International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

A Review of Scalable Big Data Processing in Cloud-Native Environments

Mr. Swapnil Joshi
Assistant Professor
Department of Computer Sciences and Applications
Mandsaur University, Mandsaur
swapnil.joshi@meu.edu.in

***Abstract*—The vast amount of data created by social media, Internet of Things (IoT) devices, e-commerce platforms, and enterprise applications has presented many difficulties in efficient storage, processing, and analysis of large data sets. The limitations faced by the traditional big data systems in solving today's data-intensive workloads include scalability, flexibility, and processing speed. Cloud-native technologies have proven to be effective solutions as they allow distributed computing, elastic scalability, the ability to deploy to containers and automated management of resources. In this paper, the scalable big data processing in cloud-native environment is reviewed. The study covers how big data architectures have changed over time, with a shift from Hadoop-based to microservices, containerization, orchestration and event-driven architectures. It discusses the key big data processing methods, such as batch, real-time and stream processing, and popular frameworks such as Apache Hadoop, Apache Spark, Apache Flink and Apache Kafka. Moreover, the paper examines the application of Docker, Kubernetes and serverless computing in scalable data processing. Lastly, the issues of security, interoperability, latency and resource optimization are discussed, along with future research directions on efficient cloud-native big data systems.**

***Keywords*—Big Data Processing, Cloud-Native Computing, Scalability, Apache Spark, Stream Processing, Big Data Frameworks.**

Received: 08-04-2026 Accepted: 14-05-2026 Published: 20-05-2026

I. INTRODUCTION

Cloud-native services is a paradigm shift in the design, development, deployment and management of software applications in cloud computing [1]. It takes advantage of the native features of cloud platforms to ensure optimal application performance, availability, scalability and efficiency. The foundation of cloud-native services is a collection of methods and ideas that allow for the development of applications that are highly robust, scalable, and portable [2][3]. A foundational principle of these services is containerization, a technique that isolates and lightweight packages certain application components along with their dependencies. The use of containers optimizes resource consumption, speeds up deployment, and guarantees consistent behaviour in various contexts [4].

Simultaneously, the exponential expansion of big data has created substantial hurdles for traditional computer systems in terms of processing, analysis, and storage. Because of its velocity of production, big data poses challenges for conventional software in terms of storage, processing, and management. When it comes to preserving valuable information, big data solutions are the way to go. Numerous analytical frameworks have been developed to assist users in evaluating both structured and unstructured data, in order to fulfill user needs and evaluate and store complicated data [5][6]. The data included in big data may be accessed by a variety of programs, models, technologies, hardware, and software. These technologies primarily aim to retain precise and dependable outcomes for big data. Furthermore, cutting-edge hardware is needed for big data in order to store and analyze massive volumes of data effectively within a constrained timeframe [7]. When dealing with large datasets, "Big Data processing" is taking the raw data and making it more usable and comprehensible [8]. However, contemporary data-intensive applications frequently place high demands on processing speed and scalability, making it difficult for traditional processing methodologies to keep up. Integrating big data technologies with scalable cloud infrastructures, cloud-native environments provide a viable solution. This allows for distributed computing, fault-tolerant processing, elastic resource provisioning, and more.This paper provides a thorough overview of cloud-native scalable big data processing, including architectures, supporting technologies, and recent advances that enable effective data handling at scale.

A. Structure of the paper

The paper is structured as follows: Section II discusses the fundamentals and challenges of big data. Section III presents cloud native computing concepts. Section IV provides a review of major big data processing techniques and frameworks. The literature overview of current works on cloud-native large data processing is Section V. Finally, Section VI addresses potential avenues for further study.

II. FUNDAMENTALS OF BIG DATA PROCESSING

The phrase "big data" describes increasing amounts of diverse data that are not structured but are rather structured, unstructured, or semi-structured, in contrast to traditional data. The enormous amount of data necessitates cutting-edge technology and sophisticated computations due to its overwhelming complexity [9]. Consequently, big data applications are not compatible with traditional BI technologies. Big data may take many forms, including recordings, reports, notes, and logs, and it comes from a wide variety of sources. Massive data sets include both structured

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

and unstructured information, as well as public and private, local and distant, divided and secret, and full and partial [10].

