Karma Thinley, 2023, 11:6
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology
An Open Access Journal

Modern Engineering Approaches to Cloud and Network System Design

Karma Thinley
Samtse College of Education

Abstract - Cloud computing and advanced network infrastructures have fundamentally reshaped the architectural foundations of modern digital systems. The exponential growth of distributed applications, Internet of Things (IoT) ecosystems, large-scale data analytics platforms, and artificial intelligence (AI)-driven workloads has introduced unprecedented requirements in terms of scalability, latency, resilience, security, and operational efficiency. Traditional monolithic, hardware-dependent, and manually configured infrastructures are increasingly unable to meet these demands. Consequently, system engineering paradigms have evolved toward software-defined, virtualized, automated, and intelligence-driven models that emphasize flexibility, programmability, and elasticity. Contemporary cloud and network system design is characterized by the adoption of microservices-based architectures, containerization technologies, and orchestration frameworks that enable modular development and rapid deployment. Software-Defined Networking (SDN) and Network Function Virtualization (NFV) have redefined network management by decoupling control and data planes and virtualizing network services, thereby enhancing agility and reducing infrastructure costs. Additionally, edge and fog computing paradigms extend computational capabilities closer to data sources, addressing latency-sensitive and bandwidth-intensive application requirements. Modern engineering approaches also integrate cloud-native principles, Infrastructure as Code (IaC), DevOps practices, and zero-trust security frameworks to ensure operational consistency, resilience, and proactive threat mitigation. Artificial intelligence and machine learning further augment system management through predictive analytics, anomaly detection, automated scaling, and intelligent orchestration, enabling adaptive and self-optimizing infrastructures. This review synthesizes current research literature and industry best practices to analyze architectural patterns, performance optimization techniques, fault-tolerance mechanisms, and security-centric design strategies that define next-generation cloud-network ecosystems. It also critically examines emerging challenges, including multi-cloud interoperability, regulatory compliance, energy sustainability, and the increasing complexity of distributed observability. By providing a comprehensive and structured overview, this review contributes to a deeper understanding of how modern engineering methodologies are converging to build scalable, secure, resilient, and intelligent cloud-network infrastructures.

Keywords - Cloud Computing; Cloud-Native Architecture; Distributed Systems; Microservices Architecture; Containerization; Infrastructure as Code (IaC); DevOps; Software-Defined Networking (SDN); Network Function Virtualization (NFV); Edge Computing; Fog Computing; Network Automation; AI-Driven Networking; Zero-Trust Security; Resilience Engineering; Multi-Cloud Architecture; Sustainable Cloud Systems; Intelligent Infrastructure Management.

I. INTRODUCTION

The rapid digital transformation of enterprises, governments, and service providers has fundamentally reshaped the design requirements of modern computing systems. Organizations increasingly depend on cloud-based platforms and distributed network infrastructures to support mission-critical operations, real-time analytics, and globally accessible services. The proliferation of data-intensive applications, artificial intelligence workloads, and connected devices has placed unprecedented demands on system scalability, availability, and performance. As a result, traditional static and hardware-bound infrastructures are no longer sufficient to meet evolving operational expectations (Crooks & Valsan, 2019).

Modern applications must operate in environments characterized by fluctuating workloads, geographically distributed users, and stringent performance constraints. High availability, elasticity, and fault tolerance have become baseline requirements rather than optional enhancements. Additionally, real-time processing capabilities are essential for domains such as financial technology, telemedicine, industrial automation, and smart cities. These demands have accelerated the transition toward cloud-centric and network-aware architectural models (Popovic & Naumovic, 2019).

Engineering methodologies have consequently evolved from manual configuration and vertically integrated systems toward programmable, automated, and software-defined frameworks. Infrastructure is no longer treated as fixed hardware; instead, it is abstracted into flexible, virtualized resources that can be dynamically provisioned and managed. Automation, orchestration, and policy-driven control mechanisms now form the backbone of scalable system design (Gilbert et al., 2019).

The convergence of cloud computing and advanced networking technologies has created highly distributed ecosystems where computation, storage, and networking resources operate cohesively. This convergence requires interdisciplinary engineering approaches that integrate principles from distributed systems, networking, cybersecurity, and software engineering. The emphasis has shifted from isolated system optimization to holistic ecosystem design (Azizi et al., 2015).

This review explores the dominant engineering paradigms shaping contemporary cloud and network infrastructures. It examines architectural transitions, automation strategies, virtualization technologies, resilience mechanisms, and security frameworks that collectively define modern system engineering practices (Habl et al., 2017).

