A Context Model for Holistic Monitoring and Management of Complex IT Environments

Mathias Mormul and Christoph Stach
Institute for Parallel and Distributed Systems
University of Stuttgart
Universitaetsstraße 38, 70569 Stuttgart, Germany
{firstname.lastname}@ipvs.uni-stuttgart.de

***Abstract*—The increased usage of IoT, containerization, and multiple clouds not only changed the way IT works but also the way IT Operations, i. e., the monitoring and management of IT assets, works. Monitoring a complex IT environment leads to massive amounts of heterogeneous context data, usually spread across multiple data silos, which needs to be analyzed and acted upon autonomously. However, for a holistic overview of the IT environment, context data needs to be consolidated which leads to several problems. For scalable and automated processes, it is essential to know what context is required for a given monitored resource, where the context data are originating from, and how to access them across the data silos. Therefore, we introduce the Monitoring Resource Model for the holistic management of context data. We show what context is essential for the management of monitored resources and how it can be used for context reasoning. Furthermore, we propose a multi-layered framework for IT Operations with which we present the benefits of the Monitoring Resource Model.**

***Index Terms*—context model, IT Operations, AIOps, monitoring**

I. Introduction

IT monitoring is a necessary prerequisite for a smooth flow of all processes in a company and the basis of IT Operations. Resource statistics of virtual machines (VMs) and bare-metal servers such as CPU and RAM load are monitored by IT infrastructure monitoring (ITIM) systems and application statistics by application performance monitoring (APM) systems. With the rise of containerization technologies such as Docker[1], containers are monitored and managed as well by orchestration platforms, e. g., Kubernetes[2]. Static information about the environment can be found in the cloud management platforms such as OpenStack[3] or Device Management systems. Additionally, monitoring systems are often designed for specific cloud platforms [1], i. e., each public cloud provider typically provides its own monitoring system, which adds even more monitoring systems to manage. Practice shows that no single monitoring system exists that can take over all monitoring tasks [2] and a recent study showed that many companies use more than ten monitoring systems in parallel [3]. Therefore, this lack of a cross-domain solution leads to context data being stored in multiple silos, gathered by different teams, sometimes with little to no communication in between [2] and leads to a complex management of multiple monitoring systems in parallel.

To handle the increased complexity and massive amounts of context data, *AIOps* (Artificial Intelligence for IT operations), coined by Gartner [4], introduced the use of machine learning to the domain of IT monitoring. Desired consequences are increased automation and a proactive approach instead of a reactive one. Gartner recommends several functionalities to accomplish AIOps such as historical and streaming data management, ingestion of numerical and alphanumerical data, and methods for anomaly detection and root cause determination. Furthermore, Gartner sees the construction of models describing the IT environment as a requirement for AIOps.

However, neither Gartner nor any tools aiming at providing AIOps present such a model. The definition of what context is required for a monitored resource and further analysis, where the context data are originating from and how to access them across the multitude of monitoring and management systems is essential for enabling a holistic overview of the IT environment. For example, relevant context about a virtual machine is given by an ITIM system (e. g., CPU load), the cloud management platform (e. g., IP and date of creation), and Service Level Management systems (e. g., what minimum availability was agreed upon?).

To solve these problems and consolidate all important context data, we introduce the Monitoring Resource Model. We aim at providing a common basis for the functionalities required for AIOps mentioned above. Furthermore, we propose a multi-layered architecture describing the acquisition, management, and analysis of context data and based-upon actions.

The remainder of this article is structured as follows: Section II contains a small background about monitoring systems as well as a motivating scenario. In Section III, we introduce the Monitoring Resource Model. Section IV describes our proposed multi-layered architecture and how the Monitoring Resource Model benefits the individual layers. Section V discusses related work. Lastly, Section VI contains the conclusion of this paper as well as future work.

II. Background and Use Case Scenario

A monitoring system is a tool to collect, store, analyze, and visualize monitoring data [5]. The system administrator defines,

This work is partially funded by the BMWi project IC4F (01MA17008G).