A. Big Data Types

A huge amount of data is being collected from many places, like social networks, banking networks, and government networks. This exponential data expansion is caused by smart gadgets, the Internet of Things, and other similar technologies [11]. Companies have failed in recent decades when it comes to effectively and long-term data storage. Conventional technologies have this problem since they are expensive and don't have enough store space. A new storage strategy supported by robust technology is needed for huge data. Numerous groups can be established to classify big data. Big data is shown in Figure 1 as being classified [7].

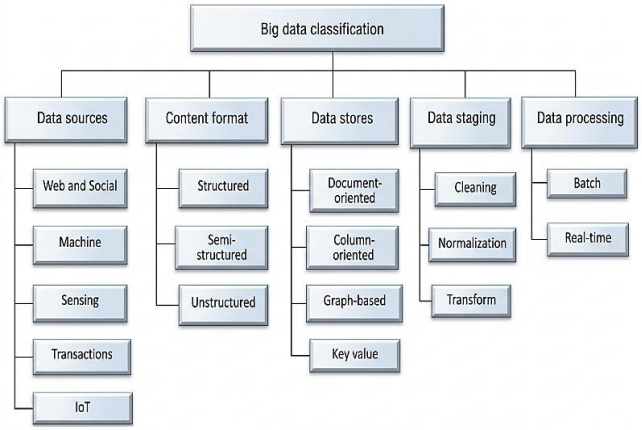


Fig. 1. Big Data Classification

B. The 5Vs of Big Data

Big Data is significant because it allows companies to quickly and accurately collect, store, manage, and analyze massive volumes of data in order to provide valuable insights. Also, in order to get the data to the right process, activity, or predictive analysis/hypothesis, Big Data generators need to reliably create scalable data volumes (Volume) of diverse types (Variety) at controllable generation rates (Velocity). Based on these five features, Big Data has been defined (see Figure 2 below):

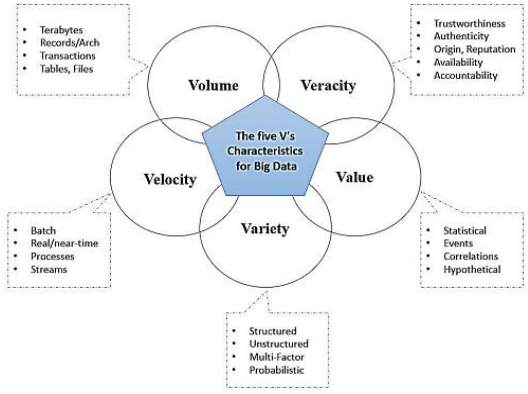


Fig. 2. 5Vs of Big Data

Volume refers to the massive amounts of work that usually need to be done. Big data processing and storage are challenging for many reasons, including the following: scalability (vertical, horizontal, or both) to allow storage and processing power growth; availability (data and methods to operate on it must be accessible); and bandwidth and performance (data must be accessible at the right time).

Big Data frameworks need to be able to deal with a wide variety of data types, as they are frequently generated from numerous sources. A solution to these disparities and the need to consolidate data is Big Data. Data from commonplace items, which may come from a wide variety of places including cellphones, Internet traffic, and wearable electronics, is at the heart of the Internet of Things (IoT), a subfield of Big Data [12].

Data sources vary in their rates, which is what velocity is all about. Data from wireless sensor systems, for example, is updated continuously, whereas Enterprise Data Warehouses (EDWs) are updated once a day. It is essential for Big Data to be able to manage data coming at varying velocities in order to combine data from many sources.

Real data value, or the prospective data value based on the information they contain, is what value is all about. Massive datasets are meaningless if they don't yield useful insights for the researcher.

The reliability of the data is what is meant when talking about its veracity; this includes issues with the data's secrecy, integrity, and accessibility. When data comes from an untrustworthy source, it loses all value. As a result, businesses must check the data for accuracy and validate the results of any analysis. In order to store and analyze data more effectively, the five V's of Big Data work together.

C. Traditional vs. Modern Big Data Architectures

Relational databases and data warehouses have dominated data processing for years. These systems were designed to handle structured data and ensure transactional integrity, making them ideal for applications such as financial transactions and inventory management. Especially in the context of relational databases, ensuring data consistency and enabling complex queries and reports have been crucial [13][14]. Data warehouses, on the other hand, serve to gather and study vast quantities of data from the past, which can be used in business intelligence and strategic decision-making. But with the emergence of big data, new challenges and opportunities have been created, resulting in the birth of modern big data architectures. Big data is data that is too big, too complex or comes in such large volumes that it's difficult for traditional data processing systems to effectively manage.

