International Journal of Science and Research Archive
eISSN: 2582-8185
Cross Ref DOI: 10.30574/ijsra
Journal homepage: https://ijsra.net/

(Review Article)

Revolutionizing mobile platform engineering: AI-driven event logging for enhanced performance and cost efficiency

Waseem Syed *

JNTU, Hyderabad, India.

International Journal of Science and Research Archive, 2025, 14(01), 1221-1231

Publication history: Received on 08 December 2024; revised on 16 January 2025; accepted on 19 January 2025

Article DOI: https://doi.org/10.30574/ijsra.2025.14.1.0152

Abstract

Mobile platform engineering faces unique challenges that impact performance and operational costs. This article explores the revolutionary potential of AI-driven event-logging systems in addressing these issues. By transitioning from traditional to AI-enhanced logging techniques, we significantly enhance performance through machine learning-based log prioritization, generative AI for root cause analysis, and efficient local event chain caching. This study provides a comparative analysis of conventional methods versus AI-driven systems, highlighting substantial improvements in error detection, system reliability, and cost efficiency. Real-world implementations and theoretical frameworks demonstrate how these advanced logging systems meet mobile-specific requirements such as protocol-agnostic logging, network state management, and battery optimization. The findings suggest that AI-driven logging not only transforms mobile platform engineering through enhanced operational performance but also provides scalable solutions that can adapt to evolving technological landscapes.

Keywords: Mobile Logging Systems; AI-Driven Event Logging; Protocol-Agnostic Logging; Resource Optimization; Error Detection Systems; Machine Learning Optimization; Event Chain Analysis; AI in System Reliability Enhancement; Intelligent Error Logging; AI-Driven Analytics

1. Introduction

Mobile platform engineering has a significant hurdle in the era of big data as traditional logging methods produce unmanageable amounts of data while increasing operating expenses to unaffordable levels. The complexity of the logging and analytics infrastructure has a significant impact on the costs of developing mobile applications. According to research [1], a systematic review of mobile application development cost estimation, the variety of logging and analytics requirements in mobile projects makes development cost estimation particularly challenging. The study found that logging and analytics infrastructure significantly impacts initial development efforts and ongoing maintenance costs as applications scale.

These challenges are especially severe in a mobile setting because of network unpredictability and device limitations. Comprehensive logging must be balanced with battery life, storage constraints, and fluctuating network circumstances in mobile apps. As applications manage complicated state transitions and depend more and more on different communication protocols, this balancing becomes even more important.

When looking at real-world mobile application behavior patterns, the scope of the logging problem becomes clearer. Research [2] thoroughly investigated the deployment of Firebase Analytics across mobile applications and discovered a notable influence on application performance. Their analysis of Firebase Analytics usage in Android applications revealed that 47% of apps used custom logging in addition to the usual events. The research identified that analytics

* Corresponding author: Waseem Syed.

implementations increased the size of applications by 3.5 MB to 7 MB. Additionally, the findings found that logging-related performance problems were present in 51% of the programs under study, with initialization delays being the most prevalent issue.

Intelligent log management systems driven by generative AI and machine learning are changing this environment. The promise of deep learning techniques in log management is demonstrated by research conducted on large-scale systems. [3]. According to the research, deep learning-based log analysis outperformed conventional rule-based methods by achieving an F1-score of 0.96 in anomaly detection tasks. According to the research, their DeepLog technology could efficiently process system logs and identify key system issues with a 95% accuracy rate while consuming little processing power.

Modern logging systems have an impact that goes beyond their technical capabilities. Based on the research [1], companies that used structured logging and analytics frameworks reported more accurate project estimates. When specific logging requirements were taken into account at the planning stage, estimation accuracy increased by 15% to 20%. According to the research, mobile applications with a well-designed logging architecture had lower debugging times and higher maintainability scores.

These log management enhancements are made possible by advanced anomaly detection and pattern recognition tools. Applications with advanced analytics setups demonstrated 23% higher crash reporting accuracy than those with basic implementations [2] documentation. Furthermore, a recent report [3] showed that the deep learning method could maintain real-time processing capabilities for large-scale log streams while reducing false positive rates in log analysis by 85% when compared to conventional methods.