[1]https://www.docker.com/

[2]https://www.kubernetes.io/

[3]https://www.openstack.org/

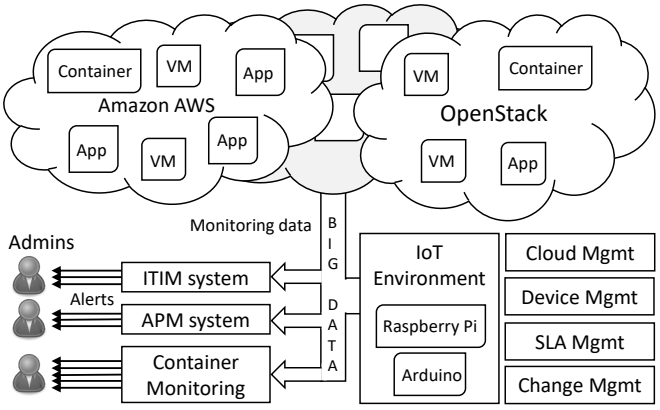

Figure 1. Examplary scenario of monitoring a complex IT environment

which Key Performance Indicators (KPIs) for which resources are required to see the current health status of the resource. In general, the monitoring data are stored centrally at a monitoring server. To ease the management of a monitoring system and present the monitoring data in a clear and concise way, a visualization framework is used. The user can create dashboards to visualize the current status of the monitored resources. However, due to the large amount of monitored resources and their metrics, visualization alone is not a viable option to keep an overview of a monitored environment. Therefore, automatisms are required that analyze incoming monitoring data and alert the system administrator in case of a problem. The default approach to this is the creation of alerting rules, e. g., *"if VM.CPU $>$ 90 % then send email to Admin"*. This rule is automatically evaluated on new monitoring data.

In the following, we introduce a motivating scenario to demonstrate the need for a context model in IT Operations. As shown in Fig. 1, private clouds such as OpenStack as well as public clouds such as Amazon AWS are used. Multi-cloud solutions are helpful when dealing with challenges, such as resilience against data loss caused by the failing of single cloud providers, optimization of costs, and improvement of quality of service [6]. According to Gartner, this strategy becomes common for 70 percent of organizations by 2019 [7]. In parallel, IoT Environments must be monitored as well. ITIM systems are used to monitor the state of VMs and IoT Devices, APM systems monitor applications, and Container Monitoring systems monitor the state of containers. In addition, static information about the monitored resources are available in management systems. Cloud management platforms hold information about the IPs and hypervisor of VMs, their creation date, etc. Analogous, Device Management systems provide information about bare-metal servers and IoT devices. SLA management provides information about agreements with, e. g., the public cloud provider about a guaranteed availability of the provided resources.

As shown, the context data are stored across multiple systems that are not connected to each other and managed by individual personnel. Alerting rules are defined to alert administrators in case of any problems. As an example, the APM system sends an alert about a non-responsive application. However, the application is hosted within a container, which is hosted on a VM, which in fact has a failure and is the root cause. This means that any failure in the VM leads to several cascading alerts. How severe this problem is can be found out by checking the corresponding SLAs. Also the problem might be caused by an erroneous hypervisor which again requires another monitoring system or a look into the cloud management system.

Out of this, we derive the following challenges:

$C1$: **Data silos.** Data silos are the main reason preventing a holistic view of the IT environment [2]. Therefore, the Monitoring Resource Model needs to consolidate context data from multiple data sources.

$C2$: **Big Data.** Multi-cloud scenarios with VMs, containers, applications, and IoT devices lead to a plethora of data points that must be monitored continuously and highly frequently [8]. This leads to a massive amount of heterogeneous data such as time-series data and log data. The Monitoring Resource Model needs to support this heterogeneity to be usable in heterogeneous IT environments.

$C3$: **Alert Fatigue.** In complex IT environments, often, hundreds or even thousands of alerts are created per hour of which many are noise and do not require attention. In general, this leads to alert fatigue – a problem describing that personnel starts ignoring alerts and therefore, might overlook a critical alert. The Monitoring Resource Model needs to consider alerts of the resources as well and aim at reducing the amount of generated alerts.