II. EVOLUTION OF CLOUD SYSTEM DESIGN

From Monolithic to Microservices Architecture

Early enterprise systems were typically designed as monolithic applications, where all functional components were tightly integrated into a single deployable unit. While such architectures simplified initial development and deployment, they often became rigid and difficult to maintain as applications grew in scale and complexity. Scaling a single component required redeploying the entire application, leading to inefficiencies and increased risk of downtime (Gadhavi et al., 2016).

As application ecosystems expanded, the limitations of monolithic designs became more apparent. Large codebases hindered agile development practices, and interdependent modules increased the likelihood of cascading failures. Moreover, technology stack rigidity limited innovation, as adopting new programming languages or frameworks required extensive system-wide modifications (Yan et al., 2017).

Microservices architecture emerged as a solution to these constraints by decomposing applications into loosely coupled, independently deployable services. Each microservice encapsulates specific business functionality and communicates with others through lightweight APIs. This modular approach enables independent scaling, faster release cycles, and improved fault isolation (Flaishans et al., 2016).
The microservices paradigm aligns well with cloud environments, where elastic resource allocation supports granular scaling of individual services. Containerization technologies further enhance this model by ensuring consistency across development and production environments. However, microservices introduce operational complexities related to service discovery, distributed logging, observability, and network latency management (Cao, 2020).

To address these challenges, engineering practices now incorporate service meshes, distributed tracing frameworks, and centralized monitoring platforms. These supporting technologies ensure that the benefits of microservices—agility and scalability—

are not overshadowed by operational overhead (Bolodurina, 2018).

Containerization and Orchestration

Containerization represents a significant milestone in cloud system evolution. Unlike traditional virtual machines, containers share the host operating system kernel while maintaining isolated execution environments. This lightweight abstraction improves startup speed, resource efficiency, and portability across heterogeneous environments (Dieves, 2016).

Containers package applications along with their dependencies, ensuring consistent behavior across development, testing, and production stages. This consistency mitigates environment-related deployment failures and enhances reproducibility. As organizations increasingly adopt DevOps practices, containerization has become integral to continuous integration and continuous deployment pipelines (Azizi et al., 2018).

However, large-scale container deployments require sophisticated management mechanisms. Orchestration platforms automate container scheduling, scaling, health monitoring, and networking. These systems ensure optimal resource utilization while maintaining application reliability under dynamic workloads (Popovic & Naumovic, 2019).

Self-healing capabilities represent a major advancement in orchestration frameworks. Failed containers are automatically restarted or replaced, ensuring service continuity. Load balancing mechanisms distribute traffic efficiently, while horizontal scaling adapts to workload variations in real time (Habl et al., 2017).

Together, containerization and orchestration form the foundation of cloud-native engineering. They enable dynamic infrastructure management, reduce operational complexity, and support highly resilient distributed systems (Azizi et al., 2015).

Infrastructure as Code (IaC)

Infrastructure as Code (IaC) transforms infrastructure provisioning from manual configuration into automated, declarative processes. Engineers define infrastructure components—such as networks, virtual machines, and storage resources—using configuration files that can be version-controlled and reused (Crooks & Valsan, 2019).

This approach enhances reproducibility and transparency, allowing teams to replicate environments across regions or cloud providers with minimal manual intervention. IaC reduces human error, which is a common source of system misconfiguration and security vulnerabilities (Gadhavi et al., 2016).

By integrating with DevOps workflows, IaC enables automated infrastructure provisioning during application deployment. This integration supports continuous delivery models, where infrastructure changes are treated similarly to software updates (Yan et al., 2017).

Policy enforcement can also be embedded into IaC templates, ensuring compliance with organizational standards and regulatory requirements. Automated validation mechanisms further improve system reliability and security posture (Flaishans et al., 2016).

Overall, IaC promotes consistency, scalability, and operational efficiency, making it a cornerstone of modern cloud engineering (Cao, 2020).

Modern Network Engineering Approaches

Software-Defined Networking (SDN)

Traditional networks rely on tightly coupled control and data planes within networking devices. This architecture limits flexibility and complicates network-wide policy implementation. Software-Defined Networking (SDN) addresses these limitations by separating control logic from forwarding functions (Bolodurina, 2018).

In SDN architectures, centralized controllers manage network behavior programmatically. This abstraction enables dynamic configuration, rapid policy deployment, and real-time traffic management. Network administrators gain a global view of the

infrastructure, enhancing visibility and control (Dieves, 2016).