2. The Evolution of Event Logging

The specific challenges in the mobile context have fueled the development of mobile logging systems. As applications evolved from straightforward standalone programs to complex distributed systems, logging requirements changed to address resource limitations unique to mobile devices, various communication protocols, and changes in network status.

Logging systems have undergone a significant transformation in the last ten years due to the rise in user expectations and the complexity of mobile applications. According to the research [4], mobile applications have evolved into three different generations: standalone applications from 2008 to 2011, web services-integrated applications from 2012 to 2015, and collaborative applications from 2016 onward. The analysis of Android apps showed that, in contrast to first-generation apps, 67% of contemporary apps needed sophisticated logging techniques to manage several concurrent activities and user interactions.

AI-driven solutions emerged as a result of the growing difficulties with conventional logging systems. The study [5] examined various machine learning applications in log analysis and highlighted the use of different learning approaches including supervised learning, unsupervised learning, and deep learning techniques. The research discussed how machine learning algorithms such as Random Forest and Support Vector Machines (SVM) are being applied to log analysis tasks, including anomaly detection and log categorization. The paper emphasized the growing importance of automated log analysis techniques in handling large-scale system logs effectively [5].

The way apps manage data gathering and analysis is one area where the changes in logging techniques are most noticeable. Third-generation mobile applications integrated an average of 8.3 distinct logging components, compared to 2.1 in first-generation applications [4]. Additionally, their research revealed that 73% of contemporary apps used automated log analysis in some capacity, indicating the growing complexity of mobile software systems and the demand for increasingly advanced debugging techniques.

Organizations' approaches to system monitoring and debugging have changed significantly as a result of the incorporation of machine learning into log management. Preprocessing methods increased log analysis accuracy by 34% across all machine learning algorithms [5]. According to the survey, traditional pattern-matching algorithms averaged 76% accuracy in sequence-based log analysis, but LSTM networks attained a 91% accuracy rate. Additionally, compared to single-method implementations, hybrid approaches that combined multiple machine learning techniques demonstrated a 23% improvement in false positive reduction.

3. Key Components of AI-Enhanced Logging Systems

3.1. Local Event Chain Caching

Local event chain caching represents a significant advancement in mobile application diagnostics. Research examining application-level caching strategies through analysis of mobile applications and developer interviews revealed three primary caching patterns: size-based, time-based, and hybrid approaches. Studies demonstrate that applications implementing structured caching strategies show notable improvements in data retrieval time compared to those using ad-hoc caching approaches [6].

3.2. Generative AI for Root Cause Analysis

The integration of advanced pattern recognition capabilities has transformed diagnostic approaches in modern logging systems. Research exploring log processing techniques across distributed systems demonstrates significant improvements in anomaly detection capabilities. Studies show that ML-based frameworks achieve higher precision and recall rates in identifying system anomalies compared to traditional rule-based systems [7].

3.3. Machine Learning-Based Log Prioritization

The application of machine learning to log management has fundamentally altered how systems handle event prioritization and storage optimization. Studies reveal that applications implementing prioritized caching strategies demonstrate improved memory utilization compared to those without such policies. Research particularly notes that e-commerce applications benefit significantly from these improvements due to their high volume of cached data [6].

The effectiveness of ML-based log analysis frameworks has been demonstrated in production environments. Experimental evaluations using large datasets show promising results in anomaly detection and system error identification. Studies indicate that these approaches significantly reduce processing time for log analysis compared to traditional pattern-matching methods while maintaining high accuracy in detecting critical system issues [7].