III. Monitoring Resource Model

Our goal is to provide a Single Pane of Glass to administrators, i. e., integrate information from multiple sources to have a single unified view on a monitored resource including all important context data from monitoring systems, static information from management systems, and dependencies to other resources. For this, we introduce the Monitoring Resource Model for monitored resources in complex IT environments to solve the above-mentioned challenges $C1 - C3$. An excerpt of the Monitoring Resource Model is shown in Fig. 2. Each monitored resource is represented as a *Resource* and requires following attributes:

- **UID:** Each resource is defined by a unique identification (UID). The UID is used to differentiate the resources and reference other resources when they are in relation to each other.
- **KPIs:** KPIs describe a list of Key Performance Indicators for this resource that are monitored by the respective monitoring systems and are used to describe the health status of a monitored resource. Each KPI is defined by a name, e. g., CPU load in percent, and its most recent numerical value. In addition, average values over the last minute and 10 minutes are calculated. As single values of, e. g., the CPU load, can spike for a few seconds, the average over the last one and ten minutes represent

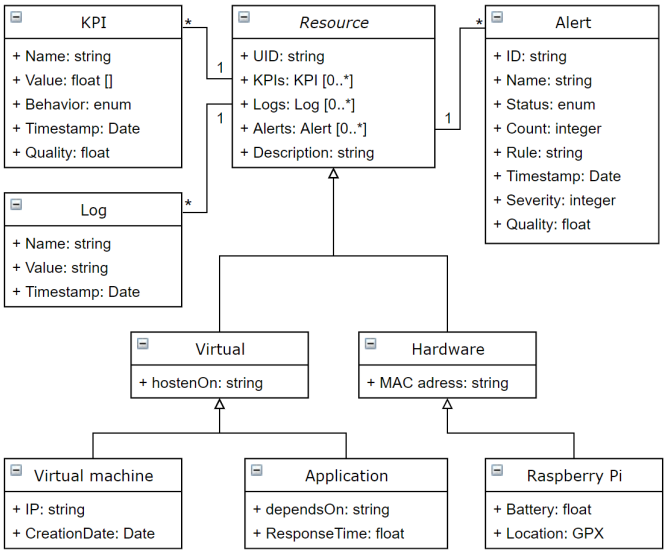

Figure 2. Excerpt of the Monitoring Resource Model

a more reliable view. A timestamp shows the actuality of the value and and the attribute Quality shows how accurate the value is. Furthermore, the behavior of this KPI is classified into one of multiple classes, e. g., *static, periodical*, or *unpredictable*.

- **Log Data:** Log data oftentimes provides more in-depth information about a monitored resource. In comparison to numerical context data, for which alerting rules can be defined that continuously check if new context data exceed predefined thresholds, log data already contains warning or error statements, which signal and direct to the problem. Analogous to KPIs, Log Data contains a list of logs. Each log is defined by a name, i. e., the name of the log file, and its most recent log entry as a string.
- **Alerts:** Alerts is a list of alerts for this resource. Each alert is defined by an ID, name, status, count, rule, timestamp, and severity. Since many alerts of the same type can occur multiple times, an ID is required to distinguish them. The name provides an easily understandable meaning of the alert, e. g., *CPU load critical*. Status describes if the alert is taken care of by personnel, i. e., *open*, *in process*, or *closed*. Alerts may be sent in a regular interval if the problem is not fixed, i. e., the status of the first alert is set to *open* or *in process*. In this case, instead of creating a new *Alert* instance, the attribute count of the original alert is increased by one to signal a recurring alert as it refers to the same problem. Rule describes the alerting rule the alert originates from. Timestamp contains the date of when the alert was sent and severity describes how urgent the response to the alert should be.
- **Description:** Description contains a simple and short description of the resource.

Additional attributes are based on the class of resource, e. g., hardware resources also have a MAC address whereas

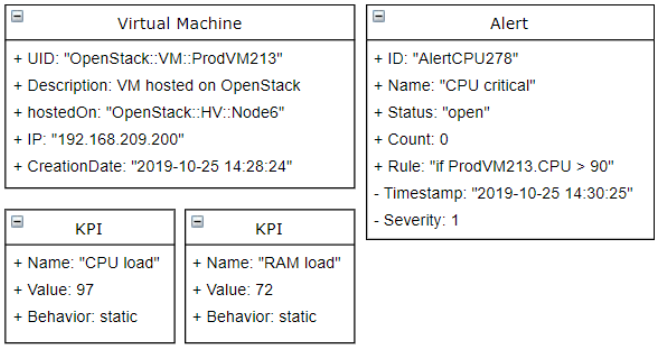

Figure 3. Exemplary MRM instance for a virtual machine

virtual resources have a hostedOn-relation. By referencing a UID of another resource, this relation describes that a virtual resource is hosted on another virtual resource or a hardware resource. For example, an application is hosted on a VM, which is hosted on a hypervisor, which is hosted on a bare-metal server. In this way a topology of the IT environment can be modeled which can be helpful in several ways, e. g., in a root cause analysis as described in our motivating scenario. VM resources may provide static information such as the IP and the creation date, application resources a response time, and IoT devices a battery charge. The MRM can be extended by creating additional resource classes that further specialize existing resource classes, e. g., Amazon EC2 instances, that specialize the Virtual machine class and contain information about SLAs or costs.

