IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received August 28, 2018, accepted October 5, 2018, date of publication October 15, 2018,
date of current version November 19, 2018.

Digital Object Identifier 10.1109/ACCESS.2018.2875975

An Overview of Data-Driven Techniques for IT-Service-Management

PATRICK KUBIAK[1,2] AND STEFAN RASS[3]
[1] Alpen-Adria-Universität Klagenfurt, 9020 Klagenfurt, Austria
[2] Application Management Services Department, Volkswagen Financial Services AG, 38112 Brunswick, Germany
[3] Institute of Applied Informatics, System Security Group, Alpen-Adria-Universität Klagenfurt, 9020 Klagenfurt, Austria

Corresponding author: Stefan Rass (stefan.rass@aau.at)

ABSTRACT High availability of information technology (IT)-applications and –infrastructure components is a significant factor for the success of organizations because more and more business processes depend entirely on IT-services. The result is that the complexity of IT-environments is permanently increasing and therefore cost intensive maintenance processes become more difficult. Several frameworks for IT-service-management like IT Infrastructure Library bundle established best practices to support the task of IT-service operating. This paper provides an overview of data-driven techniques, which can be used in addition to the standards of best practice frameworks to improve reactive and proactive maintenance tasks. We present a selection of different machine learning and statistical methods from a theoretical and practical perspective, which enable detailed insights into the IT-environment status. Furthermore, we present different approaches for the analysis of IT-service tickets, which are the primary communication path between the customer and service provider in case of issues. These methods enhance the resolution-finding process by a higher automatization level or by improvement of the situational understanding in case of root cause analysis and problem determination. Moreover, prediction of the future status allows the initiation of proactive actions to possibly prevent critical situations before their occurrence; existing methods, their potentials, and shortcomings as well are in the center of this paper. We close our overview with a systematic guide for practitioners to select the proper method for their setting, based on a series of questions about the application at hand.

INDEX TERMS Event mining, IT-operations, IT-service-management, IT-ticket analysis, log file analysis, machine learning, statistics, text analysis.

I. INTRODUCTION, MOTIVATION & STRUCTURE

The meaning of information technology (IT) for organizations is increasingly rising since decades. In the case of digitalized business processes, it is necessary to deliver continuous availability of IT-services (ITS) to ensure flawless business operation. A service is defined as a delivered value from a service provider to a customer on time and in a suitable quality [1]. Moreover, service is described as ''the first, and most obvious [...] critical success factor'' [2] for service-oriented IT-management nowadays. The availability of ITS is a calculable measure but even more kind of a touchpoint between service provider and customer. Especially missing ITS availability is noticed by the customer and could result in negative consequences for the service provider [3] who is responsible for all costs and risks that are necessary to fulfill service delivery while the customer is just consuming the services without getting in touch with any providing processes or tasks [4]. Therefore, one of the major challenges for the service provider is to manage a complex and possibly heterogeneous landscape of applications and infrastructure components, both following summarized as IT-infrastructure (IT-I), on which the availability of ITS depends on. All activities from planning, providing, controlling and optimizing this IT-I are part of IT-service-management (ITSM), which is the connector between the customer and the service provider [5]. Service management is defined as ''[...] a set of specialized organizational capabilities for providing value to customers in the form of services'' and its core is the act of transforming capabilities and resources into valuable services [6]. Therefore, the core of ITSM is the transformation of applications and infrastructure components into ITS. Because of the increasing dependencies between business processes and ITS, it is almost impossible for the customers' employees to do their work

efficiently without the support of ITS. Furthermore, the missing availability of ITS could result in expensive damage for the organization [4] because of possibly lost acting ability towards their business fields. The result is that the availability of ITS is a key metric of service delivery quality [7] and an important key performance indicator for the success of ITSM efforts [8]. Availability should be as high as possible or at least as high as agreed in the service level agreement between service provider and customer. As mentioned before, ITS are directly influenced by a stable or unstable IT-I, and there are several ways to optimize the stability with a broad portfolio of processual, reactive or proactive approaches. A lot of these approaches are bundled in ITSM best practice frameworks or processes, but there are some additional and primarily analytics-based approaches that can support preventing system issues or reducing system downtimes for guaranteeing a maximum of ITS availability.

To delimit our survey from previous work, we start with an overview of related surveys that overlap from method perspective and/or from the focus on such methods in the ITSM area. Based on this, we differentiate our study from others by providing a new holistic perspective on available methods in association with common ITSM situations. The motivation of this work is to provide an overview of different processual, reactive and proactive ways that ensure high availability of ITS, and to give practitioners and researchers a guideline towards identifying a set of most suitable candidate methods depending on their applications. A common way is an orientation on ITSM best practices frameworks or processes, which include practical assistance for service-oriented IT. For professional IT-organizations, it is quite typical to use such guidelines. Therefore, we present a short section about ITSM frameworks and an overview of previous research work within the ITSM discipline. One intention of this paper is to show that implementing ITSM frameworks could bring several advantages for organizations like higher ITS quality, higher customer satisfaction, and lower costs. Related to this, there are different challenges ITSM has to deal with nowadays, which we explain in the following section. The main body of this work is the presentation of different analytics-based approaches that are actually not part of ITSM frameworks, but which can be used in addition to them. The method selection as made here is based on relevant stages of the presented ITS maintenance procedure to show that our method selection provides improvements for all of these stages. We begin with a general presentation that provides the necessary information for the understanding of the analytics-based approaches. The mediated theoretical knowledge then leads to different practical use cases for data-driven techniques that can improve the availability of ITS by avoiding issues with proactive approaches or by supporting the problem diagnosis to keep the mean time to repair (MTTR) as short as possible. We then provide a selection guide that includes some key questions to be answered for data analytics projects, whose answers then lead to recommendations from the pool of presented methods and suggestions on what aspects to pay particular attention to. This culminates in a guideline for practitioners in data analytics projects who face real-world ITSM problems.

II. RELATED SURVEYS

The task of analyzing data in the ITSM context is challenging because data science, in general, is an interdisciplinary field that draws, among others, from different research areas like statistics, software engineering, and data management. Furthermore, results published in this area are not bundled in a single venue and especially basic approaches, which include the fundamentals for the following mentioned data-driven techniques, are widely scattered. These reasons make it difficult to enable a quick deep dive into the topic, which therefore could become confusing and time-consuming, to begin with. Moreover, some important literature is highly formal and as such may require considerable mathematical background knowledge from practitioners before it can be used. Nevertheless, there are several surveys available that are good starting points for fundamental methods, which could improve the situation of handling typical ITSM challenges. These papers necessarily often deliver an isolated and limited view on specific topics and may lack practical use cases of the ITSM field nor describe how these methods could result in potential improvements within the ITS maintenance lifecycle. Our work shall fill this gap by providing an introduction to the necessary basic concepts as well as a discussion of how to choose the proper method for practical, real-world ITSM scenarios. The following review of prior surveys substantiates this need, none of which provides detailed recommendations for the selection of a particular method for a practical situation. Despite many surveys being available, a guideline to make a methodological choice with a specific focus on the ITSM domain seems unavailable so far.

Common to all these prior surveys is the absence of detailed guidance in the selection since they all focus on explaining ideas and concepts, but leave the practical choice for a method up to the reader. Given the limited scope of any such article relative to a vast lot of mechanisms, any selection guidance is necessarily incomplete, and our survey covers a representative selection over the existing methods, subsuming and extending the contents of related surveys at least. Some topics are, however, left out of scope here, such as, for example, some artificial intelligence applications like chatbots in 1st level helpdesk support or other (farther related) topics.

III. IT-SERVICE-MANAGEMENT

The role of IT as a primarily software producing department is changing into service orientation since the late 1980's. Therefore, the main focus moved from software development to management of ITS with responsibility along the whole ITS lifecycle [15]. Especially operating tasks to ensure the availability of ITS are expensive. More than 70% of IT-spending on average is paid for operating and maintenance activities to keep the IT-I and thereby the ITS going [16]. Costs are always

TABLE 1. Related surveys.

Ref.	Summary	Pros & Cons
[9]	- Provides a review of multi-label learning with an emphasis on state-of-the-art multi-label learning algorithms - Analyzes and discusses eight representative multi-label learning algorithms under the common notation	(+) A comprehensive introduction to multi-label algorithms and popular evaluation metrics including an extensive review of the eight algorithms (-) Limited to multi-label learning
[10]	- Presents a systematic analysis of 24 evaluation metrics for classification tasks - Defines a taxonomy for measure invariance with respect to all relevant label distribution changes in a classification problem	(+) Includes evaluation metrics for binary, multi-class, multi-label and hierarchical classification problems (+) Analyzes the ability of the metrics to preserve their value under changes in the confusion matrix (-) Loss functions for classifier evaluation are missing and left to the practitioner for specification
[11]	- Provides a structured and comprehensive overview of research on anomaly detection - Groups available approaches and applies them to specific application domains	(+) Describes several approaches and evaluates them regarding advantages and disadvantages (+) Gives guidance for the different methods of how to differentiate between normal and abnormal behavior (+) Combines the theoretical perspective with practical applications (-) Leaves the IT-operations domain undiscussed
[12]	- Presents a comprehensive view on online failure prediction methods (OFP) at systems' runtime - Provides a taxonomy for these methods and describes major concepts in detail	(+) Good introductory literature and suitable for beginners in this field (+) Is based on a systematic taxonomy (+) Focus on the IT-operations domain (-) Limited on predictive methods
[13]	- Introduces a taxonomy for sequential pattern mining algorithms - Provides a comparative performance analysis of many of the key techniques and discusses theoretical aspects	(+) Detailed view of relevant sequential pattern mining algorithms (+) Systematic taxonomy and detailed comparison of the algorithms (-) sequential pattern mining is only one possible event mining approach for the IT-operations domain
[14]	- Presents different data-driven techniques with a focus on IT-operations - The main focus is on system behavior analysis based on system events	(+) Focus on IT-operations domain (+) A comprehensive view of methods for system event analysis (+) Combines the theoretical perspective with practical applications (-) Events are important but not the only possible source for IT-I analysis (-) Only little details on the IT-ticket analysis
This article	- Presents a global view on processual, reactive and proactive approaches to improve the availability of ITS - Surveys a bride spectrum of fundamental approaches and adopts them into the ITSM field - Provides a practical selection guide for these methods in dependency of specific questions	(+) Presents improvements resulting from data-driven techniques for all relevant stages of the ITS maintenance procedure (+) Selection guide allows practitioners easy access to identify suitable methods for real-world ITSM scenarios (-) Formalism only used to the minimum required extent

a critical success factor (CSF) for organizations to remain competitive and a possible discrepancy for ITSM is the need to steadily lower the costs of services while improving delivery quality and speed in recent years [7]. Besides reducing costs, the following key aspects for ITSM were defined in [4].

1) Design and operation of ITS depend on defined objectives and get measured against them
2) ITS have to support the customers' business processes as best as possible
3) ITS have to be as user-friendly as possible in case of customer acceptance

4) Efficiency is as essential as effectivity and has to be optimized permanently

A. ITSM BEST PRACTISE FRAMEWORKS AND PROCESSES

To support the achievement of these objectives different ITSM frameworks and processes are available, which bundle a collection of best practices for the ITS lifecycle. A best practice is defined as a method or technique that has consistently shown better results than its alternatives [17]. Well-known ITSM frameworks include the IT Infrastructure Library (ITIL), the Control Objectives for Information and related Technology (CobiT) or the Microsoft Operations Framework (MOF). Fig. 1 shows a ranked list of popular frameworks and processes for supporting ITSM strategies. The list results of a survey with responses of 261 senior-level executives around the world [8].

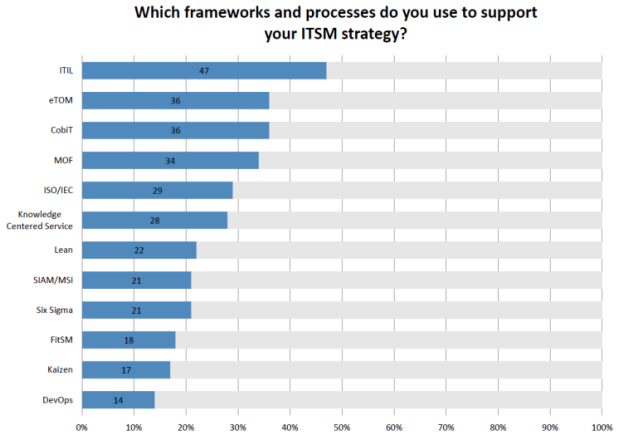

FIGURE 1. Popular ITSM frameworks and processes.

This prior work exhibits ITIL as the most popular ITSM framework, and even more, ITIL is the basis for some other frameworks that are (not) mentioned in the list above like the HP ITSM Reference Model of Hewlett-Packard, the IT Process Model of IBM or the MOF of Microsoft. These are two reasons why ITIL is often called the de facto standard for best practice process descriptions in ITSM [18]. Various ITSM research work focuses on ITSM with ITIL because it is the most common framework. Tab. 2 gives a short overview about previous ITSM research works (with ITIL), which validate the mentioned hypothesis that the use of ITSM frameworks results in different advantages and/or which describe different ITSM research fields for a deeper understanding.

As seen in this overview, there is a relatively broad spectrum ITSM research focuses on. Nevertheless, ITSM research is driven by a scattered and relatively small community though the number of publications is steadily increasing, judging from a recherché using different academic databases and search engines [25]. Of course, the implementation of such frameworks is connected to effort and costs, but the resulting long-term benefits are worth it respectively the costs can be justified by the generated benefits [19].

TABLE 2. Overview of previous ITSM research.

Reference	Summary of the research work
[15]	Describes challenges of ITIL implementation and resulting advantages of ITIL depending on the maturity level of the implementation.
[19]	Presents six case studies and compares the situation of IT-management before and after the implementation of service orientation. Shows three benefit categories, eight cost categories and finally six success factors of re-organization projects for service-oriented IT-management.
[20]	Explores if a direct correlation between the use of ITIL and customer satisfaction exists and determines if customer satisfaction is an indication of effective service provision.
[21]	Describes ITIL, CobiT, CMMI and ISO 9001 and presents a survey about possible motivation, signification, and implication of the adoption of those frameworks.
[22]	Presents four case studies of successful ITIL v2 implementations and compares some in the literature suggested CSF against these case studies. Defines additional three new CSF which were missing in the literature so far.
[23]	Presents a descriptive-comparative review of ITS design processes and analyzes seven ITSM frameworks (ITIL v2, ITIL v3, ISO/IEC 20000, CobiT 4.0, CMMI-SVC, MOF 4.0 and ITUP) by a systems approach.
[24]	Investigates the ability of pooling different IT-frameworks (ITIL, CobiT, PMbok, CMMI, ISO 27001, Val IT and eTOM) and combines their characteristics in a proposed unified maturity model.

As shown by previous research, the following benefits of the use of ITSM frameworks are beyond question – Higher service quality [19], [20], [26]; improved customer satisfaction [20], [26], [27]; higher efficiency due to process and service standardization [19], [26]; improved response and resolution rate by the service provider [27] and better use of IT-resources [27]. Research so far confirms that the use of ITSM frameworks is advantageous and supports the achievement of ITSM objectives with best practices. Besides the recommendations of those frameworks, there are some new techniques that are actually not part of them. These techniques extend the established best practices by analytics-based approaches for extra added value by automation, prevention, easier problem determination or faster troubleshooting.