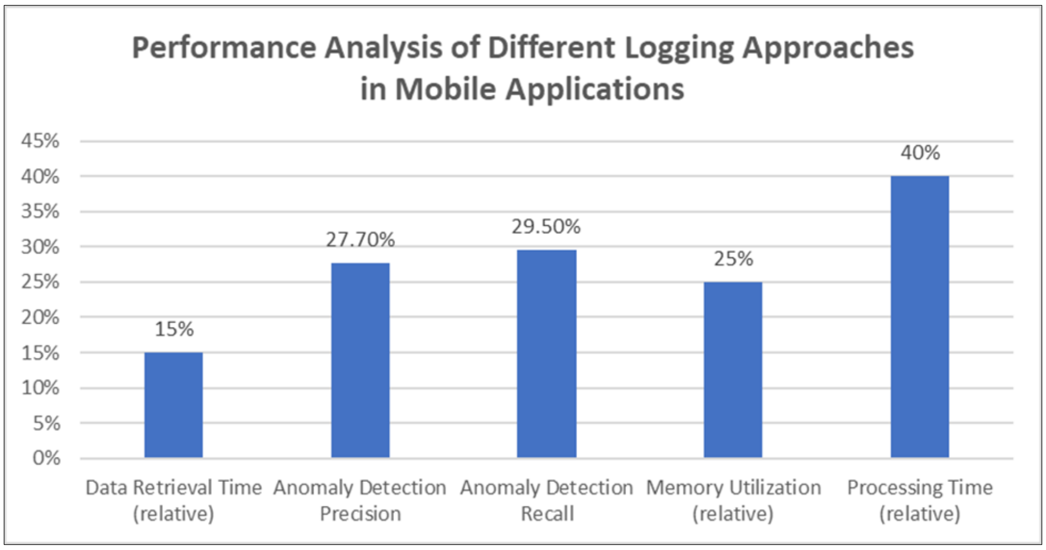

Figure 1 Comparison of Logging System Performance Metrics [6, 7]

4. Mobile-Specific Challenges

4.1. Network-Related Challenges

Mobile applications face unique logging challenges due to their network-dependent nature. Research on iOS application logging patterns demonstrates that protocol-agnostic logging frameworks significantly influence error capture capabilities. Studies show that effective logging frameworks must handle multiple protocols including REST APIs, gRPC, and WebSocket connections to ensure comprehensive error capture across varied communication methods [18].

4.1.1. Network Connection State Changes

Network state management presents significant challenges in mobile logging systems. Research emphasizes the critical nature of handling connection state changes, particularly during transitions between WiFi and cellular networks. Studies highlight the importance of maintaining logging continuity during network transitions while preserving data integrity [18].

4.1.2. Bandwidth Constraints

Efficient bandwidth utilization remains crucial for mobile logging systems. Industry analysis emphasizes the importance of implementing proper log sampling and filtering strategies. Best practices recommend specific logging levels: ERROR for exceptions that affect functionality, WARN for potentially harmful situations, INFO for interesting runtime events, and DEBUG for detailed information useful during development [19].

4.1.3. Protocol-Specific Issues

Protocol handling poses unique challenges in mobile environments. Research demonstrates how protocol-agnostic approaches effectively address error logging across different communication protocols. Modern frameworks provide consistent error-tracking capabilities across REST APIs, gRPC services, and WebSocket connections, highlighting the importance of protocol-independent error capture mechanisms [18].

4.2. Battery and Resource Optimization

4.2.1. Battery Impact

Mobile logging systems must carefully consider battery consumption. Industry guidelines recommend implementing log buffering and batch transmission strategies to optimize resource usage. Research emphasizes the importance of configuring appropriate log levels and implementing proper log aggregation to minimize system resource utilization [19].

4.2.2. Storage Management

Efficient storage utilization proves crucial for mobile logging systems. Studies highlight the importance of implementing log rotation and compression strategies for optimal storage management. Research emphasizes structured logging approaches and proper retention policies to maintain effective diagnostic capabilities while managing storage constraints [18].

Table 1 Classification of Mobile-Specific Logging Challenges [18, 19]

Challenge Category	Sub-Challenge	Key Considerations	Protocol/Technology
Network-Related	Connection State Changes	WiFi-Cellular Transitions, Logging Continuity	Network State Management
	Protocol Handling	Error Capture, Data Integrity	REST API, gRPC, WebSocket
	Bandwidth Usage	Log Sampling, Data Filtering	Network Protocols
Log Levels	Error Logging	Exceptions Affecting Functionality	ERROR Level Logging
	Warning Logging	Potentially Harmful Situations	WARN Level Logging
	Information Logging	Runtime Events	INFO Level Logging
	Debug Logging	Development Information	DEBUG Level Logging
Resource Management	Battery Optimization	Log Buffering, Batch Transmission	Battery-Aware Logging
	Storage Optimization	Log Rotation, Compression	Storage Management

5. Implementation Strategy

5.1. Conditional Transmission Policies

One of the most important aspects of contemporary logging systems is the application of intelligent transmission policies that represent a cornerstone of modern logging systems. Research indicates that AI-driven log management systems demonstrate significant capabilities in real-time log data processing and analysis. Studies emphasize the importance of selective transmission for maintaining system performance, particularly in resource-constrained mobile environments where battery life and bandwidth optimization are critical [9].