SDN facilitates automation in data centers and cloud environments, where frequent configuration changes are common. Programmable interfaces enable integration with orchestration platforms and security systems (Azizi et al., 2018).

Furthermore, SDN supports network virtualization, allowing multiple logical networks to coexist on shared physical infrastructure. This capability improves resource utilization and simplifies tenant isolation in multi-tenant cloud environments (Gilbert et al., 2019).

Despite its advantages, SDN introduces concerns related to controller scalability and single points of failure. Redundant controller architectures and distributed control models are employed to mitigate these risks (Azizi et al., 2015).

Network Function Virtualization (NFV)

Network Function Virtualization (NFV) replaces proprietary hardware appliances with software-based network functions deployed on standard servers. Firewalls, load balancers, and intrusion detection systems can now operate as virtual instances within cloud infrastructures (Habl et al., 2017).

This virtualization reduces capital expenditure by eliminating specialized hardware requirements. It also accelerates service deployment, as new network functions can be instantiated rapidly without physical installation (Gadhavi et al., 2016).

NFV complements SDN by enabling flexible traffic routing through virtualized functions. Together, they create programmable and agile network environments suited for modern cloud services (Yan et al., 2017).

Scalability is significantly improved, as virtual network functions can be replicated or resized based on demand. This elasticity aligns with cloud-native principles of dynamic resource allocation (Flaishans et al., 2016).

However, performance optimization remains a challenge, particularly for high-throughput applications. Advanced techniques such as hardware acceleration and optimized virtualization layers are employed to address these concerns (Cao, 2020).

Resilience and Reliability Engineering

Modern distributed cloud and network systems operate in environments where component failures are not exceptional events but expected occurrences. Hardware faults, network congestion, software bugs, and even cyberattacks can disrupt services at any time. Consequently, resilience engineering has evolved from being a reactive recovery strategy to a proactive architectural principle. Systems are now designed under the assumption that failures will occur, and the primary objective is to maintain service continuity and performance despite such disruptions (Crooks & Valsan, 2019).

Redundancy and replication constitute the foundation of reliability engineering in cloud infrastructures. Critical services, databases, and storage systems are replicated across multiple physical nodes, availability zones, or geographic regions. This eliminates single points of failure and ensures data durability. Replication strategies—such as synchronous and asynchronous replication—are selected based on trade-offs between consistency, latency, and fault tolerance. Load balancers further enhance reliability by distributing incoming traffic across replicated resources, preventing overload and improving response times (Popovic & Naumovic, 2019).

Elastic auto-scaling mechanisms add another dimension to resilience. Instead of provisioning fixed resources, modern cloud systems dynamically allocate compute and storage capacity based on real-time demand. Horizontal scaling increases the number of service instances, while vertical scaling adjusts resource capacity per instance. This elasticity prevents performance degradation during peak traffic conditions and reduces operational costs during low-demand periods. Auto-scaling policies are often driven by performance metrics such as CPU

utilization, memory usage, or request latency (Gilbert et al., 2019).

A more advanced reliability strategy is chaos engineering, which intentionally introduces controlled failures into production or staging environments to test system robustness. By simulating node crashes, network delays, or service interruptions, organizations can identify hidden vulnerabilities before they cause real-world outages. This approach shifts reliability assurance from theoretical planning to empirical validation. The insights gained from such experiments enable engineers to strengthen fallback mechanisms and improve incident response protocols (Azizi et al., 2015).

Finally, geographically distributed deployments significantly enhance fault tolerance. Multi-region and multi-availability zone architectures ensure that localized failures—such as power outages or natural disasters—do not compromise global service availability. Traffic routing mechanisms automatically redirect users to healthy regions during outages. Such distributed resilience strategies are now fundamental requirements for mission-critical cloud services (Habl et al., 2017).

Security-Centric Design

Security has transitioned from a supplementary consideration to a central design principle in modern cloud and network systems. As infrastructures become more distributed and interconnected, the attack surface expands significantly. Traditional security models based on fixed network perimeters are inadequate in environments characterized by remote workforces, multi-cloud deployments, and API-driven communication. Consequently, security must be embedded into architectural decisions from the earliest design stages (Gadhavi et al., 2016).

Zero-Trust Architecture

Zero-trust architecture represents a fundamental shift in cybersecurity philosophy. Rather than assuming that entities within a network boundary are trustworthy, zero-trust models operate under the principle of "never trust, always verify." Every access request—whether originating internally or externally—is subject to strict authentication and authorization checks. This approach significantly reduces the risk of insider threats and lateral movement by attackers (Yan et al., 2017).

