StackState

Monitoring Maturity Model

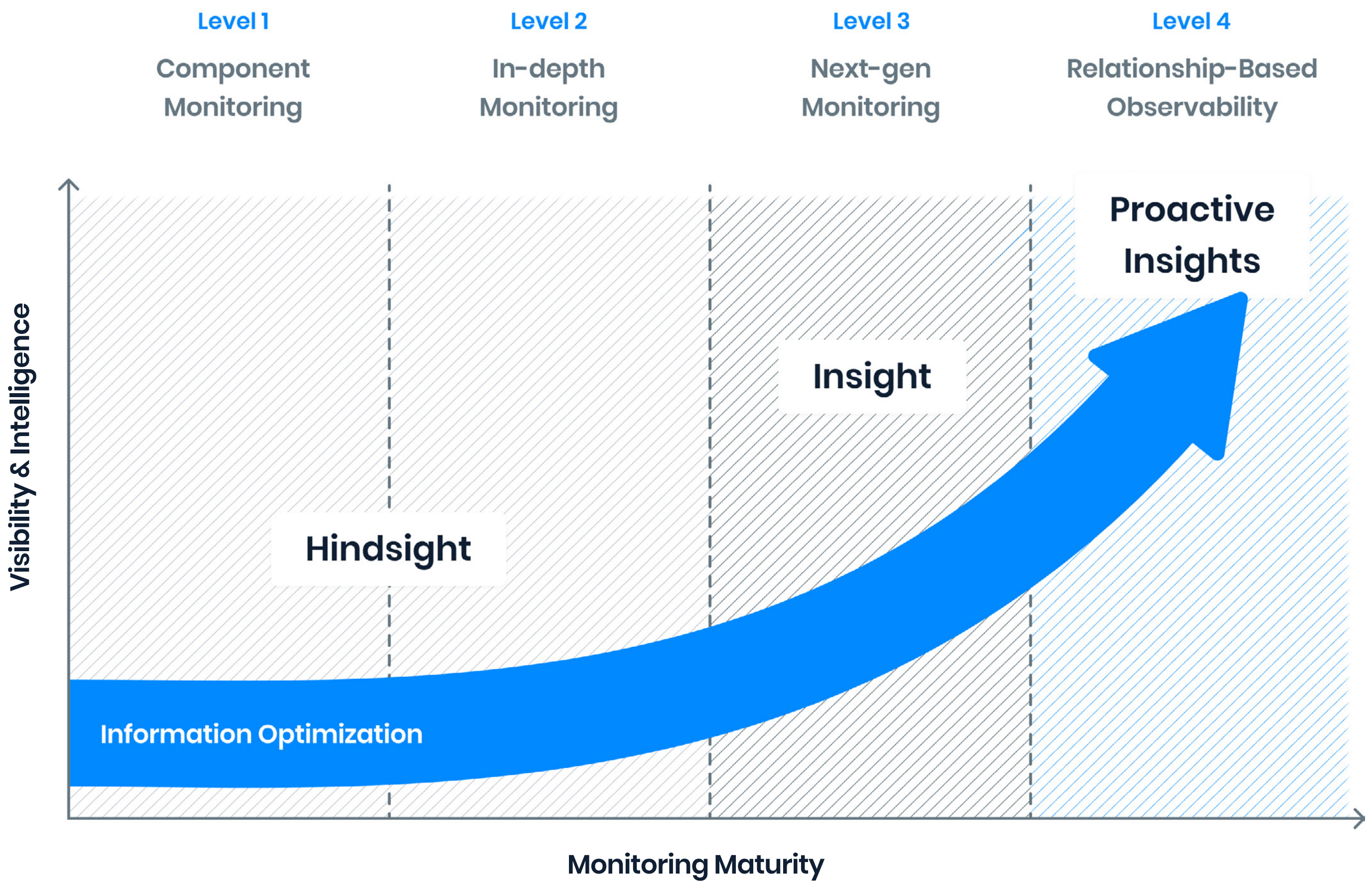
For IT Operations - 2020

Table of contents

Why the Monitoring Maturity Model?