B. ACTUAL CHALLENGES FOR ITSM

The challenge for ITSM is the need to steady lower the costs of services while improving delivery quality and speed in

recent years [7]. Furthermore, it is necessary to validate new technologies and innovations in case of changing customer needs, which result from digitization and increase the complexity of the IT-I. Stable IT-operations are the daily business and therefore one of the major tasks for ITSM. The conflict is due to the situation to not neglect stable IT-I operating while enabling new technologies for new customer services, increase the speed of service implementation and lower the costs of the services at the same time. From a cost perspective, it is not possible to just increase the number of employees when these additional tasks are required. To face these challenges, it is necessary to:

1) speed up ITS implementation and change cycles
2) relieve human capacities in the IT
3) ensure a maximum issue free IT-I operating and to reduce the time for root cause analysis

To achieve faster ITS implementations and change cycles paradigms like cloud computing or DevOps are available. Cloud computing allows high scalability of (high performance) IT-I and can be flexibly adjusted depending on the actual needs. This flexibility enables good steering of resources and makes the permanent allocation of budget in the hold of hardware unnecessary. Thus, fixed costs can be reduced. Another advantage is that there are less human capacities for the IT-I management necessary because tasks like patching, updating or restoring processes belong to the responsibility of the cloud provider. Resulting by new business models caused by digitization it is import to reduce the time-to-market for such products to stay competitive. DevOps is the fusion of IT-development and -operating to increase the quality of software products and to accelerate the speed of software development and deployment to satisfy the customers' business needs for new services and functions. The major characteristic of DevOps refers to the collaboration of IT-development and -operating in a partnership and agile manner to decrease the complexity, costs and time for ITS releases. According to the customers' requirement of faster service provision, self-services are another way to accelerate the access to services, which before typically belonged to the responsibility of the IT. For example, resetting passwords or changing authorizations of users could be granted to key users in the departments to relieve the IT-employees and to reduce the time for service provision in comparison to times caused by probably long service request processes. As mentioned before, stable IT-I operating is the major task for ITSM, which therefore allocates the most human capacities. There are different techniques available to reduce human efforts in this case like automatization, advanced system status analysis or system issue prediction. Automatization allows relieving human capacities by taking time-consuming and returning tasks, provided that they are sufficiently simple. For example, robotic process automation (RPA) allows automating tasks like system after-go-live checks, which refer to a fixed list of activities to ensure that the system is working correctly after changes have been implemented. The activity list consists of steps like logging in to the system, starting specific transactions, etc. A robot can learn these steps and therefore to autonomously check all required parts of the test without manual efforts. Another option is the use of chatbots as an alternative for or in addition to helpdesk call center agents. Human call center agents do often have to answer similar questions of users which also could be answered by a chatbot. RPA allows the autonomous fulfillment of returning tasks, which results in a capacity increase for the IT-employees. Chatbots, on the other hand, could reduce the number of helpdesk employees and thus reduce the costs for the enterprise. To minimize downtimes and to reduce human efforts within root cause analysis (RCA) there are several approaches available that support the analysis of the systems' state like pattern mining for system events. Pattern mining grants insights into the relationship between different events to ease the identification of potential issues and to speed up the resolution thereafter. Even better is the prevention of system issues before their occurrence because high availability of ITS increases the customer satisfaction on the one hand and reduces human efforts for RCA and resumption of the correct system state on the other hand.

C. DATA SOURCES FOR DATA-DRIVEN TECHNIQUES

Before diving into details of methods to analyze data, let us briefly look at where such data comes from. Systematically, we can distinguish *technical* from *human* data sources: technical sources would include any kind of sensory, supervision, monitoring or surveillance system that collects event or other data for subsequent analysis. Part of such analysis, but not exclusively so, is human-generated data. Mostly, this comes in the form of tickets, i.e., structured notifications that human operators create either on the grounds of their own expertise, current information or from analysis of data from technical sources. If we assume that a human operator will, in any case, retain the right of final decisions, we hereafter focus on event data and ticket data processing, since the latter is based on events, and reflects the human element in the data collection processes.

IV. IT-SERVICE MAINTENANCE PROCEDURE

The following section describes a routine maintenance procedure for ITS, which typically includes issue detection, determination, and resolution for the IT-I. Maximizing the automation of this ITS maintenance procedure is one of the ultimate objectives for ITSM [28] because it is important to ensure a fast and efficient way from problem detection to resolution to keep downtimes as short as possible. Fig. 2 illustrates an example of such a procedure, which is structured in four stages [5], [28].

At the first stage, the IT-I is monitored by software agents that collect different system metrics like CPU utilization, memory usage, storage I/O or the response times of the systems, etc. This performance data is compared to predefined thresholds like a minimum free storage capacity level, a maximum response time of an application or a time scheduled and depending batch job processing. Any violation of these

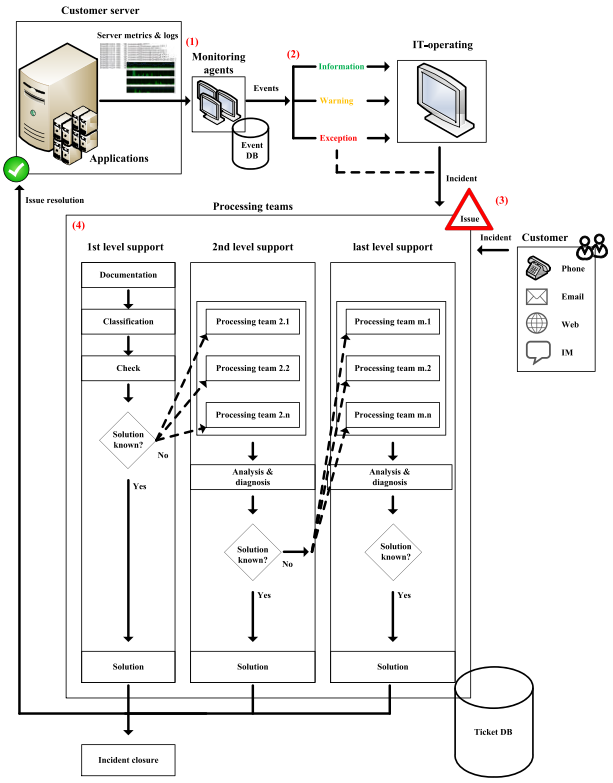

FIGURE 2. ITS maintenance procedure.

thresholds results in automatically generated alert to inform the IT-operators about potential anomalies [28].

This information process is the second stage of the procedure. The monitoring agents are not only collecting system metrics but also generate different types of events that depend on the monitored behavior, too. ITIL, for example, defines three different types of events. An information is an event that confirms a successful completion of an activity, e.g., a planned data transfer within a batch job [1], [4]. A warning is an event that informs about an untypical behavior, which is not dramatically critical at this moment but possibly requires special attention and additional monitoring activities, e.g., a rising response time or reaching a threshold [1], [4]. The third event type is called an exception. Events of this type are critical from a runtime perspective of the IT-I and imply a behavior that probably results in an issue. Special activities are required to counteract or resolve the issue, e.g., a system is not responding or violating a threshold [1], [4].

If the IT-operators get notified about such a critical event, they create an incident ticket for the service desk (SD) at the third stage. Furthermore, it is possible to automatically generate incident tickets by the monitoring agents in dependency of the event type without the need of any manual (i.e., human) interaction by the IT-operators. The particular characteristic of events is that the IT-operators in some cases can identify issues before the customer notices a problem because the ITS is possibly still available. The customer has also the possibility to notify the SD if he registers a failure or a problem within the IT-I. The SD is the single point of contact for all system users and the only entry point for incidents to ensure a structured issue resolution process [5].

The fourth stage describes this issue resolution process and starts with the notification of the SD which is the 1st level support. Important tasks of the SD are documentation and classification of the incidents because both are indispensable for the further process. It is necessary to collect all information about the issue like which component is affected, which kind of problem exists etc. This information is the basis for the classification or prioritization of the incident [1], [5]. The prioritization depends on the urgency (how critical is the incident in case of possible economic losses?) and the impact (how many users are affected?) of the incident [1]. The SD then tries to solve the issue within the first contact phase. If it is not possible to resolve the issue, the SD assigns the incident to specific processing teams of the further support levels. These teams are specialized for different problem domains and do have therefore a more profound knowledge in case of RCA, and the solution finding as the 1st level support does have. This assignment process goes on until one of the processing teams can resolve the issue. The number of support level varies, but the last level support usually is the supplier of the component, e.g., hardware producer of the server (component) or software developer [5]. Anyway, all information from issue description over all single solution steps to the resolution are persistently stored in the ticket DB, which therefore contains all history and ideally in detail described incident data. The intention of this maintenance procedure is to solve as many incidents as possible in an as short time as possible and therefore to minimize the MTTR. In some cases, a quick workaround is thus preferred over a ''clean'' and sustained resolution of the root cause. Finding this kind of solutions is a task of the problem management, which is responsible for identifying activities that ensure a long-term higher IT-I stability [1]. For the presentation of the following data-driven techniques, we focus on improvements for all four stages of the maintenance procedure, which therefore is the basis for our method selection. The monitoring data of stage one and the event data of stage two can be used to accelerate RCA due to, e.g., a detailed analysis of event relationships or advanced presentation of the actual system state. This allows a better situational understanding of the IT-I status and easier access to fault-failure-chains to identify issue triggers. This data can also be used to predict system issues and enable proactive strategies to counteract the problems before their occurrence. Thus, data-driven techniques can improve the ITS availability by reducing the MTTR or by completely preventing disruptions because potential issues can be identified before they occur. Stage three and four refer to the situation that an issue somehow was noticed and has to be solved by the processing teams to further ensure correct ITS functionality as fast as possible. Ticket data contains all necessary information of these issues and data-driven techniques allow the extraction of this textual

information to improve the resolution finding process (RFP) by, e.g., automatization or advanced problem determination in case of RCA. Some methods identify if the monitoring agents missed the detection of a critical system state, which caused an issue. The identification of such situations allows a better adjustment of the monitoring configuration to improve the ratio of system-generated tickets to user-generated tickets. Thus, the monitoring could catch a higher amount of critical situations before the customer gets notice of issues, which could lead in higher customer satisfaction. Depending on the problem description of the tickets it is possible to automate the routing to the right processing team. This reduces the time from issue detection to the start of the RFP because there is no time for wrong assignments wasted. The RFP could further be improved due to the generation of resolution recommendations based on historical resolution descriptions, which could lead in a higher first-time resolution rate as well. These and other improvements of the maintenance procedure could result due to the use of data-driven techniques, which we primarily selected to present improvements for the common ITS maintenance procedure.

V. CLASSIFICATION OF IT-SERVICE TICKETS

Tickets are an essential part of ITSM because this is the typical way how the system user expresses his requests related to incidents or configuration changes in the existing IT-I [29]. A ticket usually contains data like the creation date, the problem description, the affected system component, the resolution description, etc. This means that ticket data is a mixture of structured data like the creation date and free text like the problem description. There are different ways an incident ticket can reach the processing teams: by the customer, by the IT-operators or automatically generated by the monitoring agents, and especially incident tickets should be solved as soon as possible to minimize the business impact. Therefore, it is imperative that the tickets reach the right processing team in dependency of the problem domain that possibly causes the issue without wasting time because of wrong assignments. Classification of tickets is an interesting field to improve and accelerate the resolution process in case of automated routing to the right processing teams, resolution recommendations based on historical resolution descriptions of similar problem domains and RCA by identifying the whole taxonomy of the described issue in the ticket. Nevertheless, there are some challenging tasks, which are the result of the tickets' characteristics [30], [31].

1) The number of tickets is, depending on the scale and the complexity of the IT-I, vast, so that manual labeling of tickets is simply infeasible
2) Ticket data is a mixture of human and machine-generated text
3) The system-generated text could contain a specific vocabulary like error codes, which differ in dependency of the creating monitoring agent
4) The user-generated text could contain spelling and grammar errors
5) Assigning the ticket to the right processing team is often nontrivial because in-depth expert knowledge and/or insights in the IT-I are required

The basis of ticket analysis is the ability to classify and extract information from text, which has been a research field that evaluated different statistical approaches and machine learning (ML) algorithms in recent years [32]. One of the cited references presents an empirical comparison of a Bayesian classifier and a decision tree algorithm for text categorization on two data sets from a performance perspective. The finding is that both methods achieve reasonable performance and that controlled tradeoffs between false positives and false negatives are allowed [33]. Another comparison evaluates five different learning algorithms – Find Similar, decision tree, Naive Bayes, Bayes Nets and Support Vector Machines (SVM) – regarding learning speed, real-time classification speed and classification accuracy. The results show that a simple linear SVM is an effective and efficient algorithm and the best choice for the used data sets in case of speed and accuracy, and at least 35 times faster than the decision tree, which was the most accurate classifier [34]. The suitability of SVM for text categorization was also confirmed in another reference of the same year [35]. Further work on automatic text classification based on SVM is found in [36]–[38] and relating to Naive Bayes in [39] and [40]. An important field for text mining is topic modeling in which each document can be examined and clustered based on the most prominent topics using an unsupervised approach [30]. Several works in this area are based on Latent Dirichlet Allocation (LDA) [41]–[44] or key phrase extraction for short texts [45].

A. HIERARCHICAL MULTI-LABEL CLASSIFICATION

A lot of research work focuses on flat classification problems, but many important real-world problems rely on hierarchical classification (HC). Flat classification refers to binary or multi-class classification while HC organizes the predicted classes typically in a tree or a Directed Acyclic Graph (DAG). The main difference between a tree and a DAG is that in the DAG a node can have more than one parent node [46]. Flat classification is the simplest way to deal with HC problems because it typically only predicts the classes of the leaf nodes and therefore ignores the class hierarchy. Finally, it provides an indirect solution for HC problems because the classification of a leaf node implies that all ancestor classes are assigned to that instance [46]. This simple approach has the disadvantage of having to build a classifier that analyzes a large number of classes (all leaf classes) without exploring information about parent-child class relationships [46]. Using ticket classification to improve the resolution process of issues makes it necessary to get information about the parent-child relationship, e.g., in the case of RCA. Therefore, HC is necessary to understand the whole taxonomy of the problem domain described in the ticket. Different research focuses on hierarchical classification (partially including comparison to flat classification) [46]–[54]. Furthermore, ticket

classification requires hierarchical multi-label classification (HMC) where the instance can belong to more than one path (i.e., to more than one class) or to a path not ending in a leaf in the hierarchy. Therefore, the instance results in a classification *vector* instead of a single class [55]. Research on such classifications has received much attention, although a lot of work is not specific on ticket classification problems [56]–[64]. The following example substantiates the above-mentioned categorization of ticket classification problems as an HMC problem and uses the ticket description of Tab. 3.

TABLE 3. Ticket description.

Reported by	Creator: SysUser1234 Department: Sales Email: SysUser1234@email.com Phone: +49 531 / 1234
Date	2017/11/11 08:30:00
Description	Application RepMan failures on opening sales report with data selection older than 2006/01/01. No access to data of NLS.cluster01.com/xx with error code "Err_NoConn_ID_123456" possible.

The issue described in the ticket is that a system user tries to open a sales report with data selection older than 2006/01/01. The data management in this example is organized in the way that data older than 10 years is written into a near line storage (NLS) to keep the volume of the operational database as small as possible because this relatively old data is not used frequently. Anyway, the task of the NLS is to allow reporting on older data if it is necessary without having to load it from a backup, e.g., a tape. To illustrate the problem domain of the ticket as an HMC problem, Fig. 3 shows a tree-based taxonomy of a possible root cause that should be identified by the classifier [55].

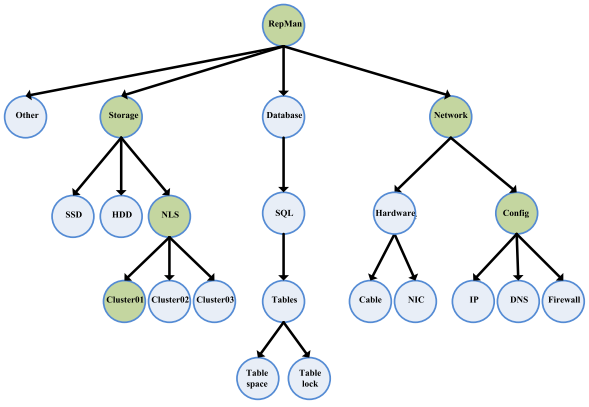

FIGURE 3. Expected tree-based taxonomy of ticket description after classification.

There are two important facts in the short text of the ticket description: cluster one of the NLS is affected, and there is an error code, which refers to a misconfiguration of the network settings. In this case, it is necessary that the classifier identifies the left part of the tree (storage) entirely because the text "NLS.cluster01.com/xx" clearly refers to cluster node number one of the NLS. Moreover, the error code contains information that the problem results from a failing connection to the NLS and the specific number of the error code refer to a network configuration problem. But the error code is not that specific that the exact type of the misconfiguration (IP, DNS or firewall) can be identified. This is why the analysis of the right part of the tree ends at the next-to-last level of the tree. This simple example shows three essential and aforementioned HMC characteristics why ticket classification is an HMC classification problem.

First, the classification path of the right tree ends one hierarchy level before reaching the leaf node because the ticket description delivers not enough information to identify the possible root cause in whole.

