Behavior-Driven Performance Testing Via Integrated MLOps Pipeline

PESARO 2026, **May 24 - May 28, - Venice, Italy**

Bharath Kumar Maganti

Southern Arkansas University · Austin, Texas | kumar.mbk2110@gmail.com

75%

Effort Reduction

92%

Peak Alignment

10×

Faster Design

Bharath Kumar Maganti

Architect – Performance Engineering, AIOps

Professional Experience

- Architect – AIOps, CSAA Insurance Group
- Lead Architect – Performance Engineering & SRE, UCOP, CA

Publications

- Energy-Proportional Load Balancing for Sustainable AI Workload Performance in Cloud Infrastructure, IEEE Greentech 2026
- PerfMLOps: Zero-Human in the Loop Performance Testing Framework, IEEE 30th International Conference on IT, 2026
- Performance Test Engineering Practice for Scaled Agile Framework Leveraging Machine Learning and Artificial Intelligence Techniques, IJCTT 2023

Agenda

01 **Problem Statement**

Limitations of manual performance testing in fast-release cycles

02 **Pipeline Architecture**

Five-phase end-to-end MLOps-integrated testing framework

03 **Implementation**

AWS SageMaker, XGBoost, New Relic, JMeter, GitOps

04 **Results & Comparison**

Efficiency gains, accuracy metrics, vs. traditional approach

05 **Implications & Future Work**

Continuous performance engineering and next steps

Problem Statement

Time-Intensive

~20 hours per cycle for manual log analysis, peak identification, and scripting

Prone to Error

Human oversight misses nuanced patterns like dynamic surges and seasonal spikes

Doesn't Scale

Lag behind agile/CI cycles; manual redesign required for every production drift

Low Fidelity

60–75% alignment with production behavior due to estimation-based workloads

The Opportunity

→ Observability platforms stream real-time production metrics

→ MLOps enables automated model training & deployment

→ GitOps supports versioned, reproducible configurations

→ CI/CD can trigger tests automatically

Pipeline Architecture — Five Phases

1	2	3	4	5
Telemetry Ingestion	**ML Forecasting**	**Workload Synthesis**	**GitOps & CI/CD**	**Reporting & Validation**
CPU, memory, request rates, latencies, errors streamed from New Relic / Splunk into S3 or Kafka	XGBoost ensemble on SageMaker predicts peak hours & days from temporal + app-specific features	Predictions translated to JMeter params (threads, ramp-up, think times, endpoint weights) as YAML/JSON	Configs committed to GitHub; GitHub Actions triggers containerized JMeter load tests automatically	p95 latency, throughput saturation auto-reported for engineer review against SLO benchmarks

Implementation Stack

AWS SageMaker

- Model training & hosting
- XGBoost ensemble
- Periodic retraining
- Anomaly detection

Observability

- New Relic metric export
- Splunk API integration
- Real-time streaming
- S3 / Kafka storage

Workload Gen

- Peak → JMeter params
- Thread count mapping
- YAML/JSON configs
- Ramp-up period calc

GitOps / CI/CD

- GitHub repository
- GitHub Actions triggers
- Docker-containerized JMeter
- Version-controlled configs

Experimental Results

Kubernetes-based e-commerce microservices — 3 days training, 3 days validation

1.2 hrs

Peak Hour MAE

vs. ~6 hr baseline

88%

Peak Day Accuracy

classification

<2 hrs

Workload Design

down from ~20 hrs

92%

Bottleneck Fidelity

similarity to prod

<8%

False Positive Rate

over-testing

75%+

Effort Saved

conservative estimate

Comparison: Traditional vs. MLOps Pipeline

Metric	Traditional Manual	ML-Integrated Pipeline	Gain
Design Effort	~20 hrs / cycle	<2 hrs (automated)	**75–90% reduction**
Peak Alignment	60–75%	92%	**+17–32% fidelity**
Bottleneck Fidelity	70–85%	92–98.5%	**+10–25% realism**
Reproducibility	Low (manual notes)	High (Git-versioned)	**Significant improvement**
Cycle Time	Days (manual redesign)	Hours (auto-retrain)	**Faster feedback loop**

Analysis & Implications

Higher Test Fidelity

Data-driven workloads capture nuanced patterns — dynamic surges, seasonal spikes — that engineers routinely overlook in manual analysis.

Reproducibility & Auditability

Git-versioned configs provide full audit trails. Tests can be replayed exactly, enabling regression benchmarking as models evolve.

Engineer Productivity

75% effort savings redirect engineering time toward higher-value tasks: anomaly interpretation, SLO refinement, architecture decisions.

MLOps Robustness

Anomaly detection and periodic retraining keep workload quality high despite shifting production traffic patterns.

Security & Governance

Telemetry masking, access controls, and human-in-the-loop review ensure accountability for business-critical performance decisions.

Future Work

Advanced Models

Transformer-based architectures for multivariate time-series to improve long-horizon telemetry forecasting

NLP on Logs

Extract user journeys from log data to enrich workload generation beyond simple peak metrics

Closed-Loop Learning

Test outcomes feed back into model retraining for a self-enhancing pipeline

Chaos Engineering

Inject failures during predicted peak periods to test resilience under realistic worst-case scenarios

Multi-Cloud Support

Extend observability ingestion to multi-cloud environments for broader applicability

Enterprise Validation

Longitudinal case studies to validate scalability, cost, and organizational adoption challenges

Conclusion

This research presents a practical MLOps-integrated, behavior-driven performance testing pipeline that continuously derives workloads from production behaviors — reducing human effort while enhancing test accuracy and reliability.

75%	92%	10×	High
Manual effort reduction	Peak behavior alignment	Faster workload design	Reproducibility via GitOps

This approach is a promising advancement for continuous, intelligent performance engineering in complex, dynamic software systems.

Bharath Kumar Maganti · Southern Arkansas University· kumar.mbk2110@gmail.com