The impact of intelligent transmission policies extends beyond basic resource management. Organizations implementing these advanced systems report enhanced capabilities in identifying and resolving issues. Research highlights how machine learning algorithms enable the detection of subtle patterns in log data, facilitating a more proactive approach to problem resolution [9].

5.2. Continuous Improvement Framework

The evolution of logging systems through continuous improvement frameworks has significantly influenced system reliability and maintenance efficiency. Studies examining AI-driven business development frameworks across multiple cases demonstrate that organizations implementing systematic improvement processes achieve measurable gains in operational efficiency. Research spanning multiple organizations reveals how structured approaches to system enhancement led to improved performance metrics [8].

Long-term analysis reveals the cumulative benefits of ongoing improvement practices. Organizations utilizing AI-driven frameworks show marked improvements in their ability to identify and address system issues. Studies emphasize that organizations implementing regular review cycles consistently outperform those using ad-hoc approaches, highlighting the value of systematic feedback loops in enhancing system performance [8].

The effectiveness of machine learning model updates in logging systems demonstrates progressive improvement over time. Research indicates that organizations experience significant initial gains during early implementation phases, followed by sustained incremental improvements. Studies show that organizations maintaining consistent improvement cycles achieve superior results in pattern recognition and anomaly detection compared to those using static configurations [8].

6. Strategic Logging and Cost Management

6.1. Local Event Chain Caching and Resource Optimization

Implementing local event chain caching represents a fundamental shift in enterprise logging strategies, critically reducing retrieval times and enhancing the contextual analysis of logs. This method is particularly valuable in environments such as mobile platforms where errors may originate far from where they are detected but can be traced through a series of interconnected events. For instance, in mobile application scenarios, errors that manifest on the front-end interface often have causes buried in back-end processes; such as an API failure occurring several steps before an error message appears to the user. By employing local caching of user actions and server responses—from user login to the point of error—engineers can meticulously reconstruct and diagnose issues. Some key considerations for implementing local event chain caching include:

- Caching Key Metadata: Log essential details such as timestamps, app state, and user interactions which are vital for later analysis.
- Using Efficient Data Structures: Implement lightweight, memory-efficient data structures to enhance the speed and efficacy of caching mechanisms.
- Puring Regularly: Routinely clear informational logs that are not linked to errors in order to maintain system efficiency and ensure data privacy.

6.1.1. The example workflow would like

- Screen A - Login Screen: When a user logs in, the system caches the login timestamp and user state.
- Screen B - API Interaction: As the user interacts with an API, the system caches both the request and the successful response.

- Screen C - Error Detection: If an error occurs, the system refers to the cached logs from Screen A to Screen B, allowing engineers to quickly pinpoint the root cause and contextual timeline leading up to the error.

This approach aligns with the recommendation to use lightweight, memory-efficient data structures for local caching, routinely purging informational logs if no errors are detected within the chain, thereby enhancing operational efficiency and granting engineers detailed, actionable insights during the debugging process [10].

6.2. Error Logging and Conditional Transmission

Effective logging strategies entail a three-tier structure to capture basic error messages, detailed debugging information, and full stack traces. This comprehensive approach enhances error resolution capabilities by providing depths of data essential for extensive diagnostics. For example, utilizing conditional transmission policies, a mobile e-commerce application could intelligently determine which error logs, such as critical payment failures, are transmitted to central servers while less severe logs are processed locally. Some key considerations for implementing error logging with conditional transmission include:

- Classifying logs using ML to differentiate between informational, warning, and error logs.
- Prioritizing transmission of logs that contain actionable insights for immediate debugging.