Second, the problem is possibly due to more than one direction because the ticket description delivers information about a network and/or a storage problem, which makes multiple labeling within the tree in case of RCA activities necessary.

Third, in case of automated ticket routing, it is necessary that not only information about the last level is identified, but also information about the level above is known. Think of a processing team structure, in which NLS problems belong to another processing team than problems with typical server storage components like HDD or SSD, and so NLS is just a subgroup of the storage processing teams. The ability of this kind of granular differentiation between problem domains over the whole hierarchy from the application to the possibly causing component and therefore automated routing is not possible without exploring information about all parent-child relationships within the tree structure. These are important reasons, which show that HMC works for ticket classification problems.

B. QUALITY AND PERFORMANCE MEASURES

An important task within HMC is the evaluation, or measurement, of the classification quality in case of misclassification errors. The evaluation of a multi-label classifier is more challenging than evaluating a single label classifier because the prediction for an instance is a vector of labels instead of a single class. So the classification vector can for the instance be fully correct, partially correct (with different levels of correctness) or fully incorrect [65]. There are some comprehensive literature surveys, which describe different evaluating metrics for multi-label classification [9], [10], [65]–[67]. These metrics can be generally categorized into two groups, example-based, and label-based metrics, as shown in Tab. 4 [9].

The example-based metrics evaluate the average difference between the predicted labels and the actual labels for each test example and then average over all examples in the test set. Different to that, label-based metrics first evaluate each label separately and then return the macro/micro average over all labels [9], [65]. The macro average is computed by the score of each local label first and then averaged over all labels while

TABLE 4. Multi-label evaluation metrics.

Multi-label evaluation metrics			
Example-based metrics		**Label-based metrics**	
Classification	**Ranking**	**Classification**	**Ranking**
Exact Match Ratio	One-error	Macro/Micro Average of Accuracy, Precision, Recall, F1-Score	Macro/ Micro AUC
Accuracy, Precision, Recall, F1-Score	Coverage		
Hamming Loss	Ranking Loss		
	Average Precision		

the micro-average is calculated globally over all instances and class labels [65], [68]. To evaluate the quality of a classifier, it is common to use a confusion matrix with four possible categories, which are partially the basis for the calculation of the metrics [69]. Tab. 5 shows the confusion matrix with its possible classification categories [70].

TABLE 5. Confusion matrix – General structure.

	Actual positive	**Actual negative**
Predicted positive	True Positive (TP)	False Positive (FP)
Predicted negative	False Negative (FN)	True Negative (TN)

TP is a classification correctly predicted as positive (a hit). FP is a classification incorrectly predicted as positive (a false mark). A TN corresponds to negatives correctly predicted as negatives (a correct rejection) and finally, a FN refers to positive classifications incorrectly labeled as negative (a miss) [69]. The metrics in Tab. 4 will be described later. For a deeper understanding and mathematical formulae explanations, we refer to the rich literature [9], [65], [67], [69]–[73]. To guarantee a fair and honest evaluation of the classifier, the quality should rather be tested on a broad range of metrics than only on one metric that is being optimized [9].

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

$$\text{F1-Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

Commonly, the performance of the learning algorithm is considered higher when accuracy, precision, recall, and F1-score are higher [65].

1) EXACT MATCH RATIO

Extends accuracy used in single label classification for multi-label prediction. It is a trivial way to evaluate multi-label classification because the case of partial correctness is ignored and considered as incorrect.

2) HAMMING LOSS (HL)

Evaluates how many times, on average, an example-label pair is misclassified. It takes into account the prediction error (an incorrect label is predicted) and the missing error (a relevant label is not predicted) normalized over the total number of classes and the total number of examples.

3) ONE-ERROR (OE)

Evaluates how many times the top-ranked label is not in the set of relevant labels of the instance.

4) COVERAGE

Evaluates how far on average a learning algorithm needs to go down the ranked list of labels to cover all relevant labels of an instance.

5) RANKING LOSS (RL)

Expresses the total number of times that irrelevant labels are ranked higher than relevant labels.

6) AVERAGE PRECISION (AP)

For each relevant label, average precision computes the proportion of relevant labels that are ranked before it and averages over all relevant labels.

The smaller the values of HL, OE, RL, and coverage are, the better the performance of the learning algorithm is. For AP the performance is better if the value is bigger, with optimal performance at a value of 1 [9], [67].

The above mentioned example-based metrics accuracy, precision, recall, F1-Score and the binary classifier area under the ROC curve (AUC) can also be used as label-based metrics within the macro/micro average approach. The idea is to compute a single-label metric based on the number of TP, TN, FP, and FN. If several labels per pattern are each part of a contingency table, it is necessary to compute an average value (macro or micro average) [67]. The larger the label-based metrics are, the better the performance of the learning algorithm is, with optimal performance at a value of 1 [9]. Beside these evaluation metrics for classification, there are some additional tools that grant a further analysis of the classifiers' quality or respectively performance, the so-called loss functions (LF). A LF with x, y, and z quantifies the quality of a classifier if x is used to make a prediction on y when the correct output is z [74]. Therefore, a LF measures the costs of misclassification, which corresponds to the two categories FP and FN of the confusion matrix shown

in Tab. 4 [75]. The Zero-One loss (Z-Loss) is typically used in pattern classification but in hierarchical settings it would on„ly count one mistake each time for a given instance [52]. The problem is that a HMC is not only fully correct or fully incorrect. The case of partial correctness of the classification would be ignored by using Z-Loss because the whole hierarchy would be marked as incorrect if only a single label is misclassified [76]. To take into account of partial correctness, the hierarchical loss (H-Loss) was proposed. The H-Loss differentiates between a completely incorrect classification and misclassification that depends on wrong labeling of a partial path within the hierarchy. The idea is that if a misclassification is made at node i of the taxonomy, the further misclassifications made in the subtree at i are irrelevant [52]. Later, the H-Loss was extended to differently weight the costs for FP and FN errors [77]. Nevertheless, the H-Loss has the limitation that it can only be used on tree-structured hierarchies and not on DAG-structured hierarchies because the nodes in a DAG may have more than one parent node and in this case it is not clear how to define the first misclassification [78]. To avoid the deficiencies of the H-Loss, the HMC-Loss was proposed and can be used on both, tree- and DAG-structured, hierarchies. Moreover, the HMC-Loss is more informative because it can trade off FP and FN errors individually and weigh the misclassification differently according to the hierarchy level of the misclassification [78]. Regarding to RCA there are some situations, in which the previous mentioned LF could be deceptively. Fig. 4 illustrates an example, in which the LF values in the case of RCA could be misleading [55].

Fig. 4 shows three scenarios, in which different classification errors occurred:

1) A: two FN within two subtrees
2) B: two FN within one subtree
3) C: one FP within one subtree

The Z-Loss would evaluate all scenarios with an error of 1 because the whole hierarchy is incorrect if any node of the tree is labeled incorrectly. The classic H-Loss would evaluate scenario A with an error of 2 while scenarios B and C would be evaluated with an error of 1. The HMC-Loss would evaluate scenario A and B with an error of 2FN while scenario C would be evaluated with an error of 1FP. Therefore, scenario C seems to have the best classifier with an "overall error" of [1, 1, 1] for the three mentioned loss functions. In the case of RCA within ticket analysis, this is dangerous. The two FN at A and B, of course, are wrong, but FNs just expand the pool of possible resolutions for the described problem. The FP at C is more critical for the resolution processing because it recommends a resolution way based on a wrong component (the database). For RCA the minor mistakes in multiple branches are not worse than a major mistake in a single branch, which leads the RCA in a completely wrong direction [55]. Therefore, it is necessary to consider the contextual information for each misclassified label, which is part of the contextual hierarchical loss (CH-Loss) [55]. To take into account the real scenario in practice, the CH-Loss was extended by integrating the knowledge from domain experts to improve the determination of ticket classification problems [79]. For the sake of completeness, there is a hierarchical extension of HL and RL which includes Bayesian decision theory [80].

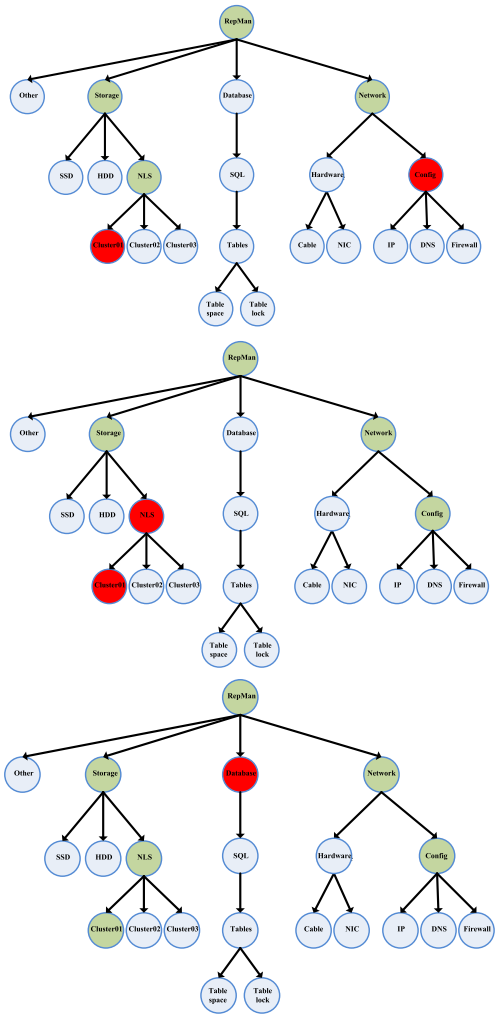

FIGURE 4. Misclassification scenarios A, B and C.

C. USE CASES FOR TICKET CLASSIFICATION

The use of classification on ticket data is an interdisciplinary research field drawing from various methods and delivering different improvements. The case of automated ticket routing to processing teams was already compared by Vector Space model, Bayesian model, SVM, classification and regression tree and k-Nearest-Neighbor (kNN) in the early 2000's [32]. The case of reducing human efforts for labeling and automated routing for tickets accelerates the resolution process because there is no time lost by wrong assignments, and so the processing teams can work on the solution earlier respectively the time between the ticket assignment and beginning of the resolution process is shorter. There are different approaches mentioned in the literature [81]–[86]. Using automated routing based on ticket classification is about to improve the

efficiency of ticket routing by 35% respectively reducing the service time by >35% [81], [85].

The pre-step for the ability of automated routing is the extraction of the tickets' contextual content by correctly analyzing the information of the ticket text. Tab. 6 gives an overview of different approaches for extracting information from tickets with further use cases based on the classified ticket data.

A further important use case of ticket classification is the ability to generate resolution recommendations based on historical ticket data. This use case offers a huge advantage for the processing teams within RCA and therefore the RFP. The possibility of generating recommendations for incoming tickets implies a chance to dramatically shorten the processing time from issue detection to resolution. One approach focuses on system-generated tickets, which partially depend on repeating events the monitoring agents generate and which therefore result in similar tickets with a similar resolution. The analysis of the historic ticket data is based on a kNN-algorithm in combination with a LDA-based topic-level feature extraction model [100]. In contrast to that, another work [101] focuses on change request tickets, which always are user-generated tickets to classify the tickets into a hierarchy of categories, tasks, and actions. This reference evaluates a set of classification techniques and ML algorithms for parameter extraction like CRF and long short-term memory neural networks (LSTM NN) on four real-world datasets. Further recent work focuses on the use of deep NN for comparison with kNN-classifier [102] or the ranking of quantified resolution quality of historical tickets when recommending resolutions for incoming tickets [103].

TABLE 6. Overview of use cases for ticket classification.

Reference	Summary of the research work
[87]	Presents a rule-based crowdsourcing approach for classification of problem tickets where users can define classification rules and a social networking platform is used to socialize and execute these rules by large practitioner communities.
[88]	Presents a platform for problem determination with multi-dimensional knowledge integration based on automated ticket classification, the integration of configuration data for the component association and the integration of monitoring agents' data for the collection of relevant system vitals.
[89]	Presents a mechanism which models each ticket as a vector of keywords and then transforms the output in a labeled hierarchy according to the tickets' content.
[90]	Presents a planning support tool for change requests which takes into account past failure reasons or best implementation practices. Compares an information retrieval, a supervised ML and a semi-supervised ML approach for the task of change request classification.
[91]	Presents a hybrid search engine that finds relevant co-occurring and re-occurring incident tickets in structured and unstructured formats to provide further insights in case of RCA and RFP.
[92]	Presents a Random Forest model based on ticket data which identifies and ranks servers with problematic behavior as candidates for modernization actions and then evaluates the impact of different modernization actions for choosing the effective ones.
[29]	Presents a ML approach based on Conditional Random Fields (CRF) which automatically identifies the server name in descriptions of the tickets and illustrates the possibility of generating new business insights with this linkage.
[93]	Presents an approach to identify missed monitoring alerts with the analysis of user-generated tickets to optimize the configuration for system-generated tickets in case of reducing FN-scenarios.
[94]	Presents an analysis based on ticket data to inspect the impact of hardware configuration and operating system type on server availability.

VI. ONLINE FAILURE PREDICTION METHODS

The IT-I environments are growing and getting more complex. Furthermore, they are changing dynamically due to changing execution environments, frequent updates/upgrades and the change (the addition or removal) of system components. Classical reliability theory and conventional methods may disregard the actual system state and are not always capable of reflecting these dynamics in the case of system runtime and failure processes. Such methods are typically useful, e.g., in designing of architectural properties for continuously higher stability and long-term prediction of the IT-I reliability. OFP is an approach for a short-term assessment that allows analyzing the events in the near future like seconds, minutes or hours at the latest [12].

A. DEFINITIONS

Short-term predictions are especially helpful to prevent potential issues or to limit the damage caused by system failures. OFP methods allow "to identify during runtime whether a failure will occur in the near future based on an assessment of the monitored current system state" [12].

TABLE 6. *(Continued.)* **Overview of use cases for ticket classification.**

[95]	Presents a large-scale study based on more than 13.500 tickets to analyze the factors affecting the labor effort of ticket resolution in datacenters that operate under the private cloud paradigm.
[30]	Presents a two-stage technique that combines hierarchical clustering using a combination of graph clustering and topic modeling at the first stage following another round of hierarchical clustering or an active learning approach for the second stage to reduce the manual labeling effort.
[31]	Presents two algorithms for extracting information from system- and user-generated tickets. Recommends clustering on descriptions for system-generated tickets and a keyword based approach for user-generated descriptions.
[96]	Presents an ensemble SVM based approach to identify user-generated tickets for optimizing the quality of system-generated tickets in case of reducing FN-scenarios.
[97]	Presents a cognitive agent who retrieves a summary of the ticket and then tags the relevant parts as a part-of-the-problem or part-of-the-solution based on hardware tickets.
[98]	Presents a hierarchical online classification framework for automatically determining the root cause of problems using an online Perceptron algorithm which is compared to a SVM classifier. Furthermore, a CRF based approach for automatically structuring of tickets is presented.
[99]	Presents a ticket classification framework using ticket partition and a signature based algorithm which is proposed to identify the problem type of an incoming ticket.

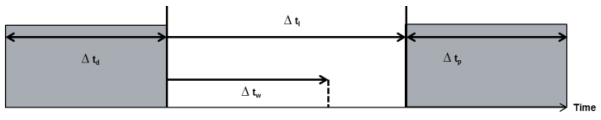

FIGURE 5. **The general timeline of OFP.**

The timeline of OFP is shown in Fig. 5 [12], with explanations following below.

The present time is denoted by t while the potential occurrence of a failure is to be predicted with a lead-time Δt_l, which is based on the current system state and assessed by system monitoring within a data window length Δt_d. The prediction period Δt_p is the time interval, in which the prediction is "sufficiently accurate". An increasing Δt_p increases the probability that a failure is predicted correctly but on the other hand, if Δt_p is too large, the prediction loses accuracy since it is not clear when exactly the failure will occur. Furthermore, it is necessary that Δt_l is larger than the time the system needs to react in order to avoid a failure or to prepare for it. This minimal warning time is denoted by Δt_w. If Δt_l is shorter than Δt_w, there would not be enough time to initiate any preparatory of countermeasures. The failure would occur before the IT operators are able to execute any proactive activities to prevent the failure occurrence [12].

To ensure the correct understanding of the further work, it is necessary to distinguish the terms "fault," "error," "symptom" and "failure," which are shown in Fig. 6 [12].

