# Rethinking Monitoring for Cloud Environments: BMC Software AIOps Case Study

Sai Eswar Garapati[1,2], Erhan Giral[1,2], and Smiljana Antonijević[1,2]

[1] BMC Software, Santa Clara, USA
smiljana_antonijevic@bmc.com
[2] BMC Software, 2103 CityWest Boulevard, Houston, TX 77042, USA

**Abstract.** This paper presents the experiences, insights, and solutions resulting from the development of BMC AIOps, a SaaS solution from BMC Software, Inc. (BMC) that applies machine learning and predictive capabilities across IT operations and DevOps environments for real-time, enterprise-wide observability, insights, and automated remediation. We first briefly chronicle the evolution of this product, and then focus on our service-centric perspective on noise reduction and root cause analysis based on incremental differential clustering for causality, and a novel clustering algorithm that BMC built to tackle service models that have thousands of services and millions of devices, and where root cause analysis needs to be performed in near real time.

**Keywords:** AIOPs · Enterprise-wide observability · Service-centric noise reduction · Incremental differential clustering for causality · Root cause analysis

## 1 Introduction

The complex and dynamic nature of today's cloud-based distributed applications have overwhelmed traditional, siloed monitoring approaches. To address this gap, DevOps practitioners and Site Reliability Engineering (SRE) teams are increasingly exploring artificial intelligence (AI) and machine learning (ML) as means to detect and predict problems faster, while recommending automated recovery and remediation steps. Cloud application development and deployment has been simplified, while cloud-based distributed application monitoring and operations are still complex.

In addressing these challenges, BMC Software, Inc. (BMC) is developing a modern SaaS solution that applies machine learning and predictive capabilities across IT operations and DevOps environments for real-time, enterprise-wide observability, insights, and automated remediation. Now in its second year of development, BMC AIOps is an award-winning solution,[1] which has started where good product development should always start—with users and their needs—and it has continued to grow, change, and adapt through a constant dialog between BMC's interdisciplinary product team and its users.

[1] BMC AIOps was awarded "Best Overall AI Solution" in the annual AI Breakthrough Awards program for 2021 (see: https://aibreakthroughawards.com/2021-winners/).

C. Zirpins et al. (Eds.): ESOCC 2022 Workshops, CCIS 1617, pp. 109–115, 2022. The final authenticated version is available online at https://doi.org/10.1007/978-3-031-23298-5_11

In this paper, we share our experiences, insights, and solutions resulting from the development of BMC's AIOps. We first briefly chronicle the evolution of the product, and then focus on one of its key elements – automated root cause analysis (RCA).

## 1.1 BMC AIOps – Background

Our AIOps solution is built on BMC's Autonomous Digital Enterprise platform, which was launched in 2019. In early 2020, BMC' engineers, data scientists, and product managers began charting the AIOps technical and business contours. Soon after, UX designers picked it up, visualizing those initial contours in the prototypes our researchers presented to BMC software users.

AIOps user research began in September of 2020, and it included several iterative phases. The study comprised of seventy-six users from thirty-six strategic customers representing several major industry groups that provide cloud or IT products and/or services. The participants included executives, system admins, service portfolio leads, VPs of cloud products, IT technical directors, principal technical architects, software developers, process advisers, DevOps team members, as well as SREs and performance engineers. In the first set of user research sessions, we sought to understand users' current practices, challenges, and needs related to root cause analysis and event noise reduction. Based on identified user needs, we developed initial solutions for transforming users' predominantly manual practices into modern and efficient AI/ML-based solutions.

These initial solutions were tested with users and iteratively improved based on user feedback. Finally, we continued adding and testing with users further functionalities such as online collaborative features, automated remediation, and so on. The study yielded more than a hundred hours of recorded user interviews, providing us with a wealth of valuable and actionable insights.

In terms of root cause analysis, one of our first goals was to learn whether our customers differentiated between root cause and probable cause, and—if so—what was the differentiator. Across customers and user types, the agreement was that probable cause allows users to explore the nature of data giving a probability of what might be the root cause. Pertaining to the visual display of root cause information, users identified service model topology as the most important element of the BMC AIOPs solution: "It comes right to the root causes, and it shows the relationship in the impacted path, so this is everything I would have expected to see", one user commented.

Related to event noise reduction, we learned that users employ a set of techniques, from simplistic versions of deduplication and correlation through tacit knowledge to more advanced correlation approaches. Efforts such as clustering events and collating sites to reduce the number of alerts and modify their priority, or filtering out noise through the normalization and enrichment processes, have mostly been done manually, but automation is increasingly desired. Our customers thus enthusiastically embraced the prospect of AI/ML based noise reduction.