Modern big data architectures are built to be flexible and scalable, allowing them to ingest, store and analyze a huge amount of data from various sources. A typical data system may involve multiple technologies such as data lakes, data streaming platforms, and advanced analytics tools. For example, data lakes enable the storage of raw data in its native format, which can then be analyzed and processed in a variety of ways. Real-time data processing is essential for applications that need instant insights, like fraud detection or real-time recommendations. Data streaming platforms like Apache Kafka are essential. Advancements in artificial intelligence

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

(AI) and machine learning (ML) have transformed the way data is utilized in processing, creating opportunities for predictive analytics and automated decision-making. These technologies can help organizations identify patterns, trends and insights that might not be apparent in traditional data analysis.

Table I compares traditional data processing systems with modern big data architectures based on scalability, storage, processing methods, flexibility, analytics capability, and support for real-time cloud-native data processing applications.

TABLE I. COMPARISON TRADITIONAL VS MODERN BIG DATA

Aspect	Traditional Data Processing Systems	Modern Big Data Architectures
Data Type	Mainly structured data	Structured, semi-structured, and unstructured data
Storage System	Relational databases and data warehouses	Data lakes, NoSQL databases, distributed storage
Scalability	Limited scalability	Highly scalable and elastic
Processing Method	Centralized processing	Distributed and parallel processing
Speed	Batch-oriented processing	Real-time and stream processing
Data Volume Handling	Suitable for moderate datasets	Handles massive datasets efficiently
Flexibility	Rigid schema design	Flexible schema and dynamic data handling
Analytics Capability	Basic reporting and BI	Advanced analytics, AI, and ML integration
Cost Efficiency	Expensive hardware scaling	Cost-effective cloud-based scaling
Fault Tolerance	Limited fault tolerance	High fault tolerance and redundancy
Use Cases	Banking, inventory management, ERP systems	Social media analytics, IoT, real-time recommendations
Architecture Style	Monolithic and centralized	Cloud-native and distributed
Decision-Making Support	Historical analysis	Predictive and real-time decision-making

D. Challenges in Big Data Processing

Big Data presents two main types of difficulties: technical and semantic. Data management tasks like querying and effective storage present engineering issues. Determining the meaning of information from massive amounts of unstructured data is the semantic issue [15]. Listed below are a number of other obstacles associated with Big Data:

- Digital processing architecture with low power consumption and high processing volume.
- The development of real-time data analysis methods using data-adaptive ML.
- Create data storage that can grow with business, allowing for effective data mining.

The following, however, are a number of critical issues with cloud-based Big Data management:

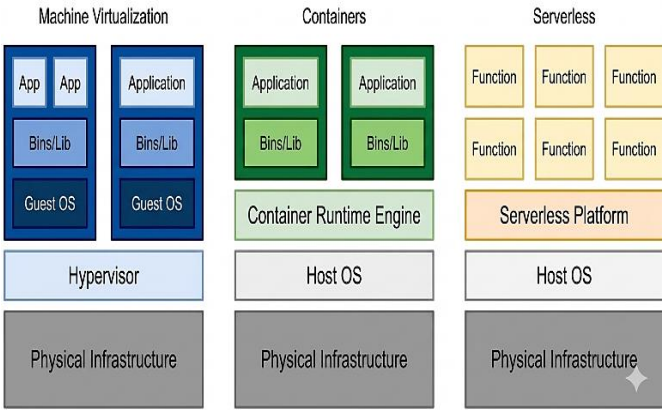
- Protecting personal information;
- Estimated results;
- Investigating data to facilitate deep analytics;
- Web and social media data augmentation for enterprises;
- Streamlining queries
- Multiple tenant performance isolation

III. CLOUD-NATIVE COMPUTING CONCEPTS

The Cloud Native Computing Foundation (CNCF) is an open-source, vendor-neutral center for cloud-native computing. This is a set of technologies that can be used to deploy and orchestrate applications across multiple servers by breaking them down into microservices and packaging them in lightweight containers [16][17]. As a developer, may take advantage of cloud computing by storing data and apps at a remote location that is accessible from anywhere in the world over the Internet. Cloud computing is defined by the National Institute of Standards and Technology (NIST) as three distinct service models that enable users to access shared IT resources.