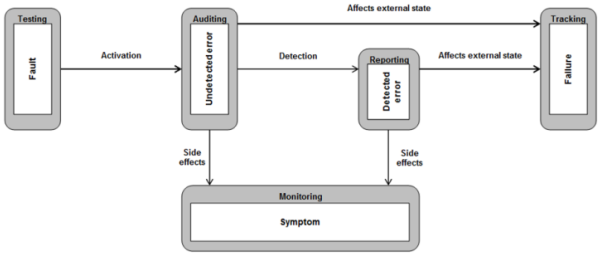

FIGURE 6. **Relation between terms: fault, error, symptom, and failure.**

A *failure* is a situation when the delivered service deviates from correct service and refers to misbehavior that can be observed by the user, which either can be a human or another IT-I system. A failure is present if and only if the ITS generates an incorrect output. Therefore, misbehavior within the IT-I is not defined as a failure as long as the ITS output is correct. A situation when things go wrong and the system's state deviates from the correct state is defined as an *error*. There are two types of errors, which can be distinguished: detected and undetected errors. An error remains undetected until an error detector identifies the incorrect state as result of the unregistered error. The root cause of an error is called a *fault,* which after activation results in an incorrect system state. Errors are called "manifestation" of faults as well [12]. In the literature, the best-known fault types are transient, intermittent and permanent faults [104]. A *symptom* is an out-of-norm behavior as a side effect, which can be identified by the monitoring of the IT-I. There can be an m-to-n relationship between failures, errors, faults, and symptoms. For example, several faults may activate one single error, or one fault may activate several errors. Furthermore, some errors result in a failure, and some do not. Some errors only result in a failure if special conditions exist and so some errors may have no influence to the ITS although the IT-I's state deviates from the correct state. Not every error does necessarily show symptoms and so on. *Testing* is a way to identify faults by finding flaws in the IT-I, e.g., a speed test for the determination of

the available network bandwidth. *Auditing* can be used to identify undetected errors by checking whether the entity is in an incorrect state, e.g., checksumming the data structure of the storage. *Monitoring* registers symptoms and therefore out-of-norm behavior by analyzing different parameters like CPU utilization or free memory space level and is able to make undetected errors visible. If an undetected error is identified, it can be made visible by *reporting* via logging mechanisms. Finally, failures can be identified by *tracking* mechanisms like analyzing the ITS response time or by sending test requests to the system [12]. Consider software with a missing free memory statement in the source code, which is the fault in this example. The fault remains dormant until the part of the software is executed. After the execution of this statement or the activation of the fault the system's state turns into an incorrect state because memory is consumed and never freed. This is the error in the example. If the allocated memory is negligibly small, it will neither be detected nor will have a negative influence on the delivery of the ITS. A failure then occurs if the statement is executed so many times and the free memory space level turns so low that the software is not able to execute further operations and therefore is not able to generate any (correct) output. This critical memory space level is then recognized as a symptom of the ''free memory'' parameter and turns from an undetected error to a detected error [12].

B. A TAXONOMY OF OFP METHODS

Fig. 7 [12] shows a taxonomy of OFP methods, which differentiates between four major branches that depend on the input data type, namely data from failure tracking, symptom monitoring, detected error reporting and undetected error auditing. Each branch is then divided into principle approaches, which then are further divided into categories of possible methods.

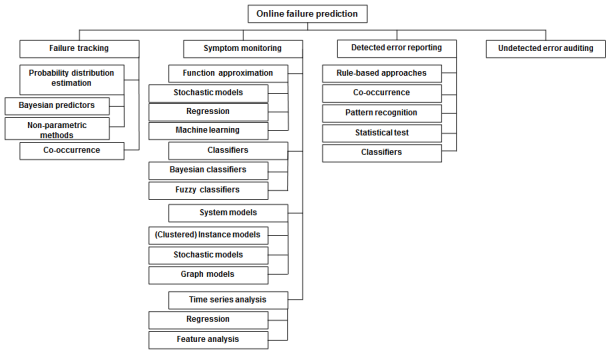

FIGURE 7. A taxonomy of OFP methods.

The intention of failure tracking is to identify upcoming failures from the occurrence of previous failures and includes the time of the occurrence as well as the failure type. Some methods try to determine the probability distribution of the time to next failure from the previous failure occurrence. It has been observed that failures sometimes occur with a temporal and/or spatial dependency because of the sharing of the same IT-I resources [105]. Therefore, methods pursuing a co-occurrence approach try to identify events that lead in a failure-prone situation and to predict failures by analyzing the relationship between event types and failures. Based on data of the IBM BlueGene/L supercomputer, three failure prediction methods were developed, which analyze the correlation between fatal and non-fatal failure error events. These methods correctly predict around 80% of the memory/network failures and around 50% of the application I/O failures [106]. The monitoring of system metrics is a common task in IT-operations. It is about to continuously measure system runtime metrics like free storage capacity or CPU utilization and to inform the IT-operators if, e.g., a predefined threshold is violated. The advantage of monitoring is that some errors affect the system before they are detected, and this influence can be discovered as a side effect (or symptoms) by the monitoring. For example, a scarce memory level may first slow down the system before it results in a failure. Monitoring can discover this slowing down the process by, e.g., measuring the response time of the system and so to identify this misbehavior before there is no memory left, which then results in a failure. Function approximation methods try to mimic a target value, which is supposed to be the output of an unknown function of measured system metrics as input data. The target function type is either the probability of failure occurrence or some system metrics such as the amount of free memory. In the first target function type, the target value is a Boolean variable, which is only available in the training data set but not during runtime. This case is shown in Fig. 8 [12]. The second target function type is used to analyze and predict the resource usage respectively the time of resource exhaustion.

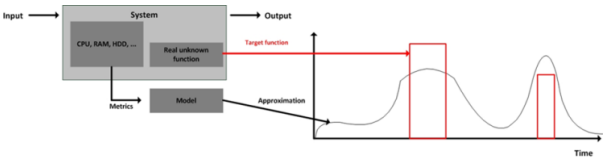

FIGURE 8. Function approximation.

In contrast to that, classifier methods evaluate the current system metrics directly instead of approximating a target function. These methods achieve prediction by classifying if the current situation is failure-prone or not. The decision boundary is either derived in a supervised manner by labeled training data or in an unsupervised manner when the classifier identifies the inference between a failure-prone and non-failure-prone situation autonomously. This case is shown in Fig. 9 [12].

Methods of the system models approach do not require training data for failure-prone and non-failure prone cases because they only learn failure free behavior. The model then calculates expected values for the monitored metrics and compares them with the true actual values. If they differ significantly, the system shows misbehavior and an upcoming

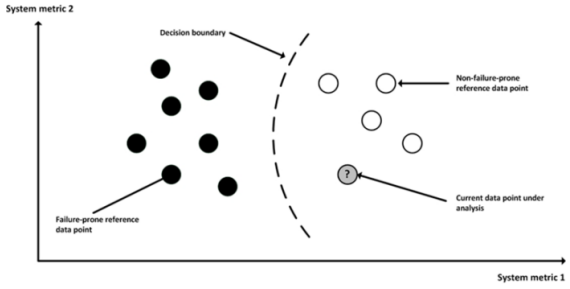

FIGURE 9. Classifier.

failure is predicted. IT-I performance metrics like CPU utilization, memory usage or storage consumption refer to continuous data items, which are described as time series [14]. Time series analysis methods, therefore, predict the future value of the metrics based on past recordings (history) and compute residual values, which judge whether the future situation could be failure-prone or not. Furthermore, the prediction of time series allows a forecast for the resource exhaustion time. Detected errors are usually reported in log files. In contrast to symptom monitoring the input data, therefore, consists of discrete and structured events which include textual information like event IDs or log messages. The task of OFP based on log file events is shown in Fig. 10 [12]. Here, the events $\{A, B, C\}$ that have occurred shortly before the present time t_0 are analyzed and used to predict whether there will be a failure in the future or not.

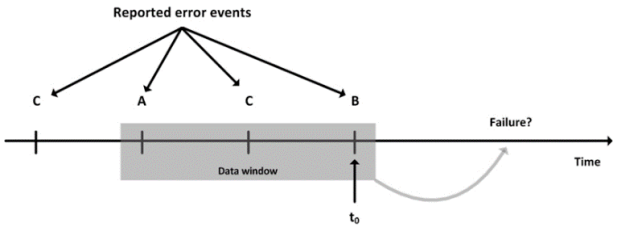

FIGURE 10. OFP based on log files.

Methods of rule-based systems try to algorithmically learn rules from a set of training data to predict a failure if a set of conditions is met. The rules should be general enough to identify as many failures as possible and specific enough to minimize the amount of false failure warning. A rule could look like this:

If CPU utilization $\geq 90\%$ AND free memory level $\leq 10\%$ THEN failure warning

Methods of the co-occurrence approach of this branch differ from the previously mentioned co-occurrence approach in the way that the analysis is based on detected errors and not on previous failures. Pattern recognition is used to identify hidden structures or systematics (''patterns'') within event sequences. The objective is to find patterns that are known to lead to failures. Statistical tests methods analyze the reported errors and, e.g., compare it to the ''statistically normal'' distribution by using statistical tests. Classifiers assign a class label to a given event sequence, which is a vector of error events because a single detected error is often not sufficient to infer if a failure is upcoming or not. Undetected error auditing is the active search for incorrect states within the IT-I. It is a way to predict failures as early as possible and therefore to identify misbehavior before it is registered by monitoring or reported in a log file. The difference to detected error reporting is that auditing consequently searches for incorrect states within the IT-I without the restriction whether the data is used at the moment or not. Thus, this branch analyzes the whole IT-I state continuously and searches for flaws that could result in failures. For example, the structure of an OS file system could be checked for consistency [12]. This taxonomy allows a good first orientation for prediction methods to early identify critical system behavior that could result in a failure. Because pattern recognition for log file events has been intensively studied in research, our further investigation focuses on this topic to exhibit that a broad spectrum of different approaches is available. Our survey presents a general introduction to this topic and a selection of algorithms depending on different IT-I monitoring situations. We believe that log file events are very suitable for system state analysis and prediction because they are an informative and objective source. If the system generates and protocols an error event within a log file, this a verified fact that something in the system went wrong. Events, e.g., generated by human-defined thresholds, on the other hand, tend to be subjective because the decision, which measures are critical or not often depends on specific rules of the IT-departments and may not be sufficient from a system perspective. Furthermore, pattern mining allows the analysis of huge log file amounts, which exceed human capacities and identify previously unknown relationships within event sequences that grant new and valuable insights to the system state.

VII. IT-INFRASTRUCTURE EVENT ANALYSIS

The IT-I is permanently monitored by agents that control their status. Information like running states of the components, errors or parameter values for CPU, RAM, HDD, etc. are monitored and saved in textual system log files. For human experts, it is much easier to explore discrete or structured events than raw textual log messages because the possible visualization forms of events help to improve the situational understanding of the IT-I behavior [14]. Therefore, it is common to generate events from textual system log files. A lot of research has focused on event mining and proposed different techniques and algorithms for discovering IT-I behavior based on log files and/or events respectively analyzing the relationships of events and system components [107]–[119]. The mentioned research also includes cases of problem/anomaly detection and problem determination within IT-I operating processing. Detailed information about the IT-I status is mainly stored in the log files, so it is recommended to convert these log files into discrete or

structured events. Fig. 11 shows a textual log file and event-based view on activities within IT-I monitoring, in which the events are generated from the log file information.

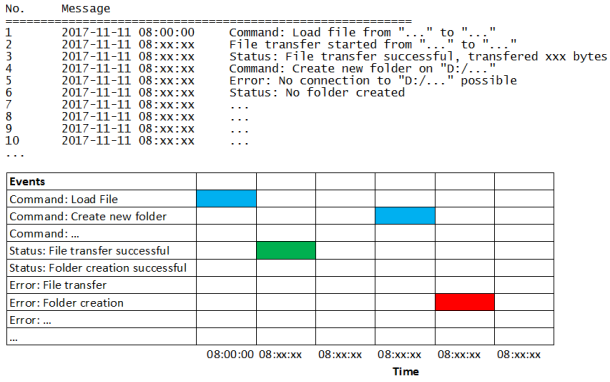

FIGURE 11. Log file and event-based view on IT-I monitoring activities.

There are different ways for the converting task, but generally, three types can be distinguished: solutions based on log parser, classification, and clustering [14]. A log parser is straightforward to develop for different formats, and there are various libraries and (commercial) tools available for specific log formats, e.g., Microsoft. The implementation of log parser based conversion in the IT-I gets problematic when the complexity and the heterogeneity of the landscape increases because detailed semantic information can only be extracted when familiarity with the format is ensured, which possibly results in high human configuration efforts. Often it is not necessary to get detailed information in case of monitoring because a high-level view on the activities is acceptable for the first step. Therefore, the conversion of log messages into events is the same problem as identifying the type of the log message. The result is that the log converting problem turns into a text classification problem. Several text classification approaches were mentioned in the section above, and some of them can also be used for this converting problem, e.g., SVM. The main problem of classification approaches is that these methods require human activities for the label training, which means that a set of labeled log messages has to be prepared by domain experts. Analogous to the log parser approach those human efforts may be time intensive and costly in dependency of the complexity and heterogeneity of the IT-I landscape. Nevertheless, the classification approach can be easier adapted to various systems than the log parser, and both approaches reach (very) accurate results in case of log message type identification. A method that does not need a lot of human effort and can be used in various contexts is clustering. Here, the inference of log messages and events is autonomously explored by the algorithm at the cost of accuracy, which sometimes is acceptable for event mining algorithms respectively as starting-point for further human analysis [14]. Because the log file to event converting process has been widely studied, there are several approaches available, e.g., [120]–[126], which could lead to a confusing situation for potential users when they are unaware of the individual pros and cons. The result is that methods become unnecessarily re-implemented or re-designed and thus create redundant work and mechanisms. A comparison and evaluation of four approaches are presented in the following study [127].

A. EVENT PATTERN MINING

The monitoring of the IT-I metrics is an important task to control the IT-I landscape and to ensure that the ITS are working as expected. These metrics refer to temporal data, which is a collection of data items associated with timestamps and which describes the status of the IT-I over time. Temporal data basically consists of time series data and event data. The first one describes the data with values of the data items being continuous while event data describes the data with discrete data item values [14]. Examples for data items represented as time series in IT-I monitoring are CPU utilization, memory usage or storage consumption while examples for data items represented as events are the reboot of a system, the start of a batch job or the request/response processing of an application. Event pattern mining is a discipline that is used to identify hidden patterns, or generally spoken, relevant relationships and/or similarities between events, which are able to help the understanding of the actual (and future) behavior of the IT-I to possibly derive proactive activities for ensuring a further stable IT-I operating. Among the vast lot of pattern mining methods, we shall consider a (necessarily non-exhaustive) selection of prominent methods. Sequential pattern mining (SPM) consists of discovering interesting subsequences in a set of (event) sequences as patterns where each sequence consists of a list of elements and each element consists of a set of items [128]. SPM was originally designed to be used for event sequences but can also be used for time series after converting them into an event sequence. Fig. 12 shows an example for converting a time series into a sequence where a, b, c and d are defined as events with an increase of 100MB, a decrease of 100MB, an increase of 150MB and an increase of 200MB in the case of storage capacity processing [129]. There are different

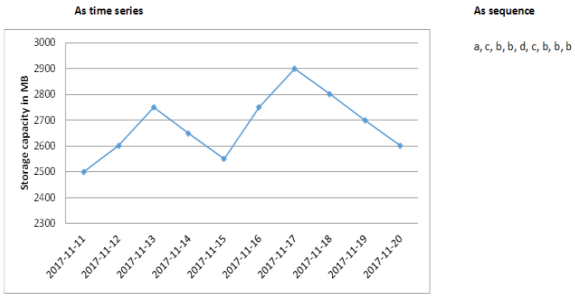

FIGURE 12. Time series into sequence conversion.

techniques for converting time series into event sequences available [130], [131].

The following example explains the functionality of SPM where Tab. 7 is a sequence database (SDB) and Tab. 8 shows the result after analyzing the sequences in the SDB [129].

TABLE 7. Sequence database.

ID	Sequence
1	$\langle\{a,b\},\{c\},\{f,g\},\{g\},\{e\}\rangle$
2	$\langle\{a,d\},\{c\},\{b\},\{a,b,e,f\}\rangle$
3	$\langle\{a\},\{b\},\{f,g\},\{e\}\rangle$
4	$\langle\{b\},\{f,g\}\rangle$

TABLE 8. SPM result with minsup 75%.