Yet uncertainties about the actual results and feasibility lingered, stemming from customers' uneven maturity, the awareness of general AI/ML limitations, as well as from the need to better understand what is considered noise and what would be algorithmically discarded and/or combined to reduce noise. As one user put it: "Noise reduction based on AI is the Holy Grail from a service provider perspective. But how do you do that?".

One of our answers to this question was the service centric perspective on noise reduction, which we detail in the following section.

## 2 A Service Centric Perspective on Noise Reduction

Not every anomaly, outlier or even failure is necessarily a problem that leads to business interruption. In fact, in a sufficiently complex environment there is always some background event load, almost like the constant, cosmic background noise. Some of these events, however, are leading indicators that turn into problems with real business impact.

In the past, various attempts have been made to reduce the noise operators face by using online or offline event processing algorithms that relied on event similarities to calculate correlations. Such techniques, however, often fall short in noise reduction because correlation does not necessarily mean causation. Also, most proposed techniques involve heavy hyperparameter tuning that simply pushes the model accuracy responsibility to the end users. The traditional way to cluster events is to identify correlation based on text/time and apply partitioning based clustering like K-Means with configured K [1]. There are others in the industry who employed a configuration threshold based on density and spectral clustering using correlations identified from textual and service domains which suffer from manual tuning and lack of causality issues.

But let's first identify what is noise. There are different kinds of noise events:

- Short Term Flapping Events: These are the events that constantly transition between the critical state and the normal state in a short period. These alarms are commonly generated due to disturbances on metrics configured with alarms, primarily when the metrics are operating nearer to their thresholds.
- Long Term Flapping Alarms: Repeating alarms frequently make transitions between critical and non-critical states with more extended periods. These are generated by repeated on-off actions on devices or regular oscillatory disturbances in metrics.
- Standing Alarms: Another set of noise alarms are standing alarms or alarms that remain in an active state for a prolonged duration. The primary reason for these alarms is typically the inefficiencies in operations and maintenance.
- Alarm Storms/Floods: Finally, Alarm Floods occur when an abnormal situation occurs in some environment. The fault may spread to many other places through interconnections between devices and process units. These are the most important groups of alarms that signify any abnormality in the environment and contain the root cause and huge alarms from the affected components.

BMC believes that if we are to reduce the noise maximally, establishing causal relationships between events should be the goal. Such an endeavor would be best informed if the target service is first modelled for its known components (sources of events) and their transactional, as well as resource dependencies.

There are three integrated layers of topology data contained in our AIOps Solution. The topmost layer contains the connections between software components; for example, it captures the calling relationships between front-end servers and back-end servers. The Middle layer contains Infrastructure topology data between different Virtualized and

Physical Infrastructures like relationships between Containers and Virtual Machines. The Third Layer contains the Network infrastructure, including relationships between networking components like switches and routers. All these layers are connected across to give the IT infrastructure a combined topology.

That kind of modelling would have been an uphill battle in the past, as such descriptive models are often brittle. However, today's observability landscape is rich with tools and methodologies that allow us to collect near-real-time topological data from all layers of business services, all the way from end user workflows down to the network and infrastructure topologies that these workflows exercise. The difficult part, though, is to integrate the topologies that have different ontologies into a coherent knowledge graph that can catalog transactional and resource dependencies in an environment such that analysis algorithms can be situationally aware about the sources of these events.

We create a knowledge graph of the IT Infrastructure domain. The knowledge graph contains known relationships across different monitored entities for which metrics and events are collected. The knowledge graph uses a graph-based data model to represent domain knowledge. This framework allows us to capture data from experts manually or through machine learning. The knowledge graph is a directed labelled graph. Nodes and edges have a well-defined meaning. In our data model, a node represents entities, and edges represent relation. Furthermore, BMC's ADE platform features a vendor agnostic, integrative connector framework that allows us to build on rich topological information, no matter what the observability stack is composed of in the environment.

## 3 Incremental Differential Clustering for Causality

We believe maximal event noise reduction is only possible when events are sorted based on causality; once ordered for causality, it is trivial to separate the root cause events from symptomatic events that have transpired on the same dependency chain after the root cause event. Furthermore, when the causal chains are grouped by the root cause event, seemingly unrelated but causally linked incidents are corelated. Hence, we postulate that identification of the root cause ought to offer the best possible reduction of noise.

For this reason, the BMC team built a novel clustering algorithm that leverages observed topologies to reason about causal distances of event pairs by relying on observed and ontological distances of the sources of these events to cluster causally linked events together. We make no assumption about the size and number of these clusters, but rely on the topology to guide us to autotune our clustering. The resulting algorithm thus requires zero maintenance from the user both in terms of data modelling as well as in terms of hyperparameter tuning or training.