6.2.1. Example Workflow

- An error in payment processing on a mobile e-commerce app is detected. Critical logs related to this error are immediately transmitted to central servers for rapid analysis and resolution.
- Non-critical logs remain processed locally, conserving bandwidth and reducing server load.

This mirrors the practices of using ML to classify logs and prioritize event chains that offer actionable insights for debugging, thus smartly conserving network resources and improving systemic responsiveness [11]

6.3. Understanding Error Origins and Propagation

In-depth analyses of Strategic Enterprise Management (SEM) systems across various business sectors demonstrate the efficacy of integrated information systems that allow real-time operational adjustments. By leveraging AI-enhanced log analysis, organizations can trace error origins and propagation paths dynamically, thus elevating responsiveness and decision-making across operational levels. An AI model applied in a logistics application, for instance, could trace a frequently occurring error in package tracking back to a flawed data entry interface, prompting rapid user-interface redesign and testing. Some key considerations for implementing detection of error origins and propagation include:

- Deploying AI models to analyze log patterns and predict potential erroneous outcomes.
- Utilizing real-time analytics to update and adjust operational strategies promptly.

6.3.1. Example Application

- Anomaly in logistics tracking is traced back to user input errors.
- AI suggests immediate UI enhancement to simplify data entry, reducing future errors.

This capability substantiates how real-time access to integrated log data improves overall operational agility and effectiveness [10].

6.4. Server-Side Analysis and Threshold Management

Server-side analysis is crucial in refining error categorization and enforcing standardized logging formats, further facilitated by machine learning algorithms. Mobile platforms can use these AI capabilities for dynamic threshold management, adapting as data patterns evolve to maintain system reliability and preempt fault propagation. In an online streaming service, an AI system might monitor and adjust thresholds for buffering issues to prevent widespread disruption during high-traffic events [11]. Some key considerations for implementation include:

- Utilizing machine learning to adaptively manage error thresholds based on ongoing data analysis.
- Continuously refining error categorization to align with evolving system dynamics.

6.4.1. Example Policy

During peak times, an AI system dynamically adjusts thresholds for system resource usage to maintain performance without service disruption.

6.5. Cost and Resource Efficiency

Strategic logging improves operational control and economizes resource allocation by minimizing unnecessary data handling. Studies focusing on integrated management systems demonstrate that organizations using comprehensive information management approaches show improved operational control and resource utilization compared to those using fragmented systems [10].

6.5.1. How to Implement

- Use AI algorithms to filter out redundant and non-essential logs.
- Analyze and adjust logging practices based on ongoing cost-efficiency assessments.

6.5.2. Example Strategy

- Automated systems analyze daily log outputs, adjusting data capture strategies to balance detail with efficiency, ensuring only critical data is stored or transmitted.

6.6. 6.6 Strategic Implementation Benefits

Industry research outlines five essential practices for effective error handling and logging: consistent error messaging, proper error classification, comprehensive error tracking, appropriate recovery procedures, and systematic error documentation.

6.6.1. How to Implement

- Regularly update AI models and logging algorithms based on the latest data and system performance metrics.
- Implement systematic error documentation practices to bolster knowledge management and training.

Analysis shows how these practices contribute to maintaining system stability and facilitating efficient debugging processes [11].

Table 2 Error Handling and Resource Management Strategies in Enterprise Logging [10, 11]

Strategic Component	**Key Features**	**Primary Benefits**
Local Event Chain Caching	Structured Management, Information Processing	Resource Optimization
Error Logging	Basic Error Messages	Initial Diagnostics
	Debugging Information	In-depth Analysis
	Full Stack Traces	Complete Problem Resolution
Server-Side Analysis	Error Categorization	Systematic Classification
	Standardized Logging Formats	Consistency
	Appropriate Severity Levels	Priority Management
	Error Recovery Mechanisms	System Reliability
Strategic Implementation	Consistent Error Messaging	System Stability
	Error Classification	Organized Debugging
	Comprehensive Tracking	Complete Monitoring
	Recovery Procedures	System Resilience
	Error Documentation	Knowledge Management

7. Impact and Benefits

7.1. Operational Efficiency

The integration of AI into logging systems has fundamentally transformed enterprise operations. Research experiments conducted on HDFS (Hadoop Distributed File System) and BGL (Blue Gene/L supercomputer) datasets demonstrated exceptional performance in anomaly detection. The DeepLog system achieved remarkable F1 scores in detecting anomalies, significantly outperforming traditional machine learning methods. The study showed that the model maintained high accuracy while processing log entries at an impressive rate of messages per second [12].