- **Infrastructure as a Service (IaaS):** IT resources, both real and virtual, such as servers, data storage, and processing power, are made available to users by cloud service providers;
- **Platform as a Service (PaaS):** Customers can use the cloud platforms that providers offer to launch their own apps and make use of pre-built services. Neither the infrastructure nor its operating system is within the authority of the consumer [18].
- **Software as a Service (SaaS):** provides cloud-based apps with the underlying infrastructure concealed. Most of the supplied software (such as Office 365) and the underlying infrastructure are beyond consumers' control.

Figure 3 shows a high-level perspective of the various resource models utilized by cloud-based apps. These were first installed on servers that were bare metal. A more effective method of managing physical resources became available with the advent of machine virtualization. But an operating system image is still needed for each virtual machine, which increases system complexity and memory cost. Since containerization and microservices were developed, virtual machines were swiftly abandoned due to their lightweight and manageability.



International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

Fig. 3. Comparison of Virtualization, Containerization, and Serverless Computing

A. Microservices Architecture

The cloud-native applications are made up of a collection of components called microservices [19]. The microservices are deployed in a self-contained deployment unit (i.e., container). Complex data workflows are broken down into smaller, independently deployable services that are responsible for a specific piece of the functionality [20][21]. This modular design provides improved scalability, resilience, and maintainability. Microservices-based data pipelines can be dynamically scaled and provide high availability and fault tolerance for data processing. The entire pipeline can also be upgraded and replaced piece by piece.

B. Containerization and Orchestration

Containerization and orchestration are key technologies that are essential in modern cloud-native and microservices applications. They facilitate isolated execution environments and automatic service coordination for improved application deployment, scalability, resource management and operational efficiency.

- Containerization. Containerization is a method for isolating functions that uses the Linux kernel to generate isolated processes inside the host operating system and separate resources. The most widely used containerization platform is Docker, which has been under development for a decade. A microservices architecture separates the various components of a program, such as processes and libraries, into their own containers. This makes it easier to isolate resources, be transparent, and reproduce results.
- Orchestration. Orchestration is the process of automatically configuring, managing, and coordinating interdependent microservices to construct scalable and elastic functions. Due to the fact that microservices are housed in containers, orchestration becomes a matter of automating the operational effort involved in managing the life cycle of the containers [22][23]. This includes tasks such as resource provisioning, scheduling, scaling (both up and down), networking, load balancing, and implementation of the applications' workflows or processes [24].

C. Serverless Computing

Serverless computing is a relatively new approach to cloud computing that eliminates the need for users to manage the underlying operational infrastructure. Instead, it allows users to develop and run fine-grained, pay-per-use, automatically scaled applications. Serverless computing, made famous by Amazon Web Services (AWS) Lambda in 2014, is a way to run applications on the cloud without worrying about the underlying logic. Users may create and run apps with fine-grained billing and automated scaling. Serverless computing is frequently paired with Platform-as-a-Service (PaaS) in cloud computing reference models. On the other hand, it stands out because of its event-driven design, automated resource scalability, and, in specific service models, its ability to completely abstract infrastructure management [25].

D. Event-Driven Architectures

Event-driven architectures (EDA) in cloud-native data engineering are a departure from batch processing to a more real-time and reactive approach. EDAs allow these changes in data and occurrences across the system to be treated as discrete events, which allows organizations to build highly responsive, loosely coupled systems that can react to the rapidly changing nature of today's data landscape. Event streaming architectures rely on platforms such as Apache Kafka and Amazon Kinesis as their nervous systems, enabling the smooth transfer of event streams between various data sources and consumers. These benefits have a lot to do with reducing latency in data processing, and also make possible advanced features like Complex Event Processing (CEP) and stream analytics, which can help companies make real-time decisions with their data stream. Event-driven architectures can help businesses to value and use their data pipelines as active decision-makers which can increase data agility and operational intelligence [26]

IV. Big Data Processing Frameworks in Cloud-Native Environments

Cloud-native environments facilitate distributed processing frameworks, containers, virtualization, and serverless architecture, all of which offer the ability to process large amounts of data on a per-user basis without needing to maintain dedicated servers or resources. These technologies can increase application flexibility, fault-tolerance, resource utilization and deployment efficacy for modern large-scale analytics applications [27].