In Fig. 3, an exemplary MRM instance of a VM, two of its KPIs and an according alert is shown. However, manually creating instances of this model for each monitored resource is a cumbersome and error-prone task and simply does not scale since VMs and containers are started and shut off again in a matter of minutes. Alerts and KPI value need to be updated automatically without the need for human tasks. Therefore, to fully utilize the Monitoring Resource Model, we need to provide mechanisms that automate the integration of context data and the creation of MRM instances. Therefore, in the next section, we propose an architecture that integrates the MRM into the IT environment.

IV. Architecture

To enable a sensible use of the MRM, we propose a multi-layered framework consisting of five layers that enable the (i) acquisition, (ii) management, (iii) analysis, and (iv) presentation of context data and (v) automated responses. We show how the MRM is used in each layer, what its benefits are, and what additional functionalities are required to support holistic IT Operations. As shown in Fig. 4, our goal is not to replace currently used monitoring and management systems that are already monitoring and managing the resources in the IT environment. Instead, we build upon them and consolidate all important context data to receive a common data basis for holistic management and analysis.

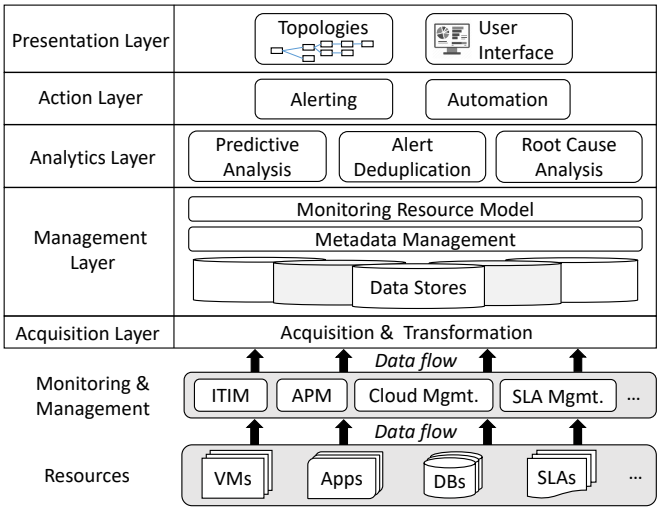

Figure 4. Overview of the framework and its embedding in the IT environment

A. Acquisition Layer

The acquisition of all available context data in regard to the monitoring and management of an IT environment is the first step required for a holistic overview. To make use of the already existing monitoring systems, adapters are created to access important context data from the individual databases, which are defined by the MRM of each monitored resource. Raw data collected by the individual systems are replicated into a new database in our framework. Based on the functionalities of the source systems, aggregated context data are accessed as well, e. g., KPI values averaged over one and ten minutes as defined in the MRM. Lastly, all alert data are accessed. Afterwards, data transformations are performed, e. g., all timestamps are transformed to a unified date schema.

Furthermore, management systems such as OpenStack are accessed as well. Here, we receive context data that are not monitored, e. g., static information such as the IP or the hypervisor of a VM. Especially event data about the instantiation of new VMs is of importance and can be used for the automatic creation of MRM instances of VMs. For this and further event data, OpenStack provides a notification API on which an adapter of our framework can listen to [9].

Of course, it is also possible to collect the raw context data directly from the monitored resources without the need for additional monitoring and management systems by providing corresponding adapters. Therefore, in the future, the data silos may vanish and only one platform is used for IT operations.

B. Management Layer

The previous layer accesses and replicates relevant context data into the data store of our framework. However, due to the heterogeneity of the data, we need to provide individual data stores to support different kinds of data, e. g., Elasticsearch[4] for log data or InfluxDB[5] for time-series data and SQL-based DBMS such as MySQL[6] for relational data.