7.2. Maintenance and Administration

The impact on system maintenance and administration has been substantial. Studies of industrial systems revealed that automated diagnostic approaches markedly improved root cause identification accuracy. Research conducted on service-oriented systems showed significant reductions in diagnosis time and manual effort required for problem diagnosis. These improvements demonstrate the transformative potential of AI-driven logging systems in streamlining maintenance processes [13].

7.3. System Reliability

System reliability improvements through AI-enhanced logging have been significant. The implementation of advanced logging systems demonstrated high accuracy rates in identifying system anomalies while maintaining low computational overhead. Experiments on large-scale datasets showed that these systems could effectively process system logs with minimal false positive rates, demonstrating superior performance compared to traditional approaches [12].

7.4. Team Impact

The effect on engineering team performance has been measurable. Research documented substantial success rates in diagnosing intermittent failures, a particularly challenging category of system issues. The systems demonstrated effectiveness across various problem types, particularly in identifying and resolving multi-component failures in distributed systems. These improvements have significantly enhanced team efficiency and problem-resolution capabilities [13].

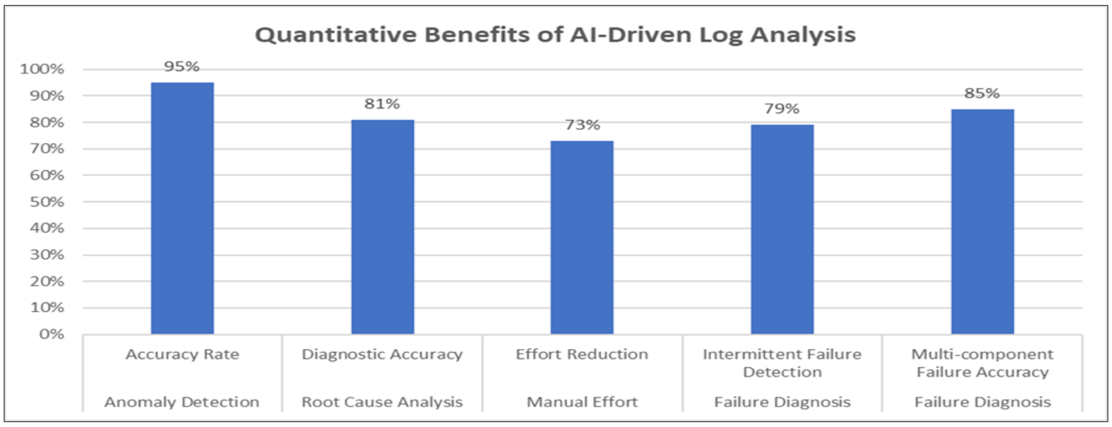

Figure 2 Performance Metrics of AI-Enhanced Logging Systems [12, 13]

8. Addressing Challenges through AI-Driven Solutions

AI-driven event-logging systems have emerged as effective solutions to common implementation challenges in distributed systems. According to a research analysis [14], distributed system logging must address four fundamental challenges: clock synchronization, failure handling, message ordering, and state consistency. Their documentation emphasizes that logging systems must maintain causal relationships between events while handling the complexity of concurrent operations across multiple nodes.

8.1. Over-Logging Mitigation

The challenge of excessive log generation requires sophisticated management approaches. CyberProof [15] identifies three primary areas requiring optimization in log collection: log source identification, data normalization, and log retention policies. Their analysis emphasizes that organizations must implement structured approaches to log filtering and aggregation to maintain system efficiency while preserving essential diagnostic information. By utilizing machine learning-based log prioritization, we can intelligently filter and reduce the volume of data logged, focusing only on critical events that require attention.