A. Big Data Processing Techniques

Big data processing techniques are the batch processing, real-time processing, and stream processing approaches, which enable efficient processing of historic data and streaming data for scalable analytics, fast decision making and low-latency cloud-native applications.

1) Batch Processing

Batch Processing is the processing of large data sets in an ordered sequence or in a specific fashion. It's used to automate repetitive and time-consuming tasks and drive efficiencies. Batch processing is the processing of data in a batch, in a single large transaction as opposed to by individual transactions.

The banking sector often processes enormous batches of customer transactions overnight, which is an example of batch processing in action. Because of this, the bank can more effectively update client account information and reconcile accounts instead of handling each transaction separately.

Processing data in bulk involves gathering it over time, storing it temporarily, and then running the processing all at once. This may be achieved by utilizing tools like Apache Spark or Hadoop and computer languages like Python or Java and may use these tools for data analysis, reporting, and integration, among other things, and they make big data sets efficient to process. Distributed computing is made possible

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

by the MapReduce programming architecture, but its performance is constrained by disk-based I/O operations. Enterprise installations requiring long-term data preservation and batch processing are ideal for Hadoop due to its maturity, reliability, and broad ecosystem.

2) Real-Time Processing

Real-time processing is one of the big data processing techniques, which enables organizations to immediately analyze and make decisions on the data that arrives in real time. Data processing in this way is essential for companies trying to be responsive to market, customer, or business dynamics changes.

For instance, a retailer might have real-time processing capabilities, as transactions occur, so they can analyze the data and make instant decisions about product recommendations and promotions based on customer preferences. Another example is a financial institution that uses real-time processing to detect and prevent fraudulent transactions as they occur. Organizations normally leverage in-memory databases, data stream processing engines and complex event processing (CEP) systems to implement real-time processing. These technologies enable fast processing of large amounts of data, without any intermediate storage.

Need to be able to process high data velocity, high data volume, and data variety to be successful with real-time processing. For this, they should have the proper infrastructure and data management techniques, such as data ingestion, data quality, and data integration.

3) Stream Processing

Stream Processing is a big data processing method, that handles real-time data streams, which come in as a continuous and sequential stream. The goal of stream processing is to analyze, process and extract insights from data in real-time as it is generated or collected, without having to store it in a database first. A stock market feed, for instance, would generate stock prices and trade data continuously, and the streaming application could be used to process the data. The stream processing engine can identify trends and anomalies in the data, and trigger alerts or other actions based on pre-defined rules.

Apache Kafka is a distributed event streaming platform, commonly used to construct real-time data pipelines and streaming applications. Apache Kafka is a messaging system in which a sequence of records is placed into a queue, and consumers can subscribe to a queue or multiple queues, and read the records as soon as they are added. Apache Kafka isn't just a processing engine, it's a distributed event streaming platform. As the nerve system of real-time data pipelines, Kafka is a reliable, fault-tolerant and high-throughput message streaming solution with low latency. Stream processing in the Kafka ecosystem with Kafka Streams.

Table II presents the big data processing techniques, such as batch processing, real-time processing, and stream processing. There are several techniques with varying processing speed, data-handling capabilities, latency, scalability, and application use for efficient.

TABLE II. COMPARISON OF BIG DATA PROCESSING TECHNIQUES

Feature	Batch Processing	Real-Time Processing	Stream Processing
Processing Style	Processes stored data in batches	Processes data instantly as it arrives	Continuously processes flowing data streams
Latency	High	Low	Very Low
Speed	Moderate	Fast	Very Fast
Data Type	Historical data	Incoming live data	Continuous real-time data
Primary Goal	Large-scale data analysis	Immediate decision-making	Continuous real-time analytics
Example Application	Banking transaction processing	Fraud detection	Stock market and IoT monitoring
Common Technologies	Hadoop, MapReduce	CEP, In-memory databases	Kafka, Flink

B. *Comparative Analysis of Big Data Frameworks in Cloud-Native Environments*

The cloud-native environments have revolutionized how big data frameworks are deployed and managed by offering scalable infrastructure, distributed resource management, and containerized execution. Big data frameworks like HDFS and Hadoop allow large-scale data analysis, real-time processing, and distributed fault-tolerant computing in cloud-based settings. The frameworks like Apache Hadoop, Apache Spark, Apache Flink, and Apache Kafka are popular due to the ability to process workloads in different ways. There are a variety of factors to consider when choosing the right framework, such as scalability, processing speed, latency, fault tolerance, and cloud-native support.