Pattern	sup(S_a)
$\langle\{a\}\rangle$	3
$\langle\{a\},\{e\}\rangle$	3
$\langle\{a\},\{f\}\rangle$	3
$\langle\{b\}\rangle$	4
$\langle\{b\},\{e\}\rangle$	3
$\langle\{b\},\{f\}\rangle$	4
$\langle\{b\},\{f\},\{g\}\rangle$	3
$\langle\{b\},\{g\}\rangle$	3
$\langle\{e\}\rangle$	3
$\langle\{f\}\rangle$	4
$\langle\{g\}\rangle$	3

Considered that the set of symbols $I = \{a, b, c, d, e, f, g\}$ represents items (events within IT-I operating like a system reboot, file transfers, storage capacity increase/decrease etc.) and a, b is an item set containing two items, which describe two events that belong together or somehow are interdependent. The sequence with ID 1 represents five events that occurred during system runtime where each letter represents an item respectively an IT-I event and letters within curly brackets represent an item set. The sequence with ID 1 describes that events a and b are events that occur together, that c happened after them, then f and g happened together, then g happened and finally, e was the last event within the first sequence. A sequence has the length of k if it contains k items where $k = |A_1| + |A_2| + \ldots + |A_n|$. Therefore, the first sequence is a sequence with a length of $|\{a,b\}| + |\{c\}| + |\{f,g\}| + |\{g| + |\{e| = 7$ items. The SDB is a list of sequences with unique identifiers (ID's) that contains different sequences. Tab. 7 could be a SDB, which contains sequences of four different system components. If a sequence S_a is contained in sequence S_b, then S_a is said to be a subsequence of S_b. For example, the sequence $\langle\{a\},\{b\},\{c\}\rangle$ is a subsequence of the first sequence while $\langle\{c\},\{b\},\{a\}\rangle$ is not because of different item order. The goal of SPM is to identify interesting subsequences within a SDB, which could show interesting relationships for the user. There are different measures available that evaluate "how interesting" a subsequence is but in the original problem of SPM the support measure (SM, or sup) is used. The SM is defined as the number of sequences that contain S_a. For example, the SM or sup(S_a) of the sequence $\langle\{a\},\{g\}\rangle$ is 2 because this pattern appears in the sequences with ID 1 and 3 in the SDB. Some authors define the SM as a ratio, i.e., relative support (RS), where RS is defined as the number of sequences containing S_a divided by the number of sequences in the SDB or $relSup(S_a) = \sup(S_a)/|SDB|$. For example, RS for the sequence $\langle\{a\},\{g\}\rangle$ is 2/4. A sequence is said to be a sequential pattern if and only if $\sup(S_a) \geq minsup$ where *minsup* is a user defined threshold respectively the must be minimum SM value, which decides if the sequential pattern is part of the result or not. The *minsup* is normally displayed as a percentage where a *minsup* of 50% in the above mentioned example stands for two out of the four sequences shown in the SDB. Tab. 8 shows the result for identified sequential patterns when *minsup* is set to 75%.

Ever since the SPM problem was introduced there were several studies and different algorithms presented, which can be generally classified into two groups, apriori-based and pattern-growth-based algorithms [14]. The bases for a breed of algorithms that depend on the apriori-based approach are the Apriori [132] (the algorithm mentioned in the example above with the result of Tab. 7) and the AprioriAll [128] algorithm. The following key features or main characteristics for "apriori-based" algorithms (ABA) are mentioned given in [133]. First, ABA are described as breadth-first or level-wise algorithms because all the k-sequences are constructed in the k^{th} iteration when traversing the search space. Second, after a number of candidate sequences are generated, a pruning method tests each candidate one by one for satisfying some user specified constraints and removing those candidates, which do not fit as a sequential pattern. Third, ABA often require multiple scanning iterations on the data basis with high processing time and I/O costs. Well known ABA are the GSP (generalized sequential pattern) [134], the SPIRIT (sequential pattern mining with regular expression constraints) [135], the SPADE (sequential pattern discovery using equivalence) [136] and the SPAM [137] (sequential pattern mining). Following the ABA of the mid-1990's, first "pattern-growth-based" algorithms (PGBA) were introduced in the early 2000's. As before, key features and main characteristics for PGBA are found in [133].

First, the search space for large candidate sequences can be partitioned, which allows parallel mining of the smaller partitions and results in efficient memory management. Second, the search space is represented as a physical tree data structure, which is then traversed breadth-first or depth-first while pruning is based on the apriori property.

Third, PGBA try the early pruning of candidate sequences and have therefore a performance advantage because of the smaller search space there is less memory required. Well-known PGBA are the FREESPAN (frequent

pattern-projected sequential pattern) [138], the WAP-MINE (web access pattern mine) [139] and the PREFIXSPAN (prefix-projected sequential pattern mining) [140]. For a deeper understanding of SPM, we refer to the respective comprehensive surveys [13], [129], [133], [141]–[143]. The case of IT-I monitoring is about some special situations, which make specific pattern mining types necessary. For example, there is a lot of interest in the patterns that enable the prediction of possible ITS issues or security intrusions, which are kind of patterns that do not happen frequently, at least in well managed IT-I environments, but have a statistically significant dependency for the IT-I operating. The problem is that the *minsup* has to be set very low to discover such infrequent patterns, which may have the consequence that a few interesting patterns get mixed with a large number of unimportant patterns. Furthermore, it is possible that the event data is noisy or skewed in the whole data collection, which has the consequence that some valid patterns could be missed [14]. Think of lost data in case of memory or buffer overflows or data of SD systems, which could be corrupted by human errors. One pattern mining approach that addresses the above mentioned problems is the fully dependent pattern (FDP) or d-pattern, which was applied on a set of network data from a large insurance company [144]. For example, the FDP identified that the three following infrequent events: NIC failure, unreachable destination and "cold start" often occurred together. The last event implies that the router has failed and restarted. Therefore, it would be possible to implement a warning after the occurrence of the first two events, which then allows the execution of proactive activities before the router fails. Another approach to this problem domain is the mutually dependent pattern (MDP) or m-pattern [145], which is similar to the Apriori method (see above), but can use more pruning techniques and does not require a *minsup* [14]. The MDP then was extended allowing the identification of two or more occurrences of the same items (item multisets) by traversing the SDB only once without computing overall multiplicity of items in multisets [146]. Periodicity or time dependencies in event sequences are other important characteristics, which could generate further interesting patterns that could lead to actionable insights. The mining of patterns in case of periodicity is challenging because of the following reasons [147].

1) Periodic behavior is not forced to be persistent, e.g., an exception event within IT-I monitoring occurs and is not available if the exceptional situation is no longer present
2) Time information may be inexact because of clock synchronization lacks, rounding or other timely delays
3) The length of the periods is unknown in advance and the range of periods may span from seconds to days
4) The number of occurrences of a periodic pattern depends on the period and can vary drastically, e.g., a period of one day has seven occurrences in a week while a one minute period has over ten thousand occurrences in a week
5) Noise (missing or random inserted events) may disrupt periodicities

To address the above challenges, the partially periodic event pattern (PPEP) or p-pattern was introduced [147]. The PPEP structures the pattern mining task into two sub-tasks, finding the periods and mining temporary associations. For the second sub-task, a level-wise algorithm is used while for the first sub-task a Chi-Squared test based approach was developed. To identify PPEP two algorithms were presented: the associate-first algorithm, which has a higher tolerance to noise, and the period-first algorithm, which is more computationally efficient. A special discipline within PPEP is the mining of recurrent patterns, which provide further interesting information like seasonal associations between items [148]. Another method for temporal mining is the pairwise temporal dependent pattern (PTDP) or t-pattern [149]. The PTDP focuses on the mining of infrequent events without the need of a predefined time window by structuring the mining task into two sub-tasks: dependence testing/candidate removal using statistical techniques, and identifying the temporal relationships between dependent event types. Generally, in temporal pattern mining, the identification of hidden time lags plays an important role regarding finding trends of incoming events respectively of predicting the future IT-I behavior because the time lag can provide the characterization of the temporal dependencies among events. Mining the time lag between events could generate useful insights for RCA and thus the issue resolution process because it is possible to construct a fault-failure-chain by correlating and merging relevant events that possibly are the issue triggers. In some research on temporal dependency discovering, it is assumed that the only dependency exists between an item a and its first following b but there is the possibility that the dependency exists between an item a and any following b. Furthermore, interleaved dependencies were not explicitly considered [14]. Fig. 13 shows an example of an interleaved dependency between item a and b, in which the dependency exists between every second a and every following third b with a different time lag of 60 to 90 minutes while the time lag between two adjacent a's is 12 hours [150]. Let item a describe the start of a batch job and item b be a performance warning at the data source system resulting from critical workload processing. The situation is that the batch job is processed two times the day with one processing overnight and one processing in the noon while a random number of users are working on the data source system, which then in combination with the batch job workload results in the performance warning. The example shows that it can be challenging to deal with a fixed time window for the mining

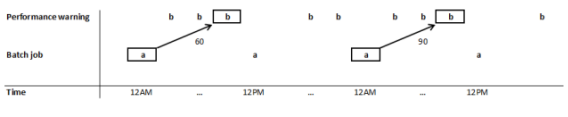

FIGURE 13. Temporal dependency with a various and interleaved time lag.

of hidden temporal dependency patterns when the time lag is various. Interleaved dependencies exist in various situations.

The general problem of identifying such interleaved time lags between events was addressed in the following works [150]–[152].

As mentioned before, temporal data consists of two types of data: time series data and event data. A limited number of references focuses on the topic of mining correlations between this continuous time series data and temporal event data [14]. Especially IT-I monitoring is an example for which such an analysis can generate interesting insights. An approach based on the analysis of IT-I data for incident analysis is presented in the following work [153]. The approach is about to discover three important aspects: discovery of correlations between time series and events, the temporal order of the dependency, and the monotonic effect of the dependency. Furthermore, there are three terms defined to demonstrate that there is a correlation between the two types of temporal data which are shown in Fig. 14 [153]. The front sub-series (F) is the part of the time series before an event happened, the rear sub-series (R) is the part of the time series after an event happened and a further sub-series (T) is constructed by random sampling. The series F, R and T use the same time window size. The idea is that there is a correlation between a time series S and an event E when there is a noticeable change in S upon every occurrence of E. Following definitions for the existing correlation between E and S are given.

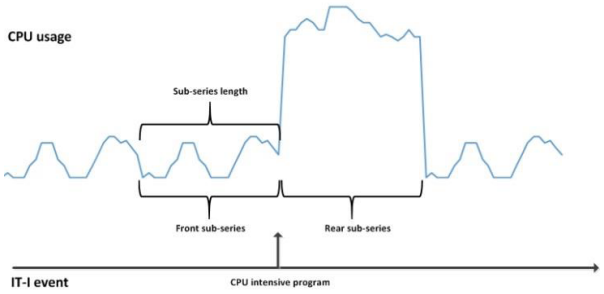

FIGURE 14. Front sub-series and rear sub-series.

1) E and S are correlated and E often occurs after the changes of S, denoted as $S \rightarrow E$, if and only if the distribution of F statistically differs from T
2) E and S are correlated and E often occurs before the changes of S, denoted as $E \rightarrow S$, if and only if the distribution of R statistically differs from T
3) E and S are correlated if there is a relationship of $S \rightarrow E$ or $E \rightarrow S$

B. EVENT SUMMARIZATION

Due to the constantly increasing importance of ITS, the complexity of IT-I environments grows proportionally strong. The result is that there are more and more IT-I components, which generate log files and events. The problem is that this huge amount of events generates so much information that for a human system analyst it is hard to handle this amount even if the events are pre-selected or pre-qualified by techniques like event pattern mining. The character of event pattern mining tends to identify and to return all interesting patterns, which could be widely more than the processing capacity of the system analyst. Beside the mining approach for events, there is a complementary approach available, called event summarization, which allows a less granular and thus a kind of high-level view on the IT-I status. Pattern mining can be distinguished from event summarization upon three characteristics: functionality, result representation, and result granularity [14]. The function of event summarization is often used for exploration and investigation than for detailed analysis. It provides a global overview to understand the IT-I status quickly and determines further suggestions for an analysis. The direction or the goals of the analysis are often not known at the beginning but get step by step discovered and defined within the analysis process. Furthermore, event summarization can be used as pre-step for event pattern mining because it possibly generates clues for the pattern mining parameterization. The result representation is thus more flexible w.r.t. the requirements of the system analyst. While event patterns are presented as the discovered patterns or rules, event summarization for example allows the representation as segmentation model [154], [155]; hidden Markov model [109], [156]; graph [157] or event relationship network [158], [159]. The result granularity of event summarization is coarser than the result of pattern mining and allows only a high-level view on the relationships. These characteristics lead into the following properties an event summarization system should have [154].

1) Brevity and accuracy: The input data should be transformed in short summaries which precisely describe the status of the IT-I components
2) Global data description: The summary should give an overview of the global structure and the timely evolution of the event sequence
3) Local pattern identification: A glance on the summary should grant a high-level view on information about normal or suspicious events respectively combinations of events that depend on each other
4) Parameter-free: Important information and useful results should be generated without the need for extra tuning by the analyst

Fig. 15 shows an example of an event sequence with events a, b, c, the identified segmental grouping, and an event summary result displayed as a segmentation model. The timeline consists of 30 timestamps [154].

The segmental grouping identifies three segments with the timespans [1], [11], [12], [20], [21], [30]. Within the segments, there are two event groups identified (events that occur with a similar frequency and the rest). In the first segment with the timespan [1], [11] the events a and b are grouped together because they appear much more frequently than event c. Following this logic, the groups in [12] and [20] are b, c and a respectively the groups in [21] and [30] are a, c and b. The darker the color in Fig. 9 is, the higher the

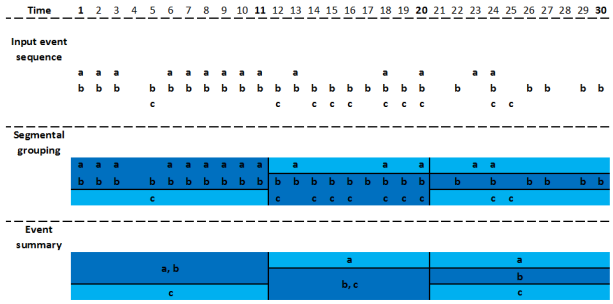

FIGURE 15. Event summarization as segmentation model.

occurrence frequency of the events within the group is. This example shows that event summarization provides a simple overview, even if a large number of events occurred. This results in a quick situational understanding for the system analyst who then is able to start a deeper analysis if necessary as indicated by the first orientation using the event summary. Although there are different methods for event summarization available, the methods can be generally categorized into two groups: summarizing with frequency change [154], [156] and summarizing with temporal dynamics [158], [159]. The first method corresponds to the example shown in Fig. 9 and identifies global intervals in the whole event sequence and then summarizes the events into groups within these intervals by a local model which groups events having a similar frequency. The second method allows revealing the temporal dynamic of the segments but fails to provide information about the temporal dynamics among events [14].

