AMERICAN IMPACT REVIEW
A PEER-REVIEWED MULTIDISCIPLINARY JOURNAL

High-Load Systems: Architectural Patterns for Scalable Web Applications

— Manuscript Draft —

Manuscript Number	e2026001
Article Type	Original Research
Received	February 9, 2026
Subject Area	Information Technology
Keywords	high-load systems, scalability, microservices, distributed architecture, load balancing, caching, event-driven architecture
Authors	Dmitry Volkov, Alexei Petrov, Ivan Sorokin
Abstract	As digital platforms scale to serve millions of concurrent users, the architectural decisions made during system design become critical determinants of performance, reliability, and maintainability. This paper presents a comprehensive analysis of architectural patterns employed in high-load web applications, examining how organizations navigate the complex trade-offs between consistency, availability, and partition tolerance.

REVIEW ASSIGNMENT

Reviewer Dalas Po

Deadline February 28, 2026

Please evaluate this manuscript for originality, methodological rigor, clarity of presentation, and significance of findings. Submit your review and recommendation to the Editor-in-Chief at **egor@americanimpactreview.com**.

This document is confidential. Do not distribute, cite, or upload to any AI tools.

Abstract

High-load systems require monitoring to ensure user satisfaction with service quality and service level in real time. The purpose of this article is to analyze the operational response to peak-load situations in a high-load system in real time, as well as its scalability. Methods of systems analysis and an evidence-based approach to metrics accounting for user requirements are employed. Results include: an analysis of the capabilities of proactive and predictive monitoring; proposed procedures for anomaly detection and ranking of factors by consumer importance; and a mathematical model of system throughput with a test example. Practical implications of the results are indicated.

Keywords: monitoring, scalability, evidence-based approach, high-load systems

1. Introduction

High-load systems, or HLS (High Load Systems), ensure the stable operation of various services, fog and cloud platforms, particularly in the processing of financial and digital transactions with heightened requirements for fault tolerance and performance. Monitoring is essential and must ensure compliance with SLA (Service Level Agreement, i.e., user satisfaction with service) in real time, predictively.

The goal of monitoring is to correct the system's capacity based on monitoring data and to provide an operational response to real-world situations, especially in critical cases. Stability parameters must not exceed the established tolerance band.

In a distributed HLS, uncertainties in monitoring output data generate noise during processing. Therefore, system profiles and patterns should be formed automatically. These profiles must account for remote administration, auto-configuration, and related functions.

It is important to analyze and scale a high-load system, considering that the "magnitude" of the load depends on the infrastructure (the HLS ecosystem) and its resilience under peak loads. Most other factors are subordinate to resilience [1].

Ecosystem scalability can be considered in terms of both evolutionary scalability (generation scalability) and heterogeneous scalability, involving interactions with other systems and architectures (heterogeneous scalability).

In this article, we focus on scalability, monitoring, and the evidence-based approach to the assessment of HLS, as well as their development with respect to user experience requirements and SLA agreements.

2. Critical Metrics

The following are classified as critical metrics:

1. **Availability** (uptime of the HLS, or Uptime), which should be at least 99.9% (approximately 44 seconds of downtime per month). There is evidence of achieving a level of 99.99999% (26 seconds of downtime).
2. **Response latency** (delay, or Latency).
3. **Error rate** (proportion of failed requests, or Error Rate).
4. **Throughput** (the volume of data processed by the HLS in peak mode per unit of time, or Throughput).
5. **Resource utilization during monitoring** - processor (CPU utilization), random-access memory (RAM usage), disk (Disk I/O), and network (Network traffic).
6. **End-user satisfaction** with service speed (APDEX, Application Performance Index), response time (Response Time), and connection stability (Packet Loss).

3. Architecture of an Effective Monitoring System

An effective HLS monitoring system consists of the following layers:

1. **Infrastructure layer** (servers, databases, network, virtualization).
2. **Application layer** (services/microservices).
3. **Business process layer** (conversions, activities, orders, payments).

4. Proactive and Predictive Monitoring

Virtualization, the microservices approach, and SLA monitoring require proactive monitoring to identify problems "before" rather than "after." This is realized through:

- **Predictive analytics** (analysis of the history of failures, overloads, and causes of load growth);

- **Anomaly detection** (identification of deviations from norms);
- **Analysis of current and potential trends**;
- **Pre-release of alerts**.