- Apache Hadoop is mainly developed to process data in a batch-oriented fashion with the help of MapReduce programming model and a distributed file system called Hadoop Distributed File System (HDFS). Hadoop is a scalable and reliable solution for handling large historical data. But it has a disk-based architecture for processing, and so it has more latency than modern in-memory processing frameworks.
- Apache Spark brings in-memory distributed computation, which helps to enhance processing performance, especially for iterative and machine learning workloads. Spark can do batch processing and micro-batch stream processing, it can be used to do real-time analytics in cloud-native environment.
- Apache Flink is tailored for processing continuous streams and event-driven applications with low latency! It offers features like advanced event-time processing and state management which are crucial for applications like IoT analytics, fraud detection, and monitoring systems [28].
- Apache Kafka is a distributed event-streaming platform that supports high-performance data pipelines and their fault tolerance. Kafka is often

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

paired with Spark and Flink for real-time data ingestion and processing in cloud-native environments.

The key differences between the frameworks are outlined below:

- **Scalability:** Hadoop, Spark, Flink and Kafka are horizontally scalable and work on distributed cloud clusters. Elastic resource scaling can be further enhanced with Kubernetes orchestration in cloud-native deployments.
- **Processing Capability:** Hadoop is geared toward batch processing, Spark is geared toward batch and micro-batch processing, while Flink is geared toward stream processing, and Kafka is geared toward event streaming and delivery.
- **Performance:** Spark and Flink deliver faster speeds of processing because they work in-memory and feature low latency; Hadoop works on disk, which slows processes down.
- **Fault Tolerance:** All frameworks offer fault tolerance by replicating data, performing checkpoints and having recovery mechanisms. In addition, Kafka provides persistent event storage and guarantees reliable message delivery.
- **Cloud-Native Integration:** New frameworks allow for cloud-native deployment, monitoring and resource management through containerization with Docker and orchestration with Kubernetes [29].
- **Application Areas:** Hadoop is suitable for historical data analytics, Spark for machine learning and iterative processing, Flink for real-time analytics, and Kafka for event-driven architectures and streaming pipelines.

Overall, selecting a big data processing framework requires balancing performance, scalability, operational efficiency, and application-specific requirements to achieve reliable and cost-effective cloud-native data processing solutions.

V. Literature Review

The reviewed studies collectively demonstrate that modern cloud-native architectures significantly improve scalability, elasticity, and processing efficiency compared with traditional Hadoop-based systems. However, challenges related to interoperability, security, orchestration complexity, and resource optimization remain active research areas. Most recent studies emphasize the growing importance of AI-driven automation, serverless computing, and real-time analytics in next-generation big data platforms.

H. Raza et al., (2026) explore the design, capabilities, and relative advantages of processing in batches and real-time in major data science frameworks including Apache Hadoop, Spark, Flink, and Storm. Scalability, latency, data heterogeneity, security, operational complexity, and other big problems are also there, and they might impact the efficiency of the framework. To help obtain excellent analytics, this review delves deeply into the idea of big data frameworks [30].

W. Elouataoui and Y. Gahi (2025) address that gap by evaluating three prevalent data architecture paradigms: Hadoop, Modern, and Cloud-Based stacks. They conduct a thorough end-to-end comparison across the full big data value chain, from ingestion to visualization. Moreover, beyond architectural assessment, implement each stack in a real-world scenario using the Amazon Books Reviews dataset, and evaluate each implementation based on key metrics such as scalability, performance, ease, and deployment costs. The findings aim to provide data professionals with a practical reference for selecting suitable data architectures in big data environments[31].