Our clustering relies on a knowledge graph that graphically maintains various domain-specific bits of knowledge, so that we can reason about the encountered events with ontological inference. This allows us to generalize what is known about anomalies in technologies and frameworks into cause-and-effect transmission conduits that point us to the root cause. A situation/cluster comprises events associated with the same or different host that are aggregated based on their occurrence, message, topology, or a combination of these factors. Events are collected from multiple sources across infrastructure, application, and network resources available from various monitoring solution vendors. If we can identify the noise into a group and identify the root cause, we consider it a success, as mentioned earlier.

Pairwise event comparison sounds expensive, and it is. Any naïve approach to pairwise event evaluation will yield quadratic runtime; hence we employ a small world approximation on the service graph while computing the causal distances between event pairs. Since the service models we tackle often have thousands of services and millions of devices, and root cause analysis needs to be performed in near real time, using a small world approximation keeps our comparison space in check (Fig. 1).

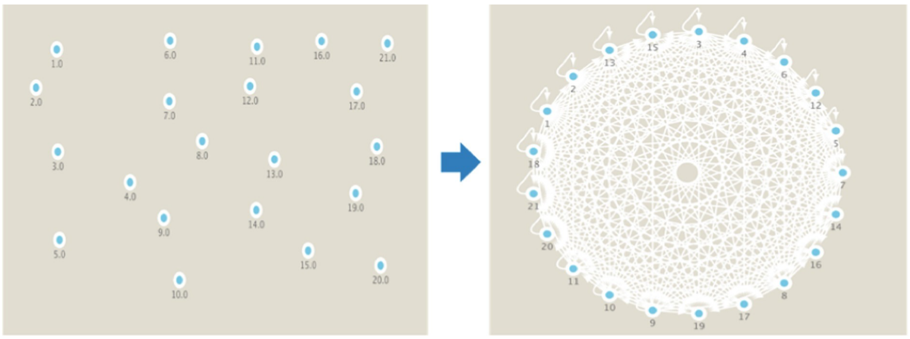


**Fig. 1.** Events depicted as nodes (left); all possible causal links between these events as K21 (right).

This conversion ensures we scale in logarithmic time complexity, as opposed to the quadratic time complexity of naïve pairwise causal distance evaluation. This technique gives us the ability to incrementally add and delete events in real time. It also acts as a regularizer by subsampling the data in a multi-layered fashion, reducing noise further in the process. We found that this approximation improves our accuracy, and, most likely due to the spatial locality it implies, is not a compromise. The dense graph is transformed into a Delaunay graph retaining long range links along with short range links, and it is further processed to extract the minimum arborescence tree. The arborescence tree is then used to identify the optimum distances where the differential clustering algorithm can recognize all possible graph cuts and identify the natural rate of change for the graph separation. This extracted tree exposes the natural characteristics of the graph and its distances to all nodes in directed paths (Fig. 2).

We track the rate of change on these trees to gauge how active it really is, as an operator would be interested in separating active emerging cases from dormant (perhaps mitigated) cases. We then find the cluster boundaries using causal distances by calculating the rate of change of their causal distances. Those directed clusters that are naturally stable for longer causal distances are retained and otherwise merged into a much more stable directed cluster. Finally, the resulting clusters indicate active situations and all their causally related events ordered by causality.

Note that this approach enables us to handle changes in the distribution and scale of the events without any manual tuning and to adapt in a dynamic fashion. At the same

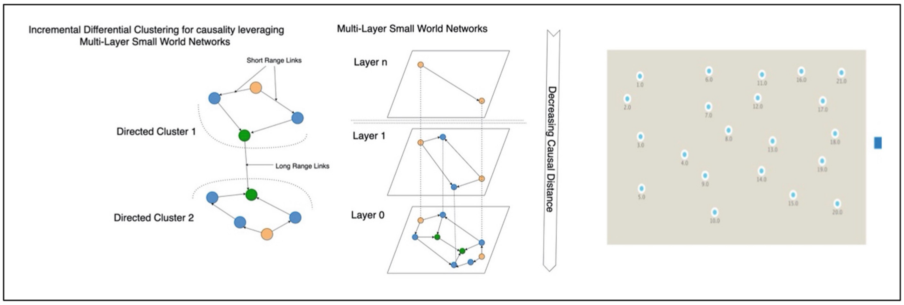

Incremental Differential Clustering for causality leveraging Multi-Layer Small World Networks
Short Range Links
Directed Cluster 1
Long Range Links
Directed Cluster 2
Multi-Layer Small World Networks
Layer n
Layer 1
Layer 0
Decreasing Causal Distance

**Fig. 2.** Event causality search space after small world approximation Delaunay graph (right).

time, we will be able to identify directed clusters at different causal distances according to underlying event distribution in a much more intuitive and natural way.

