# OBSERVABILITY VS MONITORING: WHAT'S THE DIFFERENCE?

To aid with our understanding of observability vs monitoring let's look at the evolution of the enterprise IT world. Enterprise IT, application, and business service development are increasingly complex. The interdependencies within the underlying architecture have become more fragmented, resulting in difficulty visualizing the full IT stack.

The internet delivers IT infrastructure services from hyperscale data centers at distant geographic locations. Companies are moving towards cloud-native delivery, resulting in modern distributed applications and creating a perfect storm of complexity with constantly emerging technologies, hybrid-cloud infrastructures, and businesses expecting delivery of more features faster.

Companies are consuming these services—like microservices and containers—as distributed functions across layers of infrastructure and platform services. Consumers expect regular, continuous feature improvements through new releases.

To meet these requirements, IT service providers and enterprises must aggressively manage business service performance, improve stability, and predict and prevent performance degradation and outages—all in the context of the rapidly changing and evolving IT landscape. This requires closely observing and monitoring metrics and datasets related to service performance to optimize system availability, particularly during upgrades and code launches.

Observability seems like the hot new topic in the IT world, but the reality is it has been with us for a long time. Only recently, however, has it entered the IT realm, combining with monitoring to offer a more powerful approach to business service performance management. System observability and

monitoring play critical roles in achieving system dependability—they may be interdependent, but they're not the same thing. Let's understand the differences between monitoring and observability, and how they are both critical for enhanced, end-to-end visibility.

## What is monitoring?

In enterprise IT, monitoring is the process of instrumenting specific components of infrastructure and applications to collect data—usually metrics, events, logs, and traces—and interpreting that data against thresholds, known patterns, and error conditions to turn the data into meaningful and actionable insights.

Monitoring is focused on the external behavior of a system, specifically those data targeted for collection. Monitoring is most effective in relatively stable environments, where key performance data and normal versus abnormal behavior is known. When enterprise IT was predominantly run in an organization's own data center, monitoring was an appropriate way to manage the environment.

The introduction of public and private clouds, the adoption of DevOps; the emergence of new technologies; the massive scale of data brought on by digital transformation; and the proliferation of mobile devices and IoT have created a situation where monitoring is no longer an effective approach for IT operations (ITOps).

Monitoring in IT systems focuses on overseeing the operation of servers, applications, and infrastructure to ensure they perform within expected parameters. It involves continuous surveillance of system metrics like network traffic, CPU utilization, and memory usage to detect performance anomalies or failures in real time. Effective monitoring provides alerts when metrics fall outside normal ranges, enabling IT teams to swiftly address and mitigate issues. This real-time crisis management is crucial for maintaining high availability and performance in enterprise environments.

## Describing observability

The concept of observability was introduced by R. E. Kalman in 1960 in the context of control systems theory. In control systems theory, observability is a measure of how well internal states of a system can be inferred from knowledge of its external outputs. In essence, it's a method for learning about what you don't know from what you do know. The relationship between the known and the unknown can be represented mathematically.

So, given the known state, external data, and enough time to do the mathematical calculations, the internal, unknown state of the system can be determined. This approach is well suited for modern, enterprise IT, as distributed infrastructure components operate through multiple abstraction layers. This makes it impractical and challenging to understand the health of complex services by selecting specific components to instrument for telemetry and looking for threshold breaches, events, etc.

The challenge to implementing observability in IT has been the volume, variety, and velocity of external data, combined with having the computational power and domain knowledge needed to analyze and make sense of it in real time. Effective ITOps teams now need observability platforms that can consume vast quantities of data from a variety of sources and submit that data to immediate intensive computational analysis. Fortunately, such platforms, like BMC Helix Operations Management, are now available.

Expanding on this, observability in IT extends beyond the basic functionalities of monitoring by

providing deeper insights into the behavior of systems through comprehensive data analysis. It utilizes telemetry data including logs, metrics, and traces, which helps developers and operators not just see what is happening within the system, but also understand why it is happening. This depth of insight is crucial for troubleshooting complex systems, optimizing performance, and making informed decisions about future improvements in software architecture and deployment strategies.

---

**Learn more about observability ›**

---

## What is the difference between observability and monitoring?

The main difference between observability and monitoring is that monitoring is better suited for simpler systems where parameters are well-known and performance is well-understood, while observability is better suited for more complex, multi-system environments with more potential unknowns.

For simple systems, traditional monitoring is effective and can provide some measure of insight into a system's health. Consider a single server machine. It can be easily monitored using metrics and parameters such as hardware energy consumption, temperature, data transfer rates, and processing speed. These parameters are known to be highly correlated with the health of internal system components.

Now consider a large, complex business service. It is made up of multiple applications that span public and private clouds, a distributed infrastructure, and possibly a mainframe. There are too many systems, some not directly accessible, that if monitored without knowledge of the key performance data, systems, and error conditions, will generate too much uncontextualized data as well as unnecessary alerts, data, and false flags.