H. B. Abdalla et al., (2025) examine big data processing's use of map-reducing models, the methods employed in the literature that were studied, and the obstacles that were encountered. In addition, the study covers the latest developments in several map-reduce models, including Hadoop, Hive, Pig, MongoDB, Spark, and Cassandra. This investigation looked at not just trustworthy map-reducing methods, but also a variety of measures used to measure the efficiency of big data processing in different industries. To be more precise, this paper provides a concise overview of MapReduce's history, concepts, types, methods, and applications in order to promote the MapReduce framework for processing large amounts of data [32].

A. I. Abueid (2024) presents a thorough analysis of the potential benefits and uses of cloud computing and big data, and stresses the need for these technologies for companies and groups to embrace in order to stay ahead of the competition in the modern digital marketplace. Additionally, it delves into the many security, ethical, and legal concerns that crop up when handling massive data sets, along with potential solutions to these problems [33].

S. Ponnusamy and P. Gupta (2024) examined the fundamentals of cloud-based distributed data processing and scalable data partitioning methods. Machine learning, data analytics, and real-time data processing are examples of data-intensive activities that are becoming increasingly reliant on cloud platforms, which is why this study is important. This research takes a look at several data-partitioning approaches made to tackle the specific problems with cloud computing. They were tested for their effects on the system's efficiency, load distribution, and scalability [34].

W. Qi, M. Sun, and S. R. A. Hosseini (2023) present state-of-the-art BDM systems for organizations. The most recent studies on cloud computing for managing data created by organizations are reviewed in this article. The results showed that there are several advantages to combining cloud computing with big data, the most important of which are better international trade and more corporate efficiency. The study also brought attention to some risks associated with the complex computer environment [35].

Table III highlights recent advancements in cloud-native big data processing, showing improvements in scalability, performance, fault tolerance, and real-time analytics, while identifying challenges in security, orchestration, resource optimization, and interoperability.

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

TABLE III. SUMMARY OF RECENT STUDIES ON BIG DATA FRAMEWORKS AND CLOUD-BASED DATA PROCESSING

Authors	Study on	Key Findings	Challenges	Limitations	Future Work
H. Raza et al. (2026)	Big data frameworks: Hadoop, Spark, Flink, and Storm	Spark and Flink improved real-time analytics, while Hadoop supported efficient batch processing	Scalability, latency, data heterogeneity, security, and operational complexity	Review-focused with limited experimental validation	Hybrid frameworks, AI-driven resource management, and stronger security models
W. Elouataoui and Y. Gahi (2025)	Comparison of Hadoop, Modern, and Cloud-Based architectures	Cloud-based stacks showed better scalability, flexibility, and deployment efficiency	Integration across analytics pipelines	Limited focus on orchestration and automation	Kubernetes orchestration and automated cloud-native pipelines
H. B. Abdalla et al. (2025)	Survey of MapReduce and big data processing models	Highlighted advancements in Hadoop, Spark, Hive, Pig, MongoDB, and Cassandra	Distributed processing and framework interoperability	Lack of unified benchmarking and practical validation	Advanced optimization and scalable benchmarking models
A. I. Abueid (2024)	Big data and cloud computing applications	Improved business efficiency and digital transformation capabilities	Ethical, legal, and security concerns	Mostly conceptual discussion	Privacy-preserving and AI-based security frameworks
S. Ponnusamy and P. Gupta (2024)	Scalable data-partitioning in cloud environments	Data partitioning improved scalability and load balancing	Partition optimization and workload distribution	Limited heterogeneous cloud evaluation	Intelligent partitioning and adaptive load balancing
W. Qi, M. Sun, and S. R. A. Hosseini (2023)	Big data and cloud integration for organizations	Enhanced organizational efficiency and data management	Security and data governance risks	Limited deployment benchmarking	Secure cloud-native frameworks and AI-driven governance