Continuous identity verification mechanisms form the core of zero-trust systems. Multi-factor authentication, device health validation, and behavioral analytics ensure that only authorized users and compliant devices gain access to sensitive resources. Access controls are granular and context-aware, considering factors such as user role, geographic location, and risk profile. This dynamic approach enhances adaptability to evolving threat landscapes (Flaishans et al., 2016).

Micro-segmentation further strengthens internal defenses by dividing networks into isolated zones. Even if an attacker compromises one segment, movement to other segments is restricted. This containment strategy limits damage and simplifies incident response. Software-defined networking technologies often facilitate micro-segmentation by enabling fine-grained traffic control policies (Cao, 2020).

Real-time monitoring and analytics are equally critical. Continuous logging, anomaly detection, and automated response systems enable rapid threat identification and mitigation. Machine learning models increasingly support these capabilities by detecting subtle deviations from normal patterns. Together, these mechanisms transform security from a static defense model into an adaptive and intelligent protection framework (Bolodurina, 2018).

Secure API Design

In microservices-based cloud environments, APIs function as the primary communication channels between services, clients, and third-party systems. This centrality makes them frequent targets for exploitation. Therefore, secure API design must incorporate robust authentication, authorization, and encryption mechanisms from the outset (Dieves, 2016).

Token-based authentication frameworks ensure that only verified users and services can access protected endpoints. Access tokens carry identity and permission information, reducing reliance on persistent credentials. Role-based and attribute-based access controls further refine authorization policies to prevent privilege escalation (Azizi et al., 2018).

Transport-level encryption safeguards data confidentiality and integrity during transmission. Secure communication protocols protect against interception, replay attacks, and data tampering. Additionally, rate limiting mechanisms restrict excessive request patterns that may indicate denial-of-service attempts or brute-force attacks (Crooks & Valsan, 2019).

Input validation and output sanitization are essential for preventing injection attacks and data corruption. Comprehensive logging and monitoring systems record API interactions, enabling forensic analysis and anomaly detection. By embedding these safeguards into development workflows, organizations strengthen overall system resilience against cyber threats (Popovic & Naumovic, 2019).

AI-Driven Network and Cloud Automation

Artificial intelligence is rapidly redefining how cloud and network infrastructures are managed. Traditional management approaches rely heavily on manual configuration and rule-based automation, which become insufficient as system complexity increases. AI-driven automation introduces predictive, adaptive, and self-optimizing capabilities into infrastructure operations (Gilbert et al., 2019).

Machine learning models analyze historical and real-time traffic data to predict workload fluctuations. These predictive insights enable proactive resource provisioning, preventing performance bottlenecks before they occur. Traffic engineering algorithms dynamically reroute network flows to optimize latency and throughput. Such intelligent optimization enhances both user experience and infrastructure efficiency (Azizi et al., 2015).

Anomaly detection systems leverage pattern recognition to identify deviations indicative of cyberattacks, hardware failures, or configuration errors. Unlike static rule-based monitoring, AI-based detection adapts to evolving system behaviors. This adaptability improves detection accuracy and reduces false positives, which are common challenges in large-scale distributed systems (Habl et al., 2017).

Predictive maintenance represents another transformative application. By analyzing performance metrics and failure patterns, AI systems can anticipate infrastructure degradation and schedule maintenance before critical breakdowns occur. This approach minimizes downtime and extends hardware lifecycle efficiency (Gadhavi et al., 2016).

Intent-based networking further abstracts infrastructure management by allowing administrators to define high-level operational objectives rather than specific configurations. AI systems translate these intents into network policies and automatically enforce them. As these capabilities mature, cloud-network ecosystems are expected to evolve toward increasingly autonomous operation (Yan et al., 2017).

Challenges and Future Directions

Despite substantial advancements, modern cloud and network engineering faces persistent challenges. Multi-cloud deployments introduce interoperability complexities due to variations in service interfaces, APIs, and management tools across providers. This heterogeneity complicates workload portability and increases operational overhead. Standardization frameworks and abstraction layers are being developed to mitigate vendor lock-in and improve compatibility (Flaishans et al., 2016).

Sustainability has emerged as a critical concern. Data centers consume significant energy resources, contributing to environmental impact. Green cloud engineering focuses on energy-efficient hardware, optimized cooling systems, renewable energy integration, and workload scheduling strategies that minimize carbon footprint. Sustainable design is

likely to become a regulatory as well as ethical imperative (Cao, 2020).

