H. R. M. Nezhad, and F. Toumani, "Event correlation analytics: Scaling process mining using mapreduce-aware event correlation discovery techniques," *IEEE Trans. Services Comput.*, vol. 8, no. 6, pp. 847–860, Nov./Dec. 2015.
[52] J. Evermann, "Scalable process discovery using map-reduce," *IEEE Trans. Services Comput.*, vol. 9, no. 3, pp. 469–481, May/Jun. 2014.
[53] S. Hernández, S. J. van Zelst, J. Ezpeleta, and W. M. P. van der Aalst, "Handling big(ger) logs: Connecting ProM 6 to apache Hadoop," in *Proc. BPM Demo Track*, 2015, pp. 80–84.
[54] H. A. Ishak Meddah and K. Belkadi, "Parallel distributed patterns mining using Hadoop mapreduce framework," *Int. J. Grid High Perform. Comput.*, vol. 9, no. 2, pp. 70–85, 2017.
[55] L. Cheng, B. F. van Dongen, and W. M. P. van der Aalst, "Efficient event correlation over distributed systems," in *Proc. 17th IEEE/ACM Int. Symp. Cluster, Cloud Grid Comput. (CCGRID)*, Madrid, Spain, May 2017, pp. 1–10.
[56] D. Loreti, F. Chesani, A. Ciampolini, and P. Mello, "A distributed approach to compliance monitoring of business process event streams," *Future Gener. Comput. Syst.*, vol. 82, pp. 104–118, May 2018.
[57] D. J. Cook and L. B. Holder, *Mining Graph Data*. Hoboken, NJ, USA: Wiley, 2006.
[58] C. Diamantini, L. Genga, D. Potena, and W. M. P. van der Aalst, "Building instance graphs for highly variable processes," *Expert Syst. Appl.*, vol. 59, pp. 101–118, Oct. 2016.
[59] B. Fazzinga, S. Flesca, F. Furfaro, E. Masciari, L. Pontieri, and C. Pulice, "How, who and when: Enhancing business process warehouses by graph based queries," in *Proc. 20th Int. Database Eng., Appl. Symp. (IDEAS)*, 2016, pp. 242–247.
[60] A. Burattin, A. Sperduti, and W. M. P. van der Aalst, "Control-flow discovery from event streams," in *Proc. IEEE Congr. Evol. Comput. (CEC)*, Jul. 2014, pp. 2420–2427.
[61] S. J. van Zelst, B. F. van Dongen, and W. M. P. van der Aalst, "Know what you stream: Generating event streams from CPN models in ProM 6," in *Proc. CEUR Workshop BPM Demo Session*, vol. 1418, 2015, pp. 85–89.

SHERIF SAKR (M'11–SM'12) received the B.Sc. and M.Sc. degrees in computer science from the Information Systems Department, Faculty of Computers and Information, Cairo University, Egypt, in 2000 and 2003, respectively, and the Ph.D. degree in computer and information science from the University of Konstanz, Germany, in 2007. In 2017, he was appointed to serve as an ACM Distinguished Speaker and as the IEEE Distinguished Speaker. He is currently a Professor in data systems with the University of Tartu, Estonia. He has published more than 100 refereed research publications in international journals and conferences. He is a Senior Member of the ACM. He is serving as the Editor-in-Chief for *Encyclopedia of Big Data Technologies* (Springer).

ZAKARIA MAAMAR received the Ph.D. degree in computer science from Université Laval, Quebec City, QC, Canada. He is currently a Professor with the College of Technological Innovation, Zayed University, Dubai, UAE. His research interests include cloud/fog computing and the Internet of Things.

AHMED AWAD received the B.Sc. and M.Sc. degrees from Cairo University in 2000 and 2003, respectively, and the Ph.D. degree from the Hasso-Plattner Institute, University of Potsdam, Germany, in 2010. He was a Lecturer with the Faculty of Computers and Information, Cairo University, Egypt. He is currently a Senior Research Fellow of the Data Systems Group, University of Tartu, Estonia. He has more than 40 publications in international journals, conferences, and workshops. His research interests are in big streams processing, and complex event processing and its applications, especially in business process management.

BOUALEM BENATALLAH received the Ph.D. degree in computer science from IMAG, Grenoble University, France. He is currently a Professor with the University of New South Wales, Sydney, NSW, Australia. He has published widely in international journals and conferences. His research interests lie in the areas of Web service protocols analysis and management, enterprise services integration, large scale and autonomous data sharing, process modeling, and service-oriented architectures for pervasive computing.

WIL M. P. VAN DER AALST is currently a Full Professor with RWTH Aachen University, where he is currently leading the Process and Data Science Group. He has published over 200 journal papers, 20 books (as author or editor), 450 refereed conference/workshop publications, and 65 book chapters. His research interests include process mining, Petri nets, business process management, workflow management, process modeling, and process analysis.