Proactive monitoring is implemented via predictive monitoring, latency-correlation analysis, load testing and reverse testing, and failure probability estimation. Predictive monitoring is anticipatory and forecasting in nature, whereas proactive monitoring responds operationally to deviations and functions in conjunction with a load balancer. The aim is to apply both types of monitoring in an integrated manner, setting monitoring and SLA support goals operationally by:

1. Processing incidents;
2. Responding to events;
3. Recovering from failures;
4. Automating alerts and preventing "alert storms," among other measures.

Example. The Monq platform applies proactive and reactive monitoring in a comprehensive, scalable, and flexible manner (through customizable alert settings), analyzing metrics, detecting deviations from SLA, and forecasting potential failures. It supports hybrid data collection, horizontal scaling, and intelligent analysis of deviations and patterns. Monq can consolidate data from various monitoring tools (Zabbix, Prometheus) within a single system, enabling the tracking of the state of millions of objects in the HLS ecosystem.

5. Anomaly Detection

Systems such as Monq facilitate the detection of anomalies. This is not merely the rejection of gross data [2]; anomalies cannot be deleted due to their informational value.

We propose a procedure for anomaly detection based on the Irwin criterion [3]:

> *The Irwin criterion statistic is computed as the absolute difference between consecutive observations divided by the standard deviation of the sample: lambda_i = |x_i - x_{i-1}| / S, where S is the root-mean-square deviation.*

where S is the root-mean-square (standard) deviation:

*S = sqrt((1/(n-1)) * sum_{i=1}^{n} (x_i - x_mean)^2), the sample standard deviation.*

If the anomaly is strongly pronounced, the comparison may be performed with the nearest non-anomalous data point.

6. Evidence-Based Approach

The evidence-based approach to ensuring HLS resilience and scalability is grounded in sufficient criteria of effectiveness (primarily of the class "practical realizability"). It requires a clear understanding of objectives, approaches, and justification of effectiveness in the current situation.

Despite the progress of automation, it is impossible to fully automate the monitoring process using intelligent algorithms, owing to the weak formalizability of situations and the presence of anti-patterns [4]. Moreover, evidence-based reasoning is poorly taught in higher education institutions [5].

In the context of scaling and improving SLA, the evidence-based approach is highly relevant, particularly in connection with automation and the necessity of relying on principles such as "See and Do," "Try, Fail, and Fix," "Test the Framework," and others, as well as on the combination of microservices with scalability (transfer of solutions and approaches), adaptability (flexibility) and on-the-fly updating, and SLA configuration of the platform and service.

The required level of evidential rigor should be identified either heuristically or mathematically. There are three approaches:

1. Based on monitoring and experiments;
2. Based on verification and validation;
3. Based on formal (axiomatic) proof systems and others.

Within these three groups, subgroups can also be distinguished.

7. Matrix Assessment of SLA Capabilities

Monitoring enables the construction of a matrix assessment of SLA capabilities. We propose the following procedure:

1. Scaling the influence of SLA factors;
2. Identifying the probabilities of SLA capabilities;
3. Analyzing the digital profiles of the capability matrix;
4. Expert-heuristic assessment of factors and their importance to SLA;
5. Integral assessment of SLA importance (by factors);
6. Ranking factors by importance.

Frequently, monitoring data contain gaps, which are filled relevantly using statistical methods for imputation [6].

Evidence-based reasoning, when combined with a functional approach such as exploratory monitoring, requires the relevance of the assessed factors.

8. Development Trends in HLS Monitoring

HLS monitoring is evolving, in particular toward systems of the following classes:

1. **AI-driven** (machine-based prediction of failures, anomaly detection, and auto-configuration);
2. **Self-healing** (self-recovery, automatic failure remediation);
3. **Observability** (patterns for analyzing correlations among metrics and logs).

The transition to soft-skill models for SLA analysis is a necessity; therefore, intelligent services are engaged in the intelligent monitoring of HLS. Distributed monitoring support is organized according to the scheme: "user - network (platform) - system - monitoring - decision-maker (analyst)." Flexible approaches are employed, including ontological, neural network-based, fuzzy, and situational monitoring plans.

9. Mathematical Model of System Throughput

If modeling the probability of HLS failure, one should start from the failure probability of an individual service element (instance). For example, for N instances with failure probabilities

p_i, and taking into account the exponential decline in the probability of failure across all copies of a service, one can determine:

$$P_{fail} = \prod_{i=1}^{N} p_i^{n_i}$$, where n_i is the number of instances in the i-th copy of the service. This represents the compound failure probability under replication.

where n_i is the number of instances in the i-th copy.

A microservice system should be subjected to load testing with tracking of performance scalability [7]. Let us consider a model example.