The MRM instances access the required context data from the individual data stores. However, to handle the complexity of managing massive amounts of data across multiple data stores, a metadata management is essential across all layers of this framework. In the Acquisition Layer, metadata must be captured about where the context data are coming from, their format, date of access, etc. This information is important for MRM instances, e. g., to see how recent KPI values are. In the Management Layer, metadata management is useful for the management of the data stores. For example, context data cannot be stored forever due to the massive amounts. Therefore, retention policies must be defined to decide how long data are stored and what happens after this period, e. g., deletion or aggregation. Also, not all personnel may have the rights to access all context data and MRM instances, therefore, user and access management is required. Managing multiple retention policies and user/access management across the different data stores can be supported by metadata management. Furthermore, the MRM itself has metadata such as version and revision number. In the Analytics Layer, metadata about analysis results such as the timestamp or the used analytics function, its parameters, input data, etc., can be used to reconstruct and understand a specific analysis result.

C. Analytics Layer

The Analytics Layer provides the possibilities to analyze the context data. As mentioned in the introduction, AIOps focuses on the use of artificial intelligence and machine learning methods in IT Operations to gain more insights into the IT environment. For this, both supervised and unsupervised learning is required.

1) Unsupervised learning: In unsupervised learning such as clustering, the goal is to find previously unknown relationships between the data. For example, clustering can be used for the deduplication of alerts. Deduplication describes the detection and removal of duplicate, redundant data. In the case of alerts, features of alerts such as their timestamp can be used to cluster similar alerts. For example, if multiple alerts for different VMs are sent and all their timestamps and hypervisors are the same, then a probable root cause might be an erroneous hypervisor. Furthermore, all those alerts could be aggregated into one alert to reduce the amount of alerts. The same process can be applied to different resources as well. For example, VMs can be clustered based on their features. This way, the anomalous behavior of single VMs can be detected when the VM is *outside* of the clusters, i.e., it behaves differently than the majority of the VMs.

2) Supervised learning: Using advanced machine learning models, the future behavior of a KPI can be predicted. Long-Short-Term-Memory (LSTM) neural networks can be used for this purpose as they are intended to memorize long-term

[4]https://www.elastic.co/

[5]https://www.influxdata.com/

[6]https://www.mysql.com/

dependencies in time-series data [10]. Training data, i. e., the raw context data, are used as input to learn a model describing the behavior of the resource. The predicted behavior is compared with the actual behavior. Deviations outside a certain margin are detected as anomalies, which lead to dynamic alerting thresholds instead of statically predefined thresholds. Furthermore, trends in the data can be recognized, e. g., the values are *rising* or *falling* in the last minutes.

In both cases, it is important to know, which features are relevant and where to find the context data. The MRM provides this information and reduces manual work such as the identification of relevant features. Furthermore, MRM instances can be enriched with the analysis results, e. g., a recognized trend is stored in the *behavior*-attribute of a KPI.

D. Action Layer

The *Action Layer* contains all mechanisms that act based on data from the previous layers. Actions can be divided into two classes: alerting and automation. Alerting creates alerts that are sent to administrators, whereas automation aims at solving problems without the need for human intervention.

1) Alerting: Since all alerts from the data sources are sent to this framework first for reasons such as deduplication, an own alerting system is required to forward the alerts to the responsible personnel via several channels such as email, Telegram[7], Slack[8], etc. Furthermore, since the MRM opens up new possibilities for anomaly detection (described in the previous section), new alerts need to be created as well. An expressive alerting framework is required which needs to support basic logical expressions such as *OR*, *AND*, and *XOR*, comparative operators $<$, $\leq$, $=$, etc. We propose a Complex Event Processing (CEP) engine for this purpose as it fulfills the mentioned functions and further such as the creation of time-windows. Furthermore, CEP is intended to handle massive amounts of data and therefore fits our scenario well.

2) Automation: Due to the rising complexity of IT environments, automated responses to problems are essential. Instead of alerting an administrator, the execution of a script or a workflow can be triggered. Besides problem solving, automation also is required to adapt to changes in the IT environment dynamically. For example, the instantiation of a new VM may lead to the automatic installation of a monitoring agent on said VM.

E. Presentation Layer

As long as human intervention is required, the context data and thereby, the IT environment need to be visualized. The *Presentation Layer* is the final layer and used as the access point to the framework for administrators. Customizable dashboards are used to display all important information of the IT environment such as the MRM instances, analysis results, alerting rules, and the topology of the IT environment. Furthermore, administrative tasks regarding the framework, e. g., retention policies for the data stores, are supported with a graphical user interface.