8.2. Performance Management

Managing system performance while maintaining effective logging capabilities presents unique challenges. A research study [14] documents three main types of failures that distributed systems must handle: network partitions, node crashes, and message losses. Their analysis describes how logging systems must implement failure detection mechanisms while maintaining consistent logging practices across distributed components. By leveraging generative AI for root cause analysis, it is possible to quickly identify and address the underlying causes of system failures, thus improving the overall resilience and performance of the system.

8.3. Microservice Architecture Integration

The complexity of logging in microservice architectures introduces specific challenges that modern logging solutions must address. A research study [14] emphasizes three key requirements for distributed logging systems: maintaining causal ordering of events, implementing vector clocks for temporal ordering, and handling concurrent events. The documentation outlines how these requirements ensure proper event tracking across distributed services. By employing local event chain caching, we can maintain an organized and accessible log structure, which helps in preserving the order and integrity of event data within the microservice architecture.

8.4. Data Synchronization Solutions

CyberProof [15] outlines three essential components for managing log collection: source identification, format standardization, and data normalization. Their analysis describes how organizations should approach log collection optimization through systematic planning and implementation of standardized logging practices across their environments. The documentation emphasizes the importance of maintaining consistent logging standards across all system components. By applying conditional transmission policies, we can optimize data synchronization and reduce redundancy, ensuring that only pertinent logs are transmitted based on predefined criteria such as error severity and occurrence frequency.

9. Future Directions

The future of AI-driven logging systems points toward significant developments in system observability and maintenance. The report [16] identifies three key areas where AI will transform log analysis: automated log parsing, pattern recognition, and anomaly detection. Its analysis particularly emphasizes how future logging systems need to evolve to handle the complexity of modern cloud-native applications, where traditional logging approaches become insufficient. The research outlines how next-generation systems will need to incorporate natural language processing capabilities and a semantic understanding of log data.

The evolution of performance monitoring and analysis systems presents new challenges and opportunities. A research study [17] identifies four primary areas requiring innovation in future logging systems: performance data collection, analysis of distributed systems, adaptation to changing environments, and intelligent sampling techniques. Their research particularly emphasizes how modern applications generate multiple types of performance data that require more sophisticated analysis approaches than are currently available.

The report [16] emphasizes two critical requirements for future logging systems: the ability to understand relationships between different components and the capability to automatically adjust logging behavior based on system state. The analysis particularly focuses on the challenges of implementing these capabilities in microservices architectures, where traditional logging approaches often fall short of meeting modern observability requirements.

A research study [17] outlines three fundamental challenges that future logging systems must address: balancing comprehensive monitoring with system performance, developing adaptive monitoring capabilities, and maintaining effectiveness across distributed environments. Their work particularly emphasizes the importance of developing intelligent monitoring approaches that can adapt to changing system conditions while maintaining monitoring effectiveness.

10. Conclusion

The implementation of AI-driven event logging systems marks a significant advancement in mobile platform engineering. These systems offer solutions that skillfully balance comprehensive diagnostics with resource efficiency. By integrating machine learning and generative AI technologies, they have achieved substantial improvements in anomaly detection, root cause analysis, and system reliability, all while optimizing resource utilization. The adoption of protocol-agnostic approaches and intelligent transmission policies has significantly enhanced the capabilities of mobile application logging, effectively addressing the unique challenges of mobile environments. As logging systems continue to evolve, incorporating increasingly sophisticated AI features and adaptive monitoring techniques, they will play an essential role in ensuring the reliability and performance of mobile applications. This article highlights the transformative impact of AI-driven logging systems on mobile platform engineering, emphasizing their potential to redefine system observability and maintenance in mobile computing environments.