Observability in distributed systems presents another major challenge. As applications span numerous microservices and geographic regions, monitoring becomes increasingly complex. Advanced observability platforms aggregate metrics, logs, and traces to provide unified system visibility. However, balancing detailed monitoring with performance overhead remains an ongoing research problem (Bolodurina, 2018).

Regulatory compliance and data sovereignty requirements add further constraints. Organizations must ensure that data storage and processing align with regional legal frameworks. Engineering solutions increasingly incorporate policy-aware deployment mechanisms that automatically enforce compliance rules (Dieves, 2016).

Looking forward, research is expected to focus on autonomous network architectures, quantum-resistant cryptographic protocols, advanced edge-cloud integration for next-generation communication systems, and the continued evolution of serverless computing models. These innovations will shape the next generation of distributed infrastructure (Azizi et al., 2018).

III. CONCLUSION

Modern engineering approaches to cloud and network system design reflect a fundamental shift toward programmability, scalability, and intelligent automation. The integration of microservices architectures, software-defined networking, network function virtualization, and edge computing has transformed traditional infrastructure into dynamic, distributed ecosystems capable of supporting complex digital services.

These innovations enable elastic resource allocation, improved fault tolerance, and enhanced performance optimization. However, they also introduce new layers of complexity in orchestration, monitoring, and cybersecurity management. Addressing these challenges requires robust design principles, advanced tooling, and interdisciplinary expertise.

Artificial intelligence is poised to play a pivotal role in the next phase of infrastructure evolution. Predictive analytics, autonomous orchestration, and intent-based management models will further reduce human intervention while improving operational efficiency. The movement toward self-healing and self-optimizing systems represents a natural progression of cloud-network engineering.
At the same time, sustainability, regulatory compliance, and security resilience will remain central concerns. Future infrastructures must balance performance innovation with environmental responsibility and ethical data governance.

Ultimately, continued research, cross-domain collaboration, and strategic innovation will be essential to building resilient, secure, and intelligent cloud-network ecosystems capable of sustaining the expanding digital economy.

REFERENCES

1. Crooks, D., & Valsan, L. (2019). Building a minimum viable Security Operations Centre for the modern grid environment. Proceedings of International Symposium on Grids & Clouds 2019 — PoS(ISGC2019).
2. Popovic, N., & Naumovic, M.B. (2019). Networked and Cloud Control Systems - Modern Challenges in Control Engineering. IJEEC - INTERNATIONAL JOURNAL OF ELECTRICAL ENGINEERING AND COMPUTING.
3. Popovic, N., & Naumovic, M.B. (2019). Networked and Cloud Control Systems - Modern Challenges in Control Engineering. IJEEC - INTERNATIONAL JOURNAL OF ELECTRICAL ENGINEERING AND COMPUTING.
4. Gilbert, G.M., Naiman, S., Kimaro, H.C., & Mvungi, N.H. (2019). A Cloud-Fog Based System Architecture for Enhancing Fault Detection in Electrical Secondary Distribution Network.
5. Azizi, A., Yazdi, P.G., Humairi, A.A., Alsalmi, M., Rashdi, B.A., Zakwani, Z.A., & ALSheikaili, S. (2015). Design and fabrication of intelligent material handling system in modern