VIII. SELECTION GUIDANCE

When facing the problem of selecting the proper method for the application at hand, the vast lot of possibilities calls for a systematic approach to narrow down the candidates to zero in on the best method. To this end, we propose a series of questions designed to "rule out" or "point out" certain methods and to provide hints on where to look further. Though the workflow along these questions could be arranged as a decision tree, such a graphical representation would become large and confusing and be redundant since the same follow-up question may follow multiple precursor considerations. The questions to ponder about for a selection include (but are not limited to) the following:

1. What kind of data is available?
   - 1.1 Is the data textual (e.g., tickets)? If yes, text-mining methods should be considered (statistical methods mostly rely on numeric data and are as such not directly useful here) [41]–[45].
   - 1.2 Is the data numeric or categorical? If yes, then statistical and ML methods may apply, such as NN, classification techniques and others. For categorical data with *ordered* categories, numerical ML methods can be applied after a mapping of categories to numeric representatives (so that the order is preserved). If the data is categorical yet not ordered (e.g., attributes like "protocol $\in$ {TCP, UDP}" in a log file), then certain statistical methods such as logistic regression or decision trees may be considered for classification problems.
     - 1.2.1 For classification tasks, further questions should be answered:
       - 1.2.1.1 Is it a binary classification problem or could the instance belong to more than one class (e.g., if a system failure is caused by more than one fault)? If yes, multi-label classification should be considered [9], [65], [67].
       - 1.2.1.2 Is it a flat classification problem or are further information and dependencies along a given class taxonomy necessary (e.g., for automated ticket routing along a given processing team structure)? If yes, hierarchical classification should be considered [46], [49], [51].
       - 1.2.1.3 Is the problem a combination of 1.2.1.1 and 1.2.1.2 (e.g., for RCA where information about dependencies along a component structure are important as well as multiple marks of issue triggers within this structure)? If yes, hierarchical multi-label classification should be considered [56], [58], [64].
   - 1.3 If the data is a mix of both of the above types (textual and numeric), then a partitioning into data of homogeneous types and application of the respective methods is advisable. Mixing data of different kinds in the same model should generally be avoided or done with the greatest care. It is, however, possible to combine the outcomes of different models (provided that they are conceptually compatible, e.g., all numeric, or on the same categorical scale), yet this also requires care. Combining heterogeneous models into a meta-model of prediction is an interesting area of research with yet not too many results.
2. How much data is available?
   - 2.1 Large data sets: these enable the application of ML methods, which are generally "data hungry" [160], [161].
   - 2.2 Small data sets: small data volumes typically do not merit the application of many ML methods (especially NN, clustering, or others). Here, rule-based methods should be considered. Fuzzy logic is a popular method of letting experts specify rules with subjective uncertainty [162], [163].
3. Is training data available? Typically, this concerns matters of classification, where a set of records should carry a designated label attribute on which the ML algorithm shall be trained to assign or recognize.
   - 3.1 If training data is available, then every supervised method of ML is applicable.

3.2 If training data is unavailable, then two subsequent considerations come into play:
- 3.2.1 Should the labeling be done automatically? This means applying clustering methods, which spare lots of human efforts at the price of lower accuracy (e.g., in the case of the log file to event converting where less accuracy is acceptable [122], [124], [125]).
- 3.2.2 Can the labeling be done manually? This requires larger human effort, but can be more accurate (e.g., in the case of a maximum precise decision boundary for prediction of failure-prone system states [12]). Are large expert communities with necessary domain knowledge available? If yes, the manual labeling effort can be shared within a crowdsourcing approach (e.g., in the case of labeling large amounts of training data for tickets [87]).

4 What do we want to learn from or do with the data? Several options are available, and most precursor surveys confine themselves to providing overviews about one or more of the following:
- 4.1 Prediction of system failures [12].Recognition of interdependencies: these often manifest themselves as *patterns* (i.e., coincidental occurrences of several events) and can further be divided into:
  - 4.1.1 Pattern recognition in case of infrequent events (e.g., the amount of error events to all events is small but has a statistically significant dependency for the IT-I operators [144]–[146]).
  - 4.1.2 Pattern recognition in case of periodicity (e.g., for non-persistent events [147] or identification of recurrent events [148]).
  - 4.1.3 Pattern recognition in case of temporal dependencies (e.g., if the time window for the event analysis is unknown [149]).
  - 4.1.4 Pattern recognition in case of interleaved events with a hidden time lag (e.g., the correlation of a batch job and a performance warning depend on variable factors like the actual free system resources and the intensity of the workload [150]–[152]).
- 4.2 Getting an overview (to understand the system and its dynamics as such): here, various of the aforementioned summary methods are applicable, and the follow-up questions are:
  - 4.2.1 On what should the summary be based? This could be:
    - 4.2.1.1 Frequency change of events [154], [156]
    - 4.2.1.2 Similarities of temporal dynamics [159]
  - 4.2.2 How should the summary be represented? This could be as:
    - 4.2.2.1 Segmentation model [154], [155]
    - 4.2.2.2 Hidden Markov model [109], [156]
    - 4.2.2.3 Graph [157]
    - 4.2.2.4 Event relationship network [158], [159]
- 4.3 Should the findings from the model be explainable (e.g., allowing for a ''drill-down'')?
  - 4.3.1 If yes, then black-box approaches like NN or fuzzy logic rule-based techniques should be avoided, for their typical lack of explanatory features despite a perhaps well-justified performance. Many statistical models like regression, SVM or others are transparent and come with rich theory and reasons to why certain outcomes are obtained.
  - 4.3.2 If not, then the full palette of models becomes available.

5 Evaluation of the chosen method: some methods come with a rich theory to analyze the model quality (e.g., statistical tests, regression analysis, etc.), while others are only open to general techniques of quality assessment like confusion matrices, ROC curves or similar (e.g., NN, or similar).
- 5.1 The quality of prediction methods can be evaluated by different metrics (e.g., precision), which are based on a confusion matrix [9], 65], [67].
- 5.2 The quality of classification methods can be additionally evaluated by loss functions (e.g., HMC-Loss) [52], [55], [77], [78].
- 5.3 The quality of pattern recognition can be evaluated by different metrics (e.g., support or lift) [164].
- 5.4 The quality of event summaries can be evaluated by the compression ratio [154], [156], [159]. Accuracy should be evaluated by experiments based on synthetic datasets with freely set parameters (e.g., number of patterns) so that the summary can be tested against a known ground-truth. Real-world event data sets tend to be uncontrollable since they are generated automatically with a random size and random content. Though, experiments based on real-world datasets should not be left out of scope because the evaluation of subjective criteria (e.g., is the summary informative but easy enough to understand?) by the target group is important, too.

Remark: Before the application of any method, a data quality and completeness check are generally advisable. Is the data complete? Regardless of how many records are there, are they carrying all attributes with known, and if so reliable, values? The two main preparatory tasks here are:

1. Outlier cleaning: this should involve as much domain expertise as possible, since automated methods, say using clustering, can only provide pointers to questionable data, but the line between outliers and inliers is often fuzzy and reliably decidable only using expert knowledge.
2. Treatment of missing values: provided that the data is not systematically missing, the three standard methods include:
   - 2.1 Deleting records that are incomplete: this is the default in many software packages, and

systematically correct, yet with the caveat of perhaps substantially reducing the available lot of data (up to cutting down big data to ''small data'')

2.2 Filling in missing values by ''interpolation'' using the remaining data (either in the same record by regression models, or within the set of other known values for the missing attribute, which is bootstrapping, or a combination of both).
If the gaps are filled using the rest of the data (no matter which model is used to compute the missing values), we stress that information-theoretically, such a method cannot ''add'' anything to the dataset, since the gaps are filled with information that is in the dataset already. So, the ''new'' value is in any case redundant. For methods that draw random values, these are either dependent on the remaining data, in which they add redundancy again, or stochastically independent, in which case the added information does not necessarily say anything ''useful'' beyond what we know already.

2.3 Assigning the gaps a special representative value (e.g., treating the absence of a data item as a category/value of its own). Though avoiding issues of adding redundancy only or reducing the data lot, reasoning from this perspective requires care: after all, we would draw conclusions occasionally based on the *absence* of information, so that we implicitly induce an *open* or *closed world assumption* here.

IX. CONCLUSION

In this survey, we discussed different methods to extend ITSM best practice frameworks and processes to ensure a high ITS availability. We did a comprehensive literature review on present research in the fields of ITSM, ticket analysis, OFP methods, and IT-I event analysis and hoped to give readers a good first orientation if they are interested in these topics. The survey provides the theoretical knowledge necessary for the understanding of the different data-driven techniques like HMC, extraction of textual information or data mining for event sequences. The theoretical perspective is not limited to research in the IT-area. It rather includes relevant basic research of the mentioned topics and adapts this knowledge on examples that represent real-world situations in IT-I operating. Our selection of presented methods depends on relevant stages within a common maintenance procedure for ITS. We showed that this procedure can further be improved by the use of data-driven techniques and that its optimization is not limited to processual best practices. These improvements help to face actual ITSM challenges by, e.g., reducing system downtimes or by preventing system failures to ensure flawless business operations. Furthermore, these improvements allow high acting ability without the need for more human resources even if the complexity of the IT-I increases. From a cost perspective, this is an important advantage.

We provide a selection guideline, specific for the ITSM domain but perhaps enjoying wider applicability, which combines necessary questions for data analytics projects with the linkage to present research works. This guideline should enable readers a fast induction and ease usability of the methods presented. Likewise, the guideline suggests a selection of suitable methods for different applications at hand. To best of our knowledge, such a guideline with a focus on analytics in the ITSM domain was missing so far and offers a new combined perspective for practitioners.

An open gap in the IT-operations research field seems to be work that focuses on comparison and evaluation of these different methods. This confusing situation and the resulting lack of clarity could be the reason why it is hard to declare a specific selection of the available data-driven techniques as best practices and to implement them as part of, e.g., ITIL. Another possible research direction for future work could be the use of IT-I event analysis and symptoms monitoring in a complementary manner. Both allow the prediction of system failures but were rarely explored in combination. Such an analysis of relationships between system performance measures and error event occurrence could yield further interesting insights. Related to this topic, it could be interesting to explore if the identification of a failure-prone system state from system performance perspective correlates with the occurrence of specific events and vice-versa. This knowledge could be used for improved labeling of normal and failure-prone system states as a decision boundary for, e.g., anomaly detection techniques.

Nevertheless, we are sure that even the here mentioned methods can improve IT-I operating to ensure an ITS availability at a maximum level.

ACKNOWLEDGMENT

The authors would like to thank the anonymous reviewers for having made invaluable suggestions to the structure and content of the paper. Furthermore, they would like to thank Stefanie Alex (department: VDF-VMK) from Volkswagen Financial Services AG and Christoph Jansen (department: K-FIMI) from Volkswagen AG for evaluating our selection guideline.

REFERENCES

[1] M. Kresse and M. Bause, *ITIL V3—Alles was man Wissen Muss*. Bad Homburg, Germany: Serview, 2011.
[2] J. F. Rockart, *The Changing Role of the Information Systems Executive: A Critical Success Factors Perspective*. Cambridge, MA, USA: MIT Press, 1982.
[3] M. Beims and M. Ziegenbein, *IT-Service-Management in Der Praxis Mit ITIL*, 4th ed. Munich, Germany: Carl Hanser Verlag, 2014.
[4] M. Beims, *IT-Service Management in Der Praxis Mit ITIL*, 2nd ed. Munich, Germany: Carl Hanser Verlag, 2010.
[5] R. Böttcher, *IT-Servicemanagement Mit ITIL V3: Einführung, Zusammenfassung und Übersicht der elementaren Empfehlungen*, 2nd ed. Hannover, Germany: Heise, 2010.
[6] S. Lacy and I. Macfarlane, *Service Transition*. Norwich, U.K.: The Stationery Office, 2011.

[7] I. Giurgiu *et al.*, "On the adoption and impact of predictive analytics for server incident reduction," *IBM J. Res. Develop.*, vol. 61, no. 1, pp. 98–109, Jan./Feb. 2017.

[8] (Mar. 2017). *Delivering Value to Today's Digital Enterprise: The State of IT Service Management, 2017*. Accessed: Oct. 16, 2018. [Online]. Available: https://www.forbes.com/forbesinsights/bmc_itsm/index.html

[9] M.-L. Zhang and Z.-H. Zhou, "A review on multi-label learning algorithms," *IEEE Trans. Knowl. data Eng.*, vol. 28, no. 8, pp. 1819–1837, Aug. 2014.

[10] M. Sokolova and G. Lapalme, "A systematic analysis of performance measures for classification tasks," *Inf. Process. Manag.*, vol. 45, no. 4, pp. 427–437, 2009.

[11] V. Chandola, A. Banerjee, and V. Kumar, "Anomaly detection: A survey," *ACM Comput. Surv.*, vol. 41, no. 3, Art. No. 15, 2009.

[12] F. Salfner, M. Lenk, and M. Malek, "A survey of online failure prediction methods," *ACM Comput. Surv.*, vol. 42, no. 3, 2010, Art. no. 10.

[13] N. R. Mabroukeh and C. I. Ezeife, "A taxonomy of sequential pattern mining algorithms," *ACM Comput. Surv.*, vol. 43, no. 1, 2010, Art. no. 3.

[14] T. Li *et al.*, "Data-driven techniques in computing system management," *ACM Comput. Surv.*, vol. 50, no. 3, 2017, Art. no. 45.

[15] M. Marrone and L. M. Kolbe, "Einfluss von IT-service-management-frameworks auf die IT-organisation," *Wirtschaftsinformatik*, vol. 53, no. 1, pp. 5–19, 2011.

[16] L. M. Orlov, *Make IT Matter For Business Innovation*. Cambridge, MA, USA: Forrester Research, 2005.

[17] C. E. Bogan and M. J. English, *Benchmarking for Best Practices: Winning Through Innovative Adaptation*. New York, NY, USA: McGraw-Hill, 1994.

[18] J. van Bon, M. Pieper, A. van der Veen, and T. Verheijen, *Foundations of IT Service Management Based on ITIL*, 2nd ed. Zaltbommel, The Netherlands: Van Haren, 2007.

[19] A. Hochstein, G. Tamm, and W. Brenner, "Service oriented IT management: Benefit, cost and success factors," in *Proc. 13th Eur. Conf. Inf. Syst., Inf. Syst. Rapidly Changing Economy*, Regensburg, Germany, 2005, p. 98.

[20] B. C. Potgieter, J. H. Botha, and C. Lew, "Evidence that use of the ITIL framework is effective," in *Proc. 8th Annu. Conf. Nat. Advisory Committee Comput. Qualifications*, Tauranga, New Zealand, 2005, pp. 160–167.

[21] A. Cater-Steel, W.-G. Tan, and M. Toleman, "Challenge of adopting multiple process improvement frameworks," in *Proc. 14th Eur. Conf. Inf. Syst. (ECIS)*, Goteborg, Sweden, 2006.

[22] C. Pollard and A. Cater-Steel, "Justifications, strategies, and critical success factors in successful ITIL implementations in U.S. and Australian companies: An exploratory study," *Inf. Syst. Manage.*, vol. 26, no. 2, pp. 164–175, 2009.

[23] M. Mora, M. Raisinghani, R. V. O'Connor, and O. Gelman, "An extensive review of IT service design in seven international ITSM processes frameworks: Part I," *Int. J. Inf. Technol. Syst. Approach*, vol. 7, no. 2, pp. 83–107, 2014.

[24] S. Bahsani, A. Semma, and N. Sellam, "Towards a new approach for combining the IT frameworks," *Int. J. Comput. Sci. Issues*, vol. 12, no. 1, pp. 118–123, 2015.

[25] A. Yazici, A. Mishra, and P. Kontogiorgis, "IT service management (ITSM) education and research: Global view," *Int. J. Eng. Edu.*, vol. 31, no. 4, pp. 1071–1080, 2015.

[26] M. Marrone, M. Kiessling, and L. M. Kolbe, "Are we really innovating? An exploratory study on innovation management and service management," in *Proc. IEEE Int. Conf. Manage. Innov. Technol. (ICMIT)*, Singapore, Jun. 2010, pp. 378–383.

[27] A. Cater-Steel and W.-G. Tan, "itSMF Australia 2005 Conference: Summary of ITIL adoption survey responses," Univ. Southern Queensland, Toowoomba, QLD, Australia, Tech. Rep., 2008.

[28] C. Zeng, "Large scale data mining for IT service management," FIU Electronic Theses and Dissertations, 2016. [Online]. Available: http://digitalcommons.fiu.edu/etd/3051

[29] E.-E. Jan, J. Ni, N. Ge, N. Ayachitula, and X. Zhang, "A statistical machine learning approach for ticket mining in IT service delivery," in *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage.*, Ghent, Belgium, May 2013, pp. 541–546.

[30] A. Maksai, J. Bogojeska, and D. Wiesmann, "Hierarchical incident ticket classification with minimal supervision," in *Proc. IEEE Int. Conf. Data Mining*, Shenzhen, China, Dec. 2014, pp. 923–928.

[31] V. Shimpi, M. Natu, V. Sadaphal, and V. Kulkarni, "Problem identification by mining trouble tickets," in *Proc. 20th Int. Conf. Manage. Data*, Hyderabad, India, 2014, pp. 76–86.

[32] G. A. Di Lucca, M. Di Penta, and S. Gradara, "An approach to classify software maintenance requests," in *Proc. IEEE Int. Conf. Softw. Maintenance*, Montreal, QC, Canada, Oct. 2002, pp. 93–102.