Monitoring has been around for decades as a way for IT operations to gain insight into the availability and performance of its systems. To meet the growing market demands and drive more revenue, IT organizations require a deeper and more precise understanding of what is happening across their IT environments. This is not easy as today's infrastructure and applications span multiple environments and are more dynamic, distributed and modular in nature. According to a survey by Enterprise Management Associates (EMA), most enterprises find it difficult to find the right monitoring strategy to manage their environments. At the same time, over 65% of the enterprise organizations have more than 10 monitoring tools. These monitoring tools run as siloed solutions to support specific needs for different teams. This limits IT operation's ability to detect, diagnose and address performance issues. When issues occur, it is left to the IT teams to root-cause issues by combining teams, processes and tools or manually piecing together siloed data fragments. This traditional approach to monitoring is time-consuming and doesn't provide the insights that improve business outcomes.

Based on research and conversations with enterprises from various industries we created the Monitoring Maturity Model. This model reflects the 4 stages of monitoring and will help you to understand your current monitoring maturity level and what steps you need to take to improve it.



Level 1 - Individual Component Monitoring

If you are at the first level of the Monitoring Maturity Model, you are most likely running behind your industry peers. At this level, monitoring is largely done manually or not at all. IT teams will spend most of their time in understanding and filtering alerts and minimizing downtime.

At this level you are monitoring each individual component across your environment with a tool like Nagios or PRTG. Monitoring solutions at this level only report if a component is up or down. If an issue occurs that involves multiple components you will get an overload of alerts. It's up to you and your team to find out what happened and determine if it's relevant for further investigation. It's likely that most of the alerts will be irrelevant and can simply be ignored.

Monitoring the health of each component is the basis of monitoring. To gain a better understanding of the performance and availability of your systems, it's critical to move on to level 2.

Overview:

Summary:	Basic level monitoring to understand the health of systems
Challenge:	Understanding why systems are not performing optimally
Solution:	Instrumentation to get insights in metrics, logs and traces

Pros and Cons:

+ **Know the health status per component**
+ **Receive alert notifications when issues occur**

- **Only know if a component is healthy or not**
- **Risk of wasting time and resources by investigating irrelevant alerts**

Level 2 - In-depth monitoring

Most organizations we see are at level 2. At this level you are monitoring your systems from different angles to gain an overall perspective. You will need to collect 3 critical types of telemetry from the components or services you are trying to monitor: Metrics, Logs and Traces.

Metrics are measurements (such as availability, usage and traffic) that helps you to efficiently manage the business and performance of IT. A log file is the automatically produced and time-stamped documentation of events relevant to a system. Virtually all software applications and systems produce log files. And finally, traces give a detailed snapshot of a single transaction in your application. A trace records the available function-, database-, and external calls. You can use traces to troubleshoot performance issues and get code level visibility into how your app is working.

Today's IT organizations use many different tools for different purposes. And that's a good thing, because you'll need the different types of data to get a better understanding of what's going on across your systems. However, the use of different tools also comes with a price.

When everything goes haywire, multiple teams receive an overload of alerts and need to jump from tool to tool to root-cause issues. Technological and organizational boundaries further complicate the process of piecing together a complete picture of what's going on. War room meetings and finger pointing usually follow. Meanwhile, your customers are not happy and revenue is lost.

Overview:

Summary:	Different monitoring tools in place to get insights about incidents from different angles
Challenge:	Pin-pointing problems fast while data is fragmented and spread across teams and tools
Solution:	Consolidation of data

Pros and Cons:

+ **Monitoring from multiple perspectives creates more in-depth insight**
+ **Great collection of data available to root-cause issues**

- **Not aware of dependencies across systems and teams**
- **Manually root-cause issues by piecing together siloed data fragments**

Level 3 - Next-Gen Monitoring

Various organizations are migrating workloads to the cloud and are implementing continuous delivery to speed up software development. These initiatives can really put your IT teams to the test. Fast, short release cycles and dynamic cloud environments make it hard to keep track of your application and infrastructure landscape. A small change deep down in the infrastructure may seem innocuous, but can have a large impact over time. IT teams need to understand how their infrastructure and applications are interrelated and how changes can affect the business.

At monitoring level 3, you are not only tracking alerts, metrics, logs and traces, but you also take changes and dependencies into account. This data comes from tools that know a specific part of the environment like discovery-, deployment- or provisioning tools, cloud providers or container management tools such as Kubernetes and Docker.