Each of these clusters now represent a causally linked graph of event signatures. We call these structures Situations and use them to build a narrative around the incident, its root cause, and impact. Since the event signatures are already maintained as a graph that represents causality, finding the root cause is essentially a network centrality check that maximizes causality, and we use a trivial derivative of the PageRank algorithm to look up the root cause.

We invented a directed clustering algorithm to form a directed causal event graph from the causal scores measured from the events and their features. After that, we employed a custom version of Eigenvector centrality measures to achieve Probable RCA scores over the causal event graph. Each cluster can be visually shown as below with edge scores showing causal scores and node scores showing RCA scores (Fig. 3).

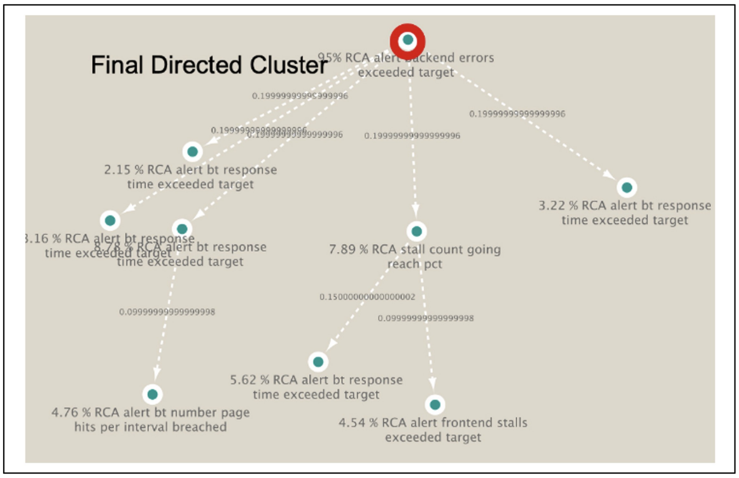
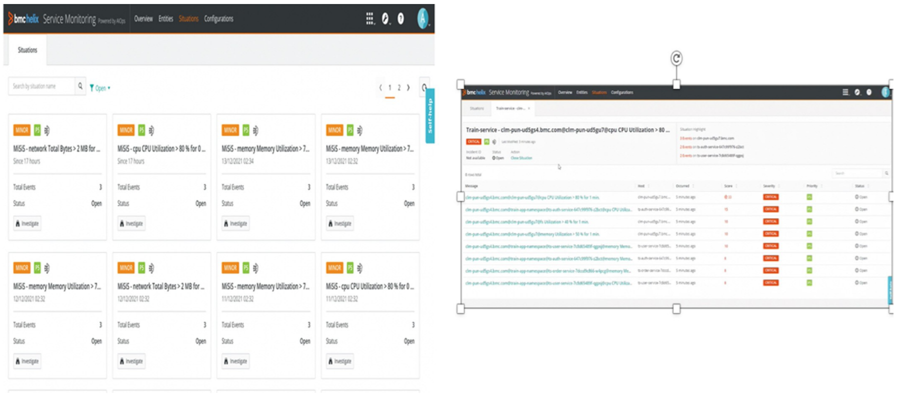

Final Directed Cluster
95% RCA alert backend errors exceeded target
0.19999999999999996
0.19999999999999996
0.19999999999999996
0.19999999999999996
0.19999999999999996
2.15 % RCA alert bt response time exceeded target
3.22 % RCA alert bt response time exceeded target
3.16 % RCA alert bt response time exceeded target
8.78 % RCA alert bt response time exceeded target
7.89 % RCA stall count going reach pct
0.15000000000000002
0.09999999999999998
0.09999999999999998
5.62 % RCA alert bt response time exceeded target
4.76 % RCA alert bt number page hits per interval breached
4.54 % RCA alert frontend stalls exceeded target

**Fig. 3.** Directed clustering algorithm.

Each of these Situations is presented in the user interface (UI) as shown below (Fig. 4).