[33] D. D. Lewis and M. Ringuette, "A comparison of two learning algorithms for text categorization," in *Proc. 3rd Annu. Symp. Document Anal. Inf. Retr.*, Las Vegas, NV, USA, 1994, pp. 81–93.

[34] S. Dumais, J. Platt, and D. Heckerman, "Inductive learning algorithms and representations for text categorization," in *Proc. 7th Int. Conf. Inf. Knowl. Manage.*, Bethesda, MD, USA, 1998, pp. 148–155.

[35] T. Joachims, "Text categorization with support vector machines: Learning with many relevant features," in *Proc. Eur. Conf. Mach. Learn.*, Chemnitz, Germany, 1998, pp. 137–142.

[36] Z.-Q. Wang, X. Sun, D.-X. Zhang, and X. Li, "An optimal SVM-based text classification algorithm," in *Proc. IEEE Int. Conf. Mach. Learn. Cybern.*, Dalian, China, Aug. 2006, pp. 1378–1381.

[37] B.-F. Zhang, J.-S. Su, and X. Xu, "A class-incremental learning method for multi-class support vector machines in text classification," in *Proc. IEEE Int. Conf. Mach. Learn. Cybern.*, Dalian, China, Aug. 2006, pp. 2581–2585.

[38] B. Rujiang and L. Junhua, "A novel conception based texts classification method," in *Proc. IEEE Int. e-Conf. Adv. Sci. Technol.*, Dajeon, South Korea, Mar. 2009, pp. 30–34.

[39] S.-B. Kim, K.-S. Han, H.-C. Rim, and S. H. Myaeng, "Some effective techniques for naive Bayes text classification," *IEEE Trans. Knowl. Data Eng.*, vol. 18, no. 11, pp. 1457–1466, Nov. 2006.

[40] M. J. Meena and K. R. Chandran, "Naïve Bayes text classification with positive features selected by statistical method," in *Proc. 1st Int. Conf. Adv. Comput.*, Chennai, India, 2009, pp. 28–33.

[41] D. M. Blei, A. Y. Ng, and M. I. Jordan, "Latent Dirichlet allocation," *J. Mach. Learn. Res.*, vol. 3, pp. 993–1022, Mar. 2003.

[42] C. Chemudugunta, P. Smyth, and M. Steyvers, "Modeling general and specific aspects of documents with a probabilistic topic model," in *Proc. 19th Int. Conf. Neural Inf. Process. Syst.*, 2006, pp. 241–248.

[43] D. M. Blei and J. D. Lafferty, "A correlated topic model of science," *Ann. Appl. Statist.*, vol. 1, no. 1, pp. 17–35, 2007.

[44] D. M. Blei, T. L. Griffiths, and M. I. Jordan, "The nested Chinese restaurant process and Bayesian nonparametric inference of topic hierarchies," *J. ACM*, vol. 57, no. 2, 2010, Art. no. 7.

[45] P. D. Turney, "Learning algorithms for keyphrase extraction," *Inf. Retr.*, vol. 2, no. 4, pp. 303–336, 2000.

[46] C. N. Silla, Jr., and A. A. Freitas, "A survey of hierarchical classification across different application domains," *Data Mining Knowl. Discovery*, vol. 22, nos. 1–2, pp. 31–72, 2011.

[47] S. Dumais and H. Chen, "Hierarchical classification of Web content," in *Proc. 23rd Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Athens, Greece, 2000, pp. 256–263.

[48] M. E. Ruiz and P. Srinivasan, "Hierarchical text categorization using neural networks," *Inf. Retr.*, vol. 5, no. 1, pp. 87–118, 2002.

[49] O. Dekel, J. Keshet, and Y. Singer, "Large margin hierarchical classification," in *Proc. 21st Int. Conf. Mach. Learn.*, Banff, AB, Canada, 2004, p. 27.

[50] P. Berkhin, "A survey of clustering data mining techniques," in *Grouping Multidimensional Data Recent Advances in Clustering*, J. Kogan, C. Nicholas, M. Teboulle, Eds. Berlin, Heidelberg: Springer, 2006, pp. 25–72.

[51] N. Cesa-Bianchi, C. Gentile, and L. Zaniboni, "Incremental algorithms for hierarchical classification," *J. Mach. Learn. Res.*, vol. 7, pp. 31–54, Jan. 2006.

[52] N. Cesa-Bianchi, C. Gentile, and L. Zaniboni, "Hierarchical classification: Combining Bayes with SVM," in *Proc. 23rd Int. Conf. Mach. Learn.*, Pittsburgh, PA, USA, 2006, pp. 177–184.

[53] R. Babbar, I. Partalas, E. Gaussier, and M.-R. Amini, "On flat versus hierarchical classification in large-scale taxonomies," in *Proc. 26th Int. Conf. Neural Inf. Process. Syst.*, Lake Tahoe, NV, USA, 2013, pp. 1824–1832.

[54] A. Kosmopoulos, I. Partalas, E. Gaussier, G. Paliouras, and I. Androutsopoulos, "Evaluation measures for hierarchical classification: A unified view and novel approaches," *Data Mining Knowl. Discovery*, vol. 29, no. 3, pp. 820–865, 2015.

[55] C. Zeng, T. Li, L. Shwartz, and G. Y. Grabarnik, "Hierarchical multi-label classification over ticket data using contextual loss," in *Proc. IEEE Netw. Oper. Manage. Symp. (NOMS)*, Krakow, Poland, May 2014, pp. 1–8.

[56] H. Blockeel, L. Schietgat, J. Struyf, S. Dzeroski, and A. Clare, "Decision trees for hierarchical multilabel classification: A case study in functional genomics," in *Proc. 10th Eur. Conf. Principle Pract. Knowl. Discovery Databases*, Berlin, Germany, 2006, pp. 18–29.

[57] C. Vens, J. Struyf, L. Schietgat, S. Džeroski, and H. Blockeel, "Decision trees for hierarchical multi-label classification," *Mach. Learn.*, vol. 73, no. 2, pp. 185–214, Nov. 2008.

[58] W. Bi and J. T. Kwok, "Multi-label classification on tree- and DAG-structured hierarchies," in *Proc. 28th Int. Conf. Int. Conf. Mach. Learn.*, Bellevue, WA, USA, 2011, pp. 17–24.

[59] I. Dimitrovski, D. Kocev, S. Loskovska, and S. Dzeroski, "Hierarchical annotation of medical images," *Pattern Recognit.*, vol. 44, nos. 10–11, pp. 2436–2449, 2011.

[60] I. Dimitrovski, D. Kocev, S. Loskovska, and S. Dzerosk, "Hierarchical classification of diatom images using ensembles of predictive clustering trees," *Ecol. Informat.*, vol. 7, no. 1, pp. 19–29, 2012.

[61] Z. Ren, M.-H. Peetz, S. Liang, W. van Dolen, and M. de Rijke, "Hierarchical multi-label classification of social text streams," in *Proc. 37th Int. ACM SIGIR Conf. Res., Develop. Inf. Retr.*, Gold Coast, QLD, Australia, 2014, pp. 213–222.

[62] G. Valentini, S. Köhler, M. Re, M. Notaro, and P. N. Robinson, "Prediction of human gene-phenotype associations by exploiting the hierarchical structure of the human phenotype ontology," in *Proc. Conf. Bioinformat. Biomed. Eng.*, Granada, Spain, 2015, pp. 66–77.

[63] Z. Sun, Y. Zhao, D. Cao, and H. Hao, "Hierarchical multilabel classification with optimal path prediction," *Neural Process. Lett.*, vol. 45, no. 1, pp. 263–277, 2017.

[64] L. Zhang, S. K. Shah, and I. A. Kakadiaris, "Hierarchical multi-label classification using fully associative ensemble learning," *Pattern Recognit.*, vol. 70, pp. 89–103, Oct. 2017.

[65] M. S. Sorower, "A literature survey on algorithms for multi-label learning," Ph.D. dissertation, Dept. Comput. Sci., Oregon State Univ., Corvallis, OR, USA, 2010.

[66] G. Tsoumakas and I. Katakis, "Multi-label classification: An overview," *Int. J. Data Warehousing Mining*, vol. 3, no. 3, pp. 1–13, 2007.

[67] E. Gibaja and S. Ventura, "A tutorial on multilabel learning," *ACM Comput. Surv.*, vol. 47, no. 3, 2015, Art. no. 52.

[68] Y. Yang, "An evaluation of statistical approaches to text categorization," *Inf. Retr.*, vol. 1, nos. 1–2, pp. 69–90, 1999.

[69] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognit. Lett.*, vol. 27, no. 8, pp. 861–874, Jun. 2006.

[70] J. Davis and M. Goadrich, "The relationship between precision-recall and ROC curves," in *Proc. 23rd Int. Conf. Mach. Learn.*, Pittsburgh, PA, USA, 2006, pp. 233–240.

[71] R. E. Schapire and Y. Singer, "Improved boosting algorithms using confidence-rated predictions," *Mach. Learn.*, vol. 37, no. 3, pp. 297–336, 1999.

[72] S. Zhu, X. Ji, W. Xu, and Y. Gong, "Multi-labelled classification using maximum entropy method," in *Proc. 28th Annu. Int. ACM SIGIR Conf. Res. Develop. Inf. Retr.*, Salvador, Brazil, 2005, pp. 233–240.

[73] D. Powers, "Evaluation: From precision, recall and F-measure to ROC, informedness, markedness and correlation," *J. Mach. Learn. Technol.*, vol. 2, no. 1, pp. 37–63, 2011.

[74] P. Liang and S. Ermon, "Lecture 2: Machine learning I (Lecture CS221: Artificial intelligence: Principles and Techniques at Stanford University)," Stanford Univ., Stanford, CA, USA, 2017, accessed: Oct. 16, 2018. [Online]. Available: http://web.stanford.edu/class/cs221/

[75] F. R. Bach, D. Heckerman, and E. Horvitz, "Considering cost asymmetry in learning classifiers," *J. Mach. Learn. Res.*, vol. 7, pp. 1713–1741, Dec. 2006.

[76] J. Rousu, C. Saunders, S. Szedmak, and J. Shawe-Taylor, "Kernel-based learning of hierarchical multilabel classification models," *J. Mach. Learn. Res.*, vol. 7, pp. 1601–1626, Jul. 2006.

[77] N. Cesa-Bianchi and G. Valentini, "Hierarchical cost-sensitive algorithms for genome-wide gene function prediction," in *Proc. 3rd Int. Workshop Mach. Learn. Syst. Biol.*, 2009, pp. 14–29.

[78] W. Bi and J. T. Kwok, "Hierarchical multilabel classification with minimum Bayes risk," in *Proc. IEEE Int. Conf. Data Mining*, Brussels, Belgium, Dec. 2012, pp. 101–110.

[79] C. Zeng, W. Zhou, T. Li, L. Shwartz, and G. Y. Grabarnik, "Knowledge guided hierarchical multi-label classification over ticket data," *IEEE Trans. Netw. Service Manage.*, vol. 14, no. 2, pp. 246–260, Jun. 2017.

[80] W. Bi and J. T. Kwok, "Bayes-optimal hierarchical multilabel classification," *IEEE Trans. Knowl. Data Eng.*, vol. 27, no. 11, pp. 2907–2918, Nov. 2015.

[81] Q. Shao, Y. Chen, S. Tao, X. Yan, and N. Anerousis, "EasyTicket: A ticket routing recommendation engine for enterprise problem resolution," *VLDB Endowment*, vol. 1, no. 2, pp. 1436–1439, 2008.

[82] P. Sun, S. Tao, X. Yan, N. Anerousis, and Y. Chen, "Content-aware resolution sequence mining for ticket routing," in *Proc. Int. Conf. Bus. Process Manage.*, Hoboken, NJ, USA, 2010, pp. 243–259.

[83] S. Agarwal, R. Sindhgatta, and B. Sengupta, "SmartDispatch: Enabling efficient ticket dispatch in an IT service environment," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Beijing, China, 2012, pp. 1393–1401.

[84] M. Altintas and A. C. Tantug, "Machine learning based ticket classification in issue tracking systems," in *Proc. Int. Conf. Artif. Intell. Comput. Sci. (AICS)*, Bandung, Indonesia, 2014, pp. 195–207.

[85] M. Botezatu, J. Bogojeska, I. Giurgiu, H. Voelzer, and D. Wiesmann, "Multi-view incident ticket clustering for optimal ticket dispatching," in *Proc. 21st ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Sydney, NSW, Australia, 2015, pp. 1711–1720.

[86] J. Xu, R. He, W. Zhou, and T. Li, "Trouble ticket routing models and their applications," *IEEE Trans. Netw. Service Manage.*, vol. 15, no. 2, pp. 530–543, Jun. 2018.

[87] Y. Diao, H. Jamjoom, and D. Loewenstern, "Rule-based problem classification in IT service management," in *Proc. IEEE Int. Conf. Cloud Comput.*, Bangalore, India, 2009, Sep. pp. 221–228.

[88] R. Gupta, H. Karanam, L. Luan, D. Rosu, and C. Ward, "Multi-dimensional knowledge integration for efficient incident management in a services cloud," in *Proc. IEEE Int. Conf. Services Comput.*, Bangalore, India, Sep. 2009, pp. 57–64.

[89] A. Medem, M.-I. Akodjenou, and R. Teixeira, "TroubleMiner: Mining network trouble tickets," in *Proc. IFIP/IEEE Int. Symp. Integr. Netw. Manage.-Workshops*, New York, NY, USA, Jun. 2009, pp. 113–119.

[90] C. Kadar, D. Wiesmann, J. Iria, D. Husemann, and M. Lucic, "Automatic classification of change requests for improved IT service quality," in *Proc. Annu. SRII Global Conf.*, San Jose, CA, USA, 2011, pp. 430–439.

[91] R. Liu and J. Lee, "IT incident management by analyzing incident relations," in *Proc. 10th Int. Conf. Service-Oriented Comput.*, Shanghai, China, 2012, pp. 631–638.

[92] J. Bogojeska, D. Lanyi, I. Giurgiu, G. Stark, and D. Wiesmann, "Classifying server behavior and predicting impact of modernization actions," in *Proc. 9th IEEE Int. Conf. Netw. Service Manage.*, Zurich, Switzerland, 2013, pp. 59–66.

[93] L. Tang, T. Li, L. Shwartz, and G. Y. Grabarnik, "Identifying missed monitoring alerts based on unstructured incident tickets," in *Proc. IEEE Int. Conf. Netw. Service Manage.*, Zurich, Switzerland, Oct. 2013, pp. 143–146.

[94] J. Bogojeska, I. Giurgiu, D. Lanyi, G. Stark, and D. Wiesmann, "Impact of HW and OS type and currency on server availability derived from problem ticket analysis," in *Proc. IEEE Netw. Oper. Manage. Symp. (NOMS)*, Krakow, Poland, May 2014, pp. 1–9.

[95] I. Giurgiu, J. Bogojeska, S. Nikolaiev, G. Stark, and D. Wiesmann, "Analysis of labor efforts and their impact factors to solve server incidents in datacenters," in *Proc. 14th IEEE/ACM Int. Symp. Cluster, Cloud, Grid Comput. (CCGrid)*, Chicago, IL, USA, May 2014, pp. 424–433.

[96] J. Xu, L. Tang, and T. Li, "System situation ticket identification using SVMs ensemble," *Expert Syst. Appl.*, vol. 60, pp. 130–140, Oct. 2016.

[97] N. Madaan, G. Singh, A. Kumar, and G. B. Dasgupta, "Neev: A cognitive support agent for content improvement in hardware tickets," in *Proc. IFIP/IEEE Symp. Integr. Netw. Service Manage.*, Lisbon, Portugal, May 2017, pp. 239–246.

[98] A. Sailer, R. Mahindru, Y. Song, and X. Wei, "Using machine learning and probabilistic frameworks to enhance incident and problem management: Automated ticket classification and structuring," in *Artificial Intelligence: Concepts, Methodologies, Tools, and Applications*, Information Reso Management Association, Ed. Hershey, PA, USA: Information Science Reference, 2017, pp. 2975–3012.

[99] J. Xu, H. Zhang, W. Zhou, R. He, and T. Li, "Signature based trouble ticket classification," *Future Gener. Comput. Syst.*, vol. 78, no. 1, pp. 41–58, 2018.