[7]https://telegram.org/

[8]https://slack.com

Discussion. In the following, we assess our approach in regard to the challenges $C1 - C3$.

MRM instances are provided with the required context data from all available data sources and present a holistic view on a monitored resource. The data silos still remain, however, the consolidation of context data via the Acquisition Layer solves the resulting problem and therefore, challenge $C1$.

Heterogeneous data such as log data and numerical data are supported by the MRM. To manage those massive amounts of heterogeneous context data, an overarching metadata management and multiple heterogeneous data stores in the Management Layer enable the sensible usage of Big Data for the MRM and thereby, solve challenge $C2$.

In the Analytics Layer, sophisticated analytics methods such as clustering and neural networks can be used to reduce the amount of alerts and create more intelligent forms of alerting such as dynamic thresholding. The results of the Analytics Layer act as the input for the Action Layer to either create new alerts or trigger automatisms and thereby further reduce the amount of alerts and diminish alert fatigue to solve challenge $C3$.

V. Related Work

To the best of our knowledge, in literature, there is no context model that focusses on IT Operations. However, in other domains such as eHealth, monitoring is required as well. A patient's heart rate needs to be monitored and doctors need to be alerted in case of emergencies. There are several approaches, e.g., by Anya *et al.* [11] and Guermah *et al.* [12], which present context models for the eHealth domain and their benefits such as context-aware health care applications. However, those context models cannot be applied to IT Operations, since essential characteristics of IT environments cannot be modeled. The focus of our context model is on IT Operations to model and capture all relevant context data such as the relationship between resources.

Our proposed framework can be seen as a Big Data platform, however, with a strong focus on IT Operations. Pääkkönen and Pakkala [13] introduce a reference architecture for Big Data platforms. Their high-level architecture consists of the layers *Data extraction*, *Data loading and pre-processing*, *Data processing*, *Data analysis*, *Data loading and transformation*, and *Interfacing and visualization*. However, an Action Layer or similar to trigger automation or alert personnel is missing which is of utmost importance in IT Operations to react to problems in the IT environment.

Based on this reference architecture, Lipčák *et al.* [14] introduce a Big Data platform for anomaly detection in Smart Grids. They describe and compare the technologies that can be used to implement the individual layers of the platform. The results of the anomaly detection are sent to a messaging queue. Here as well, no subsequent actions based on the analysis results or the data are described.

In practice, many tools exist that claim to provide AIOps such as Moogsoft[9], BMC[10], Splunk[11]. However, essential aspects are missing. Moogsoft is more related to Event Management systems based on the integrations they provide [15] with the additional usage of machine learning methods. Event Management systems collect event data, e. g., alerts, across all monitoring systems for the purposes such as alert deduplication and probable root cause analysis. For this, a context model for events is created. However, compared to our approach, only a small fraction of context data, i. e., only event data, can be accessed across the monitored IT environment and thus a holistic approach to IT Operations cannot be established. BMC presents the main components of an AIOps architecture following Gartner's vision which resembles our framework [16]. However, no information about data management or data storage is given which is essential when working with Big Data. Splunk provides advanced machine learning capabilities with IT service intelligence tool [17]. However, no other data sources can be used and therefore holistic management across the whole IT environment cannot be provided.

Broadcom[12] present a unified data model as the foundation for AIOps [18]. Their model is represented as a directed graph of attributed objects. The graph consists of nodes describing the monitored resources that have an arbitrary number of key-value pairs to describe their attributes, and edges in between to describe their relationships. They describe how the model can be used for analysis and automation. Their approach is very similar to ours, however, no information regarding the attributes of the context model or how the management of context or access to the context data are given. In our approach, we described relevant context data as well as a supporting framework.

VI. Conclusion

IT environments are becoming more and more complex due to the increased usage of IoT, containerization, and multi-clouds. Analogous, IT Operations is becoming increasingly complex. Massive amounts of heterogeneous context data are stored across multiple data silos which prevents a holistic view on the IT environment. System administrators are flooded with alerts from multiple monitoring systems. Current tools include the use of sophisticated machine learning methods and the integration of multiple data sources. However, oftentimes only a small fraction of context data is integrated, i. e., event data, and a data model for monitored resources is missing. Therefore, we introduce the Monitoring Resource Model for the holistic management of context data. Furthermore, we propose an architecture for a multi-layered framework with the Monitoring Resource Model at its core to support the acquisition, management, and analysis of context data as well as an action layer for alerting and automation purposes.