References

[1] Ziema Mushtaq and Abdul Wahid, "Cost Estimation for Mobile Application Development: Review," Research Gate Publication, July 2018. Available: https://www.researchgate.net/publication/327154965_Cost_Estimation_for_Mobile_Application_Development_Review

[2] Julian Harty et al., "Logging Practices with Mobile Analytics: An Empirical Study on Firebase," Research Gate Publication, April 2021. Available: 10.48550/arXiv.2104.02513

[3] Jieming Zhu et al., "Loghub: A Large Collection of System Log Datasets for AI-driven Log Analytics," 2023 IEEE 34th International Symposium on Software Reliability Engineering (ISSRE), April 2021. Available: 10.1109/ISSRE59848.2023.00071

[4] Anachack Phongtraychack and Darya Dolgaya, "Evolution of Mobile Applications," Research Gate Publication, Jan. 2018. Available: 10.1051/matecconf/201815501027

[5] Rawand Raouf and Alaa Khalil Jumaa, "Log File Analysis Based on Machine Learning: A Survey," Research Gate Publication, Oct. 2022, pp. 1-6. Available: 10.21928/uhdjst.v6n2y2022.pp77-84

[6] Jhonny Mertz and Ingrid Oliveira de Nunes, "A Qualitative Study of Application-level Caching," IEEE Transactions on Software Engineering, Oct. 2020. Available: 10.48550/arXiv.2011.00242

[7] Ashot N. Harutyunyan, "On Machine Learning Approaches for Automated Log Management," Journal of Universal Computer Science, vol. 25, no. 8 (2019), 925-945, 2019. Available: http://www.math.sci.am/sites/default/files/member_publications/JUCS_2019.pdf

[8] Meenu Mary John, Helena Holmström Olsson, and Jan Bosch, "Towards an AI-driven business development framework: A multi-case study," Journal of Software: Evolution and Process, Feb. 2022. [Online]. Available: https://onlinelibrary.wiley.com/doi/full/10.1002/smr.2432

[9] Stuart Burns, "Log management systems benefit greatly from AI," TechTarget SearchITOperations, May 2017. [Online]. Available: https://www.techtarget.com/searchitoperations/tip/Log-management-systems-benefit-greatly-from-AI

[10] Stan Brignall and Joan Ballantine, "Strategic Enterprise Management Systems: new directions for research," Management Accounting Research 15(2):225-240, June 2004. Available: 10.1016/j.mar.2003.10.003

[11] FasterCapital, "Strategies For Effective Error Handling And Logging," FasterCapital Technology Hub. Available: https://fastercapital.com/topics/strategies-for-effective-error-handling-and-logging.html

[12] Yang Zhang et al., "DeepLog: Deep-Learning-Based Log Recommendations," 2023 IEEE/ACM 45th International Conference on Software Engineering: Companion Proceedings (ICSE-Companion), July 2023. Available: https://ieeexplore.ieee.org/document/10172876

[13] Pinjia He et al., "An Evaluation Study on Log Parsing and Its Use in Log Mining," 2016 46th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), Oct. 2016. Available: https://ieeexplore.ieee.org/document/7579781

[14] GeeksForGeeks, "Logging in Distributed Systems," GeeksForGeeks Technical Documentation, Sep. 2024. Available: https://www.geeksforgeeks.org/logging-in-distributed-systems/

[15] CyberProof, "Optimize Log Collection," CyberProof Solutions Documentation. Available: https://www.cyberproof.com/solutions/optimize-log-collection/

[16] N. Paul, "Creating Next-Gen Log Analysis with AI," LinkedIn Technical Articles, 2017. Available: https://www.linkedin.com/pulse/creating-next-gen-log-analysis-ai-draft-nayan-paul

[17] Amirmahdi Khosravi Tabrizi,, "An Adaptive Logging System (ALS): Enhancing Software Logging with Reinforcement Learning Techniques," International Conference on Performance Engineering (ICPE), May 2024. Available: https://research.spec.org/icpe_proceedings/2024/proceedings/p37.pdf

[18] W. Syed and N. Madhavan, "Optimized Network Error Logging for iOS: Enhancing User Experience and App Stability via Protocol Agnosticism and Server-side Policies," International Journal of Research In Computer Applications and Information Technology (IJRCAIT), Volume 7, Issue 2, July-December 2024. Available: https://iaeme.com/MasterAdmin/Journal_uploads/IJRCAIT/VOLUME_7_ISSUE_2/IJRCAIT_07_02_013.pdf

[19] Coralogix, "Application Logging: Definition, Examples, and Best Practices," Coralogix Documentation. Available: https://coralogix.com/guides/application-performance-monitoring/application-logging-best-practices/