[100] W. Zhou, L. Tang, T. Li, L. Shwartz, and G. Y. Grabarnik, "Resolution recommendation for event tickets in service management," *IEEE Trans. Netw. Service Manage.*, vol. 13, no. 4, pp. 954–967, Dec. 2016.

[101] A. K. Kalia, J. Xiao, M. F. Bulut, M. Vukovic, and N. Anerousis, "Cataloger: Catalog recommendation service for IT change requests," in *Proc. Int. Conf. Service-Oriented Comput.*, Malaga, Spain, 2017, pp. 545–560.

[102] D. P. Muni, S. Roy, Y. T. Y. J. J. L. Chiang, A. J.-M. Viallet, and N. Budhiraja, "Recommending resolutions of ITIL services tickets using Deep Neural Network," in *Proc. 4th ACM IKDD Conf. Data Sci.*, Chennai, India, 2017, p. 14.

[103] W. Zhou *et al.*, "STAR: A system for ticket analysis and resolution," in *Proc. 23rd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Halifax, NS, Canada, 2017, pp. 2181–2190.

[104] D. P. Siewiorek and R. S. Swarz, *Reliable Computer Systems: Design and Evaluation*, 3rd ed. Wellesley, MA, USA: A. K. Peters, 1998.

[105] D. Tang and R. K. Iyer, "Dependability measurement and modeling of a multicomputer system," *IEEE Trans. Comput.*, vol. 42, no. 1, pp. 62–75, Jan. 1993.

[106] Y. Liang, Y. Zhang, A. Sivasubramaniam, M. Jette, and R. Sahoo, "BlueGene/L failure analysis and prediction models," in *Proc. Int. Conf. Dependable Syst. Netw. (DSN)*, Philadelphia, PA, USA, Jun. 2006, pp. 425–434.

[107] J. Valdman, "Log file analysis," Univ. West Bohemia, Pilsen, Czech Republic, Tech. Rep. DCSE/TR-2001-04, 2001.

[108] J. L. Hellerstein, S. Ma, and C.-S. Perng, "Discovering actionable patterns in event data," *IBM Syst. J.*, vol. 41, no. 3, pp. 475–493, 2002.

[109] W. Peng, T. Li, and S. Ma, "Mining logs files for data-driven system management," *ACM SIGKDD Explor. Newslett. Natural Lang. Process. Text Mining*, vol. 7, no. 1, pp. 44–51, 2005.

[110] A. Oliner and J. Stearley, "What supercomputers say: A study of five system logs," in *Proc. 37th Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw.*, Jun. 2007, pp. 575–584.

[111] A. J. Oliner, A. Aiken, and J. Stearley, "Alert detection in system logs," in *Proc. IEEE Int. Conf. Data Mining*, Pisa, Italy, Dec. 2008, pp. 959–964.

[112] W. Xu, L. Huang, A. Fox, D. Patterson, and M. Jordan, "Mining console logs for large-scale system problem detection," in *Proc. 3rd Conf. Tackling Comput. Syst. Problems Mach. Learn. Techn.*, San Diego, CA, USA, 2008, p. 4.

[113] J. Gao, G. Jiang, H. Chen, and J. Han, "Modeling probabilistic measurement correlations for problem determination in large-scale distributed systems," in *Proc. 29th IEEE Int. Conf. Distrib. Comput. Syst. (ICDCS)*, Montreal, QC, Canada, Jun. 2009, pp. 623–630.

[114] K. Nagaraj, C. Killian, and J. Neville, "Structured comparative analysis of systems logs to diagnose performance problems," in *Proc. 9th USENIX Conf. Netw. Syst. Design Implement.*, San Jose, CA, USA, Apr. 2012, p. 26.

[115] N. El-Sayed and B. Schroeder, "Reading between the lines of failure logs: Understanding how HPC systems fail," in *Proc. 43rd Annu. IEEE/IFIP Int. Conf. Dependable Syst. Netw.*, Budapest, Hungary, Jun. 2013, pp. 1–20.

[116] I. Fronza, A. Sillitti, G. Succi, M. Terho, and J. Vlasenko, "Failure prediction based on log files using random indexing and support vector machines," *J. Syst. Softw.*, vol. 86, no. 1, pp. 2–11, 2013.

[117] A. Nandi, A. Mandal, S. Atreja, G. B. Dasgupta, and S. Bhattacharya, "Anomaly detection using program control flow graph mining from execution logs," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, San Francisco, CA, USA, 2016, pp. 215–224.

[118] M. Du, F. Li, G. Zheng, and V. Srikumar, "DeepLog: Anomaly detection and diagnosis from system logs through deep learning," in *Proc. ACM SIGSAC Conf. Comput. Commun. Secur.*, Dallas, TX, USA, 2017, pp. 1285–1298.

[119] Q. Fu, J.-G. Lou, Y. Wang, and J. Li, "Execution anomaly detection in distributed systems through unstructured log analysis," in *Proc. 9th IEEE Int. Conf. Data Mining*, Dec. 2009, pp. 149–158.

[120] R. Vaarandi, "A data clustering algorithm for mining patterns from event logs," presented at the 3rd IEEE Workshop IP Oper. Manage., Kansas City, MO, USA, 2003.

[121] L. Tang and T. Li, "LogTree: A framework for generating system events from raw textual logs," in *Proc. IEEE Int. Conf. Data Mining*, Sydney, NSW, Australia, Dec. 2010, pp. 491–500.

[122] L. Tang, T. Li, and C. S. Perng, "LogSig: Generating system events from raw textual logs," in *Proc. 20th ACM Int. Conf. Inf. Knowl. Manage.*, Glasgow, Scotland, 2011, pp. 785–794.

[123] A. Makanju, N. Zincir-Heywood, and E. E. Milios, "A lightweight algorithm for message type extraction in system application logs," *IEEE Trans. Knowl. Data Eng.*, vol. 24, no. 11, pp. 1921–1936, Nov. 2012.

[124] R. Vaarandi and M. Pihelgas, "LogCluster—A data clustering and pattern mining algorithm for event logs," in *Proc. IEEE Int. Conf. Netw. Service Manage.*, Barcelona, Spain, Nov. 2015, pp. 1–7.

[125] M. Du and F. Li, "Spell: Streaming parsing of system event logs," in *Proc. IEEE 16th Int. Conf. Data Mining*, Barcelona, Spain, Dec. 2016, pp. 859–864.

[126] P. He, J. Zhu, Z. Zheng, and M. R. Lyu, "Drain: An online log parsing approach with fixed depth tree," in *Proc. IEEE Int. Conf. Web Services*, Honolulu, HI, USA, Jul. 2017, pp. 33–40.

[127] P. He, J. Zhu, S. He, J. Li, and M. R. Lyu, "An evaluation study on log parsing and its use in log mining," in *Proc. IEEE/IFIP Int. Conf. Dependable Syst. Netw.*, Toulouse, France, Jun. 2016, pp. 654–661.

[128] R. Agrawal and R. Srikant, "Mining sequential patterns," in *Proc. 11th IEEE Int. Conf. Data Eng.*, Washington, DC, USA, Mar. 1995, pp. 3–14.

[129] P. Fournier-Viger, J. C.-W. Lin, R. U. Kiran, Y. S. Koh, and R. Thomas, "A survey of sequential pattern mining," *Data Sci. Pattern Recognit.*, vol. 1, no. 1, pp. 54–77, 2017.

[130] J. Lin, E. Keogh, L. Wei, and S. Lonardi, "Experiencing SAX: A novel symbolic representation of time series," *Data Mining Knowl. Discovery*, vol. 15, no. 2, pp. 107–144, 2007.

[131] A. Camerra, T. Palpanas, J. Shieh, and E. Keogh, "iSAX 2.0: Indexing and mining one billion time series," in *Proc. IEEE Int. Conf. Data Mining*, Dec. 2010, pp. 58–67.

[132] R. Agrawal and R. Srikant, "Fast algorithms for mining association rules," in *Proc. 20th Int. Conf. Very Large Data Bases*, Santiago, Chile, 1994, pp. 487–499.

[133] I. Khan and A. Jain, "A comprehensive survey on sequential pattern mining," *Int. J. Eng. Res. Technol.*, vol. 1, no. 4, pp. 32–39, 2012.

[134] R. Srikant and R. Agrawal, "Mining sequential patterns: Generalizations and performance improvements," in *Proc. 5th Int. Conf. Extending Database Technol., Adv. Database Technol.*, 1996, pp. 1–17.

[135] M. N. Garofalakis, R. Rastogi, and K. Shim, "SPIRIT: Sequential pattern mining with regular expression constraints," in *Proc. 25th Int. Conf. Very Large Data Bases*, 1999, pp. 7–10.

[136] M. J. Zaki, "SPADE: An efficient algorithm for mining frequent sequences," *Mach. Learn.*, vol. 42, nos. 1–2, pp. 31–60, 2001.

[137] J. Ayres, J. Flannick, J. Gehrke, and T. Yiu, "Sequential pattern mining using a bitmap representation," in *Proc. 8th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Edmonton, AB, Canada, 2002, pp. 429–435.

[138] J. Han, J. Pei, B. Mortazavi-Asl, Q. Chen, U. Dayal, and M.-C. Hsu, "FreeSpan: Frequent pattern-projected sequential pattern mining," in *Proc. 6th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Boston, MA, USA, 2000, pp. 355–359.

[139] J. Pei, J. Han, B. Mortazavi-Asl, and H. Zhu, "Mining access patterns efficiently from Web logs," in *Proc. 4th Pacific–Asia Conf., Knowl. Discovery Data Mining, Current Issues New Appl.*, 2000, pp. 396–407.

[140] J. Pei, J. Han, B. Mortazavi-Asl, and H. Pinto, "PrefixSpan: Mining sequential patterns efficiently by prefix-projected pattern growth," in *Proc. 17th Int. Conf. Data Eng.*, 2001, pp. 215–224.

[141] Q. Zhao and S. S. Bhowmick, "Sequential pattern mining: A survey," Nanyang Technol. Univ., Singapore, Tech. Rep. 2003118, 2003.

[142] C. Chand, A. Thakkar, and A. Ganatra, "Sequential pattern mining: Survey and current research challenges," *Int. J. Soft Comput. Eng.*, vol. 2, no. 1, pp. 185–193, 2012.

[143] R. Boghey and S. Singh, "Sequential pattern mining: A survey on approaches," in *Proc. IEEE Int. Conf. Commun. Syst. Netw. Technol. (CSNT)*, Gwalior, India, Apr. 2013, pp. 670–674.

[144] F. Liang, S. Ma, and J. L. Hellerstein, "Discovering fully dependent patterns," in *Proc. SIAM Int. Conf. Data Mining*, 2002, pp. 511–527.

[145] S. Ma and J. L. Hellerstein, "Mining mutually dependent patterns," in *Proc. IEEE Int. Conf. Data Mining*, San Jose, CA, USA, Nov./Dec. 2001, pp. 409–416.

[146] N. Kiyota, S. Shimamura, and K. Hirata, "Extracting mutually dependent multisets," in *Proc. Int. Conf. Discovery Sci.*, Kyoto, Japan, 2017, pp. 267–280.

[147] S. Ma and J. L. Hellerstein, "Mining partially periodic event patterns with unknown periods," in *Proc. IEEE Int. Conf. Data Eng.*, Heidelberg, Germany, Apr. 2001, pp. 205–214.

[148] R. U. Kiran, H. Shang, M. Toyoda, and M. Kitsuregawa, "Discovering recurring patterns in time series," in *Proc. Int. Conf. Extending Database Technol.*, Brussels, Belgium, 2015, pp. 97–108.

[149] T. Li, F. Liang, S. Ma, and W. Peng, "An integrated framework on mining logs files for computing system management," in *Proc. 11th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Chicago, IL, USA, 2005, pp. 776–781.

[150] L. Tang, T. Li, and L. Shwartz, "Discovering lag intervals for temporal dependencies," in *Proc. 18th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Beijing, China, 2012, pp. 633–641.
[151] C. Zeng, L. Tang, T. Li, L. Shwartz, and G. Y. Grabarnik, "Mining temporal lag from fluctuating events for correlation and root cause analysis," in *Proc. 10th Int. Conf. Netw. Service Manage. (CNSM)*, Rio de Janeiro, Brazil, 2014, pp. 19–27.
[152] M.-A. Zöller, M. Baum, and M. F. Huber, "Framework for mining event correlations and time lags in large event sequences," in *Proc. IEEE 15th Int. Conf. Ind. Inform. (INDIN)*, Emden, Germany, Jul. 2017, pp. 805–810.
[153] C. Luo *et al.*, "Correlating events with time series for incident diagnosis," in *Proc. 20th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, New York, NY, USA, 2014, pp. 1583–1592.
[154] J. Kiernan and E. Terzi, "Constructing comprehensive summaries of large event sequences," *ACM Trans. Knowl. Discovery Data*, vol. 3, no. 4, 2009, Art. no. 21.
[155] Q.-K. Pham, G. Raschia, N. Mouaddib, R. Saint-Paul, and B. Benatallah, "Time sequence summarization to scale up chronology-dependent applications," in *Proc. 18th ACM Conf. Inf. Knowl. Manage.*, Hong Kong, 2009, pp. 1137–1146.
[156] P. Wang, H. Wang, M. Liu, and W. Wang, "An algorithmic approach to event summarization," in *Proc. ACM SIGMOD Int. Conf. Manage. Data*, Indianapolis, IN, USA, 2010, pp. 183–194.
[157] M. Aharon, G. Barash, I. Cohen, and E. Mordechai, "One graph is worth a thousand logs: Uncovering hidden structures in massive system event logs," in *Proc. Eur. Conf. Mach. Learn. Knowl. Discovery Databases*, Bled, Slovenia, 2009, pp. 227–243.
[158] W. Peng, C. Perng, T. Li, and H. Wang, "Event summarization for system management," in *Proc. 13th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, San Jose, CA, USA, 2007, pp. 1028–1032.
[159] Y. Jiang, C. S. Perng, and T. Li, "Natural event summarization," in *Proc. 20th ACM Int. Conf. Inf. Knowl. Manage.*, Glasgow, Scotland, 2011, pp. 765–774.
[160] T. Hastie, R. Tibshirani, and J. Friedman, *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, 2nd ed. New York, NY, USA: Springer, 2009.
[161] C. M. Bishop, *Pattern Recognition and Machine Learning*. New York, NY, USA: Springer, 2006.
[162] H. Ishibuchi and T. Yamamoto, "Rule weight specification in fuzzy rule-based classification systems," *IEEE Trans. Fuzzy Syst.*, vol. 13, no. 4, pp. 428–435, Sep. 2005.
[163] O. Cordón, M. J. del Jesus, and F. Herrera, "A proposal on reasoning methods in fuzzy rule-based classification systems," *Int. J. Approx. Reasoning*, vol. 20, no. 1, pp. 21–45, 1999.
[164] S. Ventura and J. M. Luna, *Pattern Mining With Evolutionary Algorithms*. New York, NY, USA: Springer, 2016.

PATRICK KUBIAK received the B.Sc. degree in business information technology from the Flensburg University of Applied Sciences, Germany, in 2013, and the M.A. degree in business information technology from the Kiel University of Applied Sciences, Germany, in 2015. He is currently pursuing the Ph.D. degree in informatics with Alpen-Adria-Universität Klagenfurt, Austria.

From 2015 to 2016, he was a Consultant in analytical technologies like Sap Bw On Hana, SAP Lumira, Tableau, and data lake architectures for Reply AG in Düsseldorf, Germany. Since 2017, he has been a member of the Volkswagen Financial Services AG, where he is currently pursuing the Ph.D. Program with the IT-Operations Department. His work focuses on IT application and infrastructure behavior analysis/prediction and other data-driven use cases based on machine learning and statistics.

STEFAN RASS received the double master's degree in mathematics and computer science from Alpen-Adria-Universität Klagenfurt (AAU) in 2005, the Ph.D. degree in mathematics in 2009, and the Habilitation degree in applied computer science and system security in 2014. He is currently an Associate Professor with AAU, teaching courses on theoretical computer science, complexity theory, security, and cryptography. He has authored numerous papers related to security and applied statistics and decision theory in security. He has also co-authored a book *Cryptography for Security and Privacy in Cloud Computing* (Artech House) which is closely related to the project. He has participated in various nationally and internationally funded research projects. His research interests include applied system security, complexity theory, statistics, decision theory, and game-theory.