For modeling the throughput $u(t)$ of concurrent users, a logistic model is proposed:

$$T(u) = T_0 + (T_{max} - T_0) / (1 + \exp(-k(u - u_c)))$$, a logistic growth function where T_0 is the baseline response time, T_{max} is the maximum response time under saturation, k is the steepness coefficient, and u_c is the critical (inflection) point of user count.

where the parameters are defined accordingly.

Example. For user counts u = 100, 200, 400, 800, and 1000, the average response time $T(u)$ reaches 30, 50, 80, 900, and 1200 milliseconds, respectively (Figure 1).

Figure 1. Dependence of the average response time $T(u)$ on the number of users u.

Using this model, simulation-based load testing can be performed. We present the following testing scenarios:

Scenario 1. The HLS contains N instances without redundancy.

Scenario 2. The HLS has a queue of L instances for redistribution among the remaining $N - L$ instances.

Scenario 3. The HLS specifies a distribution of service unavailability time following a failure.

Scenario 4. The HLS specifies a distribution of service unavailability time with redundancy.

It is important to conduct modeling at the orchestration stage and during load balancer configuration. This enables relevant scaling of the HLS and assessment of the time and point

of "saturation," as well as the costs of improving the performance and efficiency of the microservice architecture.

To reduce uncertainties during monitoring, one relies on patterns and anti-patterns of HLS controllability. Analytical support and evidential rigor enhance the capabilities of decision-makers, the HLS ecosystem, and more fully satisfy growing consumer expectations and SLA requirements.

10. Conclusion

This work demonstrates that ensuring SLA compliance and the resilience of high-load systems requires a combination of proactive and predictive monitoring, oriented toward anomaly detection, trend analysis, and overload prevention. Reactive approaches in distributed HLS prove insufficient under peak loads.

We have substantiated the applicability of the evidence-based approach, which enables decision-making regarding scaling and SLA configuration on the basis of monitoring data, experimental assessments, and throughput models. The proposed procedures for factor ranking and load testing are applicable at the orchestration stage and during load balancer configuration.

It has also been shown that full automation of monitoring is constrained by difficulties in process formalization and by persistent inefficient practices, which necessitate analytical support and hybrid analytical methods. The results obtained can be utilized in the design, operation, and scaling of high-load platforms and microservice architectures.

References

1. Smirnov, N. A., Chervyakov, L. M., & Bychkova, N. A. (2025). Povyshenie effektivnosti poiskovykh zaprosov vysokonagruzhennykh prilozhenii [Improving the efficiency of search queries in high-load applications]. *Izvestiya Tul'skogo gosudarstvennogo universiteta. Tekhnicheskie nauki*, 2025(2), 152-158.

1. L'vovskii, E. N. (1988). *Statisticheskie metody postroeniya empiricheskikh formul* [Statistical methods for constructing empirical formulas] (2nd ed.). Moscow: Vysshaya shkola. 238 p.

1. Zalyazhnykh, V. V. (2020). Rasshirenie oblasti primeneniya kriteriya Irvina pri obnaruzhenii anomal'nykh izmerenii [Extending the application domain of the Irwin criterion in anomalous measurement detection]. *Vestnik SibGUTI*, 2020(2), 95-100.

1. Ziborev, A. V. (2023). Antipattreny postroeniya mikroservisnykh prilozhenii v vysokonagruzhennykh proektakh [Anti-patterns in the construction of microservice applications in high-load projects]. *Universum (tekhnicheskie nauki): elektronnyi nauchnyi zhurnal*, 2023(11(116)), 29-34. Retrieved from https://7universum.com/ru/tech/archive/item/16204 (accessed September 24, 2025).

1. Semenov, A. L., Fiofanova, O. A., Babchenko, O. I., et al. (2021). Izvlech' smysl. Problemy analiza dannykh v obrazovanii [Extracting meaning: Problems of data analysis in education]. *Obrazovatel'naya politika*, 2021(3(87)), 60-65.

1. Popkov, Y. S., Dubnov, Y. A., & Popkov, A. Y. (2016). New method of randomized forecasting using entropy-robust estimation: Application to the world population prediction. *Mathematics*, 4(1), 1-16.

1. Gashimov, R. E. (2025). Proektirovanie masshtabiruemykh raspredelennykh mikroservisnykh backend-sistem dlya vysokonagruzhennykh sred [Designing scalable distributed microservice backend systems for high-load environments]. *Aktual'nye issledovaniya*, 2025(17(262)), Part 1, 9-16.