In the second case, an observability and AIOps approach is needed. Rather than selecting the data to monitor and examine the behavior of that data relative to trends, known errors, etc., all available data from all systems should be consumed. Aggregated into a high-performance data store, it should be combined with a comprehensive topology of all assets, systems, and applications that builds a comprehensive model of relationships and dependencies.

On this foundational observability layer, high-performance, domain-informed artificial intelligence and machine learning (AI/ML) algorithms can be applied to determine which externally observable data correlates with which services and infer the health of those services from their behavior. This is the power of an observability and AIOps approach, such as that used by BMC Helix Operations Management.

---

**Find out how BMC Helix Operations Management can help you tackle challenging hybrid cloud environments ›**

---

## How do observability and monitoring impact business outcomes?

Observability and monitoring are not just technical exercises; they have profound impacts on business outcomes. By ensuring systems operate smoothly and are quickly repairable when issues arise, these practices directly contribute to maintaining high levels of customer satisfaction. For example, real-time monitoring of e-commerce platforms can detect and address performance bottlenecks during peak traffic, preventing potential sales losses and preserving customer loyalty.

Furthermore, observability provides businesses with actionable insights that go beyond problem-solving. It enables predictive maintenance, which can anticipate issues before they occur, thus minimizing downtime and its associated costs. This proactive approach saves money and enhances the overall customer experience by delivering consistent service quality.

Additionally, in environments where performance directly ties to revenue, like in financial trading platforms or real-time data services, observability's role becomes critical. The ability to continuously analyze and interpret complex data streams ensures that performance stays within the threshold necessary for optimal operation, directly impacting profitability.

## How have tech advancements shaped observability and monitoring?

The landscape of enterprise IT has undergone significant transformation, driven by advancements in technology that have reshaped observability and monitoring. The proliferation of cloud computing, for example, has expanded the complexity and scale of IT environments, making traditional monitoring tools insufficient for new dynamic infrastructures. Similarly, the rise of big data has introduced challenges in managing voluminous data flows as well as opportunities for deeper insights through observability tools.

These technological shifts have necessitated the development of more sophisticated tools and practices. AI/ML are now integral to modern observability platforms, enabling predictive analytics and automated problem resolution that were not possible with earlier systems. This evolution from reactive monitoring to proactive observability reflects a broader trend towards more agile and resilient IT operations.

## Is observability used in DevOps?

The concept of observability is prominent in DevOps software development lifecycle (SDLC) methodologies. In earlier waterfall and agile frameworks, developers built new features and product lines while separate testing and operations teams tested for software dependability. This siloed approach meant that infrastructure operations and monitoring activities were beyond development's scope. Projects were developed for success and not for failure: debuggability of the code was rarely a primary consideration.

Infrastructure dependencies and application semantics were not adequately understood by the developers. Therefore, apps and services were built with low inherent dependability. Monitoring failed to yield sufficient information about the known-unknowns, let alone the unknown-unknowns, of distributed infrastructure systems.

The prevalence of DevOps has transformed SDLC. Monitoring goals are no longer limited to collecting and processing log data, metrics, and distributed event traces; monitoring is now used to make the system more observable. The scope of observability therefore encompasses the

development segment and is facilitated by people, processes, and technologies operating across the SDLC pipeline.

Collaboration among cross-functional Devs, ITOps, site reliability engineers (SREs), and quality assurance (QA) personnel is critical when designing a highly performant and resilient system. Communication and feedback between developers and operations teams is necessary to achieve observability targets of the system that will help QA yield correct and insightful monitoring during the testing phase. As a result, DevOps teams can test systems and solutions for true real-world performance. Continuous iteration based on performance feedback can further enhance the ability to identify potential issues in the systems before the impact reaches end users.

Observability offers actionable intelligence for optimizing performance, giving DevOps, SREs, and ITOps increased agility by staying ahead of any potential service degradation or outages. Observability is not only limited to technologies; it also covers the approach, organizational culture, and priorities in reaching appropriate observability targets, and hence, value-of-monitoring initiatives.

## How does observability function in real-world scenarios?

Observability isn't just theoretical; it has practical implications in real-world IT scenarios. For instance, consider a global e-commerce company that experiences sporadic system outages during high-traffic periods like Black Friday sales. By implementing an observability platform, the company can aggregate data across all system components—not just those predetermined as critical. This holistic data collection enables them to use advanced analytics to predict potential points of failure before they result in downtime, thereby not just reacting to system anomalies but also proactively managing system health.

Tools like dynamic baselining, anomaly detection, and automated root cause analysis play crucial roles in these observability platforms, supporting modern ITOps across complex and hybrid environments.

The evolution from traditional monitoring to sophisticated observability marks a significant shift towards proactive management of IT systems. By leveraging advanced tools like dynamic baselining, anomaly detection, and automated root cause analysis, businesses can both react to issues and anticipate them, ensuring higher system reliability and improved performance. Embracing these practices is essential for any enterprise aiming to thrive in the digital age, where agility and resilience are key.