| | | | |
|---|---|---|---|
| MINOR P5<br>MiSiS - network Total Bytes > 2 MB for ...<br>Since 17 hours<br>Total Events 3<br>Status Open | MINOR P5<br>MiSiS - cpu CPU Utilization > 80 % for 0 ...<br>Since 17 hours<br>Total Events 3<br>Status Open | MINOR P5<br>MiSiS - memory Memory Utilization > 7...<br>13/12/2021 02:34<br>Total Events 3<br>Status Open | MINOR P5<br>MiSiS - memory Memory Utilization > 7...<br>13/12/2021 02:32<br>Total Events 3<br>Status Open |
| MINOR P5<br>MiSiS - memory Memory Utilization > 7...<br>12/12/2021 02:32<br>Total Events 3<br>Status Open | MINOR P5<br>MiSiS - network Total Bytes > 2 MB for ...<br>12/12/2021 02:32<br>Total Events 3<br>Status Open | MINOR P5<br>MiSiS - memory Memory Utilization > 7...<br>11/12/2021 02:32<br>Total Events 3<br>Status Open | MINOR P5<br>MiSiS - cpu CPU Utilization > 80 % for 0 ...<br>11/12/2021 02:32<br>Total Events 3<br>Status Open |

Train-service - clm-pun-ud5gs4.bmc.com@clm-pun-ud5gu7@cpu CPU Utilization > 80 ...

CRITICAL P5 Last Modified: 5 minutes ago

Incident ID: Not available · Status: Open · Action: Close Situation

Situation Highlight
3 Events on clm-pun-ud5gu7.bmc.com
2 Events on ts-auth-service-647c99f976-z2bct
2 Events on ts-user-service-7c8d65489f-qgpxj

8 rows total

| Message | Host | Occurred | Score | Severity | Priority | Status |
|---|---|---|---|---|---|---|
| clm-pun-ud5gs4.bmc.com@clm-pun-ud5gu7@cpu CPU Utilization > 80 % for 1 min. | clm-pun-ud5gu7.bmc... | 5 minutes ago | 33 | CRITICAL | P5 | Open |
| clm-pun-ud5gs4.bmc.com@train-app-namespace@ts-auth-service-647c99f976-z2bct@cpu CPU Utiliza... | ts-auth-service-647c99... | 5 minutes ago | 15 | CRITICAL | P5 | Open |
| clm-pun-ud5gs4.bmc.com@clm-pun-ud5gu7@fs Utilization > 40 % for 1 min. | clm-pun-ud5gu7.bmc... | 5 minutes ago | 10 | CRITICAL | P5 | Open |
| clm-pun-ud5gs4.bmc.com@clm-pun-ud5gu7@memory Utilization > 50 % for 1 min. | clm-pun-ud5gu7.bmc... | 5 minutes ago | 10 | CRITICAL | P5 | Open |
| clm-pun-ud5gs4.bmc.com@train-app-namespace@ts-user-service-7c8d65489f-qgpxj@memory Memo... | ts-user-service-7c8d65... | 5 minutes ago | 10 | CRITICAL | P5 | Open |
| clm-pun-ud5gs4.bmc.com@train-app-namespace@ts-auth-service-647c99f976-z2bct@memory Memo... | ts-auth-service-647c99... | 5 minutes ago | 8 | CRITICAL | P5 | Open |
| clm-pun-ud5gs4.bmc.com@train-app-namespace@ts-order-service-7dccd9c866-w4pcg@memory Me... | ts-order-service-7dccd... | 5 minutes ago | 8 | CRITICAL | P5 | Open |
| clm-pun-ud5gs4.bmc.com@train-app-namespace@ts-user-service-7c8d65489f-qgpxj@cpu CPU Utiliza... | ts-user-service-7c8d65... | 5 minutes ago | 8 | CRITICAL | P5 | Open |

**Fig. 4.** Situations in the BMC AIOPs UI.

When users investigate each Situation, they see all the symptomatic events and probable root causes grouped together in the sorting order of a RCA score, which signifies the probability of that event being the root cause candidate.

Our solutions for service-centric noise reduction and for incremental differential clustering for causality were presented to users in a series of research studies conducted between March and December 2021. Probable Cause Analysis was very well received and evaluated as a “major saver”. While the term “Situations” was not familiar or intuitive to the users, the presented solution for having the BMC AIOps software automatically group and correlate events was assessed as “incredibly helpful” and “a very useful point of view”.

Towards the end of 2021, the BMC AIOps solution was initially rolled out to the first set of customers. Our next steps are to test and evaluate BMC AIOps in customers’ environments, and to continue developing and perfecting it through continuous user research engagement.

## Reference

1. Shi, N., Liu, X., Guan, Y.: An improved k-means clustering algorithm. In: 2010 Third International Symposium on Intelligent Information Technology and Security Informatics. 9th International Proceedings on Proceedings. IEEE, Jian (2010)