In future work, we plan to refine our context model and the individual layers of our framework. For example, in the Management Layer. we want to evaluate if the concepts of data lakes can be applied to our framework to provide a scalable and flexible data store as basis for advanced analytics and with a sophisticated metadata management. Lastly, we want to assess possibilities for a (fully) automated approach regarding the creation of MRM instances, their relationships to other instances, and connecting the context data to the instances.

[9]https://www.moogsoft.com

[10]https://www.bmc.com/blogs/what-is-aiops/

[11]https://www.splunk.com

[12]https://www.broadcom.com/

References

[1] D. Trihinas *et al.*, "Monitoring Elastically Adaptive Multi-Cloud Services," *IEEE Transactions on Cloud Computing*, vol. 6, no. 3, pp. 800–814, 2018.

[2] S. Niedermaier *et al.*, "On Observability and Monitoring of Distributed Systems – An Industry Interview Study," in *Proceedings of the 17th International Conference on Service-Oriented Computing*, ser. ICSOC '19, Springer, 2019.

[3] Y. Pollack, *The results of our 2019 "Future of Monitoring and AIOps" survey are in*, BigPanda, Inc. https://www.bigpanda.io/blog/the-results-of-our-2019-future-of-monitoring-and-aiops-survey/, Oct. 2019.

[4] P. Prasad and C. Rich, *Market Guide for AIOps Platforms*, Gartner, Inc. https://www.gartner.com/en/documents/3892967/market-guide-for-aiops-platforms, Nov. 2018.

[5] S. Ligus, *Effective Monitoring and Alerting: For Web Operations*. O'Reilly Media, Inc., 2012.

[6] D. Petcu, "Multi-Cloud: Expectations and Current Approaches," in *Proceedings of the 2013 International Workshop on Multi-Cloud Applications and Federated Clouds*, ser. MultiCloud '13, ACM, 2013.

[7] R. van der Meulen, *What Data Center Architects Can Learn from Building Architects*, Gartner, Inc. https://www.gartner.com/smarterwithgartner/what-data-center-architects-can-learn-from-building-architects/, Oct. 2015.

[8] J. S. Ward and A. Barker, "Observing the clouds: A survey and taxonomy of cloud monitoring," *Journal of Cloud Computing*, vol. 3, 24:1–24:30, 2014.

[9] OpenStack Foundation, *Notifications in Nova*, https://docs.openstack.org/nova/latest/reference/notifications.html, Jun. 2020.

[10] S. Hochreiter and J. Schmidhuber, "LSTM Can Solve Hard Long Time Lag Problems," in *Proceedings of the 9th International Conference on Neural Information Processing Systems*, ser. NIPS '96, MIT Press, 1996.

[11] O. Anya *et al.*, "Context-aware knowledge modelling for decision support in e-health," in *Proceedings of the 2010 International Joint Conference on Neural Networks*, ser. IJCNN '10, IEEE, 2010.

[12] H. Guermah *et al.*, "Context modeling and reasoning for building context aware services," in *Proceedings of the 2013 ACS International Conference on Computer Systems and Applications*, ser. AICCSA '13, IEEE, 2013.

[13] P. Pääkkönen and D. Pakkala, "Reference Architecture and Classification of Technologies, Products and Services for Big Data Systems," *Big Data Research*, vol. 2, no. 4, pp. 166–186, 2015.

[14] P. Lipčák *et al.*, "Big Data Platform for Smart Grids Power Consumption Anomaly Detection," in *Proceedings of the 2019 Federated Conference on Computer Science and Information Systems*, ser. FedCSIS '19, IEEE, 2019.

[15] Moogsoft, *Integrations*, https://docs.moogsoft.com/en/integrations.html, Jun. 2020.

[16] S. Paskin, *AIOps in 2020: A Beginner's Guide*, bmc blogs, https://www.bmc.com/blogs/what-is-aiops/, Apr. 2020.

[17] Splunk, Inc., *Splunk IT Service Intelligence (ITSI)*, https://www.splunk.com/en_us/software/it-service-intelligence.html, Jun. 2020.

[18] E. Giral, "AIOps Essentials: A Unified Data Model as the Foundation for AIOps," Broadcom, Inc., White Paper, May 2019.