manufacturing with industry 4.0 approaches. IEEE International Conference on Robotics and Automation.
6. Habl, A., Kipouridis, O., & Fottner, J. (2017). Deploying microservices for a cloud-based design of system-of-systems in intralogistics. 2017 IEEE 15th International Conference on Industrial Informatics (INDIN), 861-866.
7. Burramukku, N. R. (2021). Modeling and implementation of self-defending infrastructure systems using AI-driven security controls. South Asian Journal of Science and Technology, 112, 8–19.
8. Burramukku, N. R. (2021). Performance and security evaluation of Palo Alto NGFWs in hybrid cloud networks. Journal of Management and Science, 11(2), 52–59.
9. Burramukku, N. R. (2021). Enterprise firewall technologies: Evolution from perimeter defense to zero trust. European Journal of Business Startups and Open Society, 1(1).
10. Burramukku, N. R. (2021). A comprehensive review of security challenges in hybrid cloud infrastructure. European Journal of Business Startups and Open Society, 1(1), 54–60.
11. Jangala, V. K. (2021). Secure role-based access control using Spring Security and OAuth 2.0 in distributed systems. TIJER – International Research Journal, 8(3), 39–50.
12. Jangala, V. K. (2021). A systematic review of microservices architecture in enterprise Java applications. International Journal of Science, Engineering and Technology, 9(5).
13. Jangala, V. K. (2021). Continuous integration and continuous deployment tools of enterprise practices. International Journal of Scientific Research & Engineering Trends, 7(6).
14. Koukuntla, S. (2021). Test automation frameworks for modern web and microservices-based applications. TIJER – International Research Journal, 8(2), a11–a18.
15. Koukuntla, S. (2021). Scalable data processing pipelines using serverless and container-based cloud services. European Journal of Business Startups and Open Society, 1(1), 33–48.
16. Koukuntla, S. (2020). Continuous integration and continuous deployment in cloud-native software engineering: A review. International Journal of Engineering Development and Research.
17. Koukuntla, S. (2020). Accessibility and security vulnerability mitigation in modern web applications. International Journal of Creative Research Thoughts, 8(3), 3477–3489.
18. Burramukku, N. R. (2021). Cloud-native network monitoring: Tools, architectures, and best practices. International Journal of Scientific Research & Engineering Trends, 7(5).
19. Burramukku, N. R. (2021). Network digital twin architecture for predictive monitoring and optimization of enterprise networks. International Journal of Science, Engineering and Technology, 9(4).
20. Mandati, S. R. (2021). Adaptive system analysis models for secure cloud and IoT integration over wireless networks. International Journal of Trend in Research and Development, 8(3), 6.
21. Mandati, S. R. (2021). Invisible risks in connected worlds: An IT risk management framework for cloud enabled IoT systems. International Journal of Scientific Research & Engineering Trends, 7(6), 8.
22. Mandati, S. R. (2019). The influence of multi cloud strategy. South Asian Journal of Engineering and Technology, 9(1), 4.
23. Parimi, S. S. (2019). Automated risk assessment in SAP financial modules through machine learning. SSRN Electronic Journal. Available at SSRN 4934897.
24. Parimi, S. S. (2019). Investigating how SAP solutions assist in workforce management, scheduling, and human resources in healthcare institutions. IEJRD – International Multidisciplinary Journal, 4(6),
25. Parimi, S. S. (2020). Research on the application of SAP's AI and machine learning solutions in diagnosing diseases and suggesting treatment protocols. International Journal of Innovations in Engineering Research and Technology, 5.
26. Illa, H. B. (2019). Design and implementation of high-availability networks using BGP and OSPF redundancy protocols. International Journal of Trend in Scientific Research and Development.
27. Illa, H. B. (2020). Securing enterprise WANs using IPsec and SSL VPNs: A case study on multi-site

organizations. International Journal of Trend in Scientific Research and Development, 4(6).
28. Habl, A., Kipouridis, O., & Fottner, J. (2017). Deploying microservices for a cloud-based design of system-of-systems in intralogistics. 2017 IEEE 15th International Conference on Industrial Informatics (INDIN), 861-866.
29. Gadhavi, L.J., Bhavsar, A.K., Vasoya, S., & Bhavsar, M.D. (2016). Design and development of automated and reliable service provisioning cloud architecture for engineering educational domain. 2016 International Conference on Global Trends in Signal Processing, Information Computing and Communication (ICGTSPICC), 266-271.
30. Yan, W., Zhang, R., Tian, C., & Wang, H. (2017). Research and Development of the Engine Combustion System Analysis Platform Based on Cloud Design platform.
31. Flaishans, J., Fry, M.M., Hook, T., Thurman, N., Carleton, J.O., Thawley, S., Wolfe, K., Young, D.F., & Purucker, T. (2016). Scaling Watershed Models: Modern Approaches to Science Computation with MapReduce, Parallelization, and Cloud Optimization.
32. Cao, L. (2020). Design of Digital Library Service Platform based on Cloud Computing. Proceedings of the 2020 International Conference on Computers, Information Processing and Advanced Education.
33. Bolodurina, I.P. (2018). Development and Research of an Adaptive Traffic Routing Algorithm Based on a Neural Network Approach for a Cloud System Oriented on Processing Big Data ?
34. Dieves, V. (2016). Dependability in Future Battle Network System —Transport Layer Ability to Maintain Quality of Service. Wireless Sensor Network, 08, 211-228.
35. Azizi, A., Yazdi, P.G., Humairi, A.A., Alsalmi, M., Rashdi, B.A., Zakwani, Z.A., & ALSheikaili, S. (2018). Applications of control engineering in industry 4.0: utilizing internet of things to design an agent based control architecture for smart material handling system. International Robotics & Automation Journal.