Consolidating all of the data in a single place will give you a precise picture of your entire IT landscape and how it evolves over time. At this level, you'll have full stack visibility into all the components that make up your different services:

- **From business processes to code level details**
- **From on-prem to cloud**
- **And from legacy systems to microservice technologies.**

If your monitoring maturity is at level 3, you'll instantly see the cause and impact of any change or failure across the silos. Seeing what, when and why something happened allows you to accelerate your mean-time-to-repair (MTTR), quickly solve costly incidents and avoid war room meetings. If you find yourself operating at monitoring level 3, you are on the right track and ahead of most of your peers.

Overview:

Summary: Consolidated insights to reduce your MTTD and MTTR

Challenge: Still a reactive approach towards solving and preventing IT issues

Solution: Applying Artificial Intelligence for predictive insights

Pros and Cons:

+ **Full stack visibility enables a more proactive monitoring approach**

+ **Break down silos and root-cause issues faster to cut down your MTTR**

- **Lack the ability to predict issues before they occur**

Level 4 - Relationship-Based Observability

The final level of the monitoring maturity model is all about applying Relationship-Based Observability to your operations. In recent years, the concept of Observability has risen to address the persistent risk to a company's digital experiences and mission critical business applications as IT environments continue to become more complex and dynamic. The traditional focus of Observability is on metrics, logs, and traces. But unlike traditional observability solutions, Relationship-Based Observability breaks new ground at monitoring maturity level 4, allowing you to add the following unique capabilities to your current platform or solution:

- **Relationship change analysis** between transactions, microservices, service meshes, applications, containers, Kubernetes Pods, Kubernetes Nodes, virtualization and cloud platforms, and all of the resources (compute, memory, networking, and storage) that support each application in real-time over time. This means knowing exactly what these relationships are and how they changed before, during and after each incident.

- **Changes in configuration state** of the application and its entire supporting infrastructure in real-time over time. This means know exactly how the configuration state changed before, during, and after each incident.

- **Root cause based upon AI AND deterministic relationships.** Relationship-Based Observability means that the AI knows with certainty that a set of objects are related to each other and that they all support and affect the transaction of interest. This avoids the false alarms (false positives) that result from the AI confusing correlation with causation.

Overview:

Summary:	Know what changed and who should be concerned with Relationship-Based Observability
Challenge:	Building a progressively inclusive experience versus trying to "boil the ocean"
Solution:	Relationship-based Observability

Pros and Cons:

+ **Decrease MTTR by 80% by identifying root cause and alerting the right teams with the right information**
+ **Reduce the number of outages by 65% through real-time unified observability and more planful planning**
+ **Increase application releases by 3X by giving time back to developers**

- **You need to implement a plan to tie disparate systems together that is planful and progressive. But doing so will deliver the insight + information you need to be proactive about identifying and fixing issues**

The Input-Output Diagram

Level	Component Monitoring	In-depth Monitoring	Next-gen Monitoring	Relationship-Based Observability
System Input	Events	Events	Events	Events
		Metrics, Logs, Traces	Metrics, Logs, Traces	Metrics, Logs, Traces
			Changes & Dependencies	Changes & Dependencies
				Configuration State Changes
				AI + Deterministic Root Cause
System Output	Alerts	Alerts	Alerts	Alerts
		Dashboards	Dashboards	Dashboards
			Full Stack Visibility	Full Stack Visibility
				Proactive + Actionable Insight

Summary

Monitoring is at the core of any digital enterprise. Knowing what changed and who should be concerned, and being able to take action in real-time is critical to meet business SLAs and the growing expectations of customers.

This Monitoring Maturity model provides a yardstick to measure your monitoring approach as well as a guide to make improvements. As you go from maturity level 1 to 2, you will gain more in-depth insights into the systems you're trying to monitor. This will result in a better understanding of the availability and performance of your services.

The step from level 2 to 3 will get you full stack visibility across IT silos and a precise understanding of dependencies between business processes, applications and infrastructure. No matter the cloud, application or infrastructure, you're able to embrace a more proactive monitoring approach that supports the needs of today's digital enterprises.

And finally, when you make the step to level 4, you will gain Relationship-Based Observability capabilities that will help your business to maximize the investment in monitoring tools and accelerate the ROI of your observability program. The result? Smarter, faster and more effective IT Operations.

Ready to dive in?

See how Relationship-Based Observability can help your enterprise.

Schedule demo