VI. CONCLUSION AND FUTURE WORK

The production of data generated by smart devices, social media, enterprise applications, and digital services has driven the demand for scalable and efficient big data processing systems to grow. However, traditional architectures can have problems with the large volumes, high velocities and high variety of the data found today, as well as scalability, flexibility and processing power. These limitations have been mitigated by cloud-native technologies that allow for distributed computing, containerization, orchestration and elastic resource management for large-scale data processing applications. This paper summarized the history of the big data processing architectures, from traditional Hadoop-based architectures to cloud-native ones. It talked about major big data processing techniques, including batch processing, real-time processing, and stream processing; and widely used big data processing frameworks, such as Apache Hadoop, Apache Spark, Apache Flink, and Apache Kafka. The comparison revealed that there is a place for each of these solutions: reliable event-driven data streaming can be achieved with Kafka, low-latency stream analytics with Flink, fast in-memory computation with Spark and large-scale batch analytics with Hadoop. Furthermore, the technologies like Docker, Kubernetes and serverless computing enhance scalability, deployment flexibility and resource utilization in distributed environments. To create more reliable, cost-effective systems for cloud-native big data processing, future studies will focus on AI-driven orchestration, intelligent resource optimization, serverless big data frameworks, integration with the edge-cloud, and energy efficient computing and secure multi-cloud interoperability.

REFERENCES

[1] A. Parupalli and H. Kali, "An In-Depth Review of Cost Optimization Tactics in Multi-Cloud Frameworks," *Int. J. Adv. Res. Sci. Commun. Technol.*, vol. 3, no. 5, pp. 1043–1052, Jun. 2023, doi: 10.48175/IJARSCT-11937Q.

[2] J. W. Sajja, G. B. Komarina, and N. K. R. Choppa, "Enterprise Data Transformation in the Era of S/4HANA: Real-World Cloud Migration Architecture, Governance Strategies, and Lessons from the Field," *World J. Adv. Res. Rev.*, vol. 26, no. 2, pp. 3596–3619, May 2025, doi: 10.30574/wjarr. 2025.26.2.2038.

[3] J. B. Mehta, "AI-Driven Test Engineering for Cloud-Native Systems," *Int. J. Data Sci. IoT Manag. Syst.*, vol. 5, no. 1, 2026, doi: 10.64751/ijdim. 2026. v5i1.297.

[4] T. Theodoropoulos *et al.*, "Security in Cloud-Native Services: A Survey," *J. Cybersecurity Priv.*, vol. 3, no. 4, pp. 758–793, 2023, doi: 10.3390/jcp3040034.

[5] M. Parikh, A. A. Soni, S. M. Shah, and A. R. Jha, "Big Data Workload Profiling for Energy-Aware Cloud Resource

International Journal of
DATA SCIENCE AND IOT MANAGEMENT SYSTEM
Peer Reviewed, Referred & Indexed Journal
ISSN: 3068-272X www.ijdim.com Original Research Paper

Management," Jan. 2026, doi: 10.48550/arXiv. 2601.11935.

[6] V. Sanikal, "AI-Enhanced Network Security Framework for Cloud-Edge Integrated Environments," in *2026 Innovations in Machine, Engineering, and Digital Conference (IMED)*, 2026, pp. 1–6. doi: 10.1109/IMED68921.2026.11484417.

[7] A. K. Sandhu, "Big data with cloud computing: Discussions and challenges," *Big Data Min. Anal.*, vol. 5, no. 1, pp. 32–40, 2022, doi: 10.26599/BDMA.2021.9020016.

[8] K. Al-Barznji, "Big Data Processing Frameworks for Handling Huge Data Efficiencies and Challenges: A Survey," *Int. J. Data Sci. Big Data Anal.*, vol. 2, no. 1, pp. 1–9, May 2022, doi: 10.51483/IJDSBDA.2.1.2022.1-9.

[9] H. B. Abdalla, "A brief survey on big data: technologies, terminologies and data-intensive applications," *J. Big Data*, vol. 9, no. 1, 2022, doi: 10.1186/s40537-022-00659-3.

[10] Y. Muni, "Sustainable Data Centers: A Systematic Review Of Technologies And Practices For Carbon Emission Reduction," *Int. J. Adv. Res. Comput. Sci.*, vol. 16, no. 4, pp. 125–131, Aug. 2025, doi: 10.26483/ijarcs.v16i4.7309.

[11] S. Singamsetty, "AI-Based Data Governance: Empowering Trust and Compliance in Complex Data Ecosystems," *Int. J. Comput. Math. Ideas*, vol. 13, no. 03, pp. 1007–1017, 2021, doi: 10.70153/IJCMI/2021.13301.

[12] S. Dodda, N. Kamuni, P. Nutalapati, and J. R. Vummadi, "Intelligent Data Processing for IoT Real-Time Analytics and Predictive Modeling," in *2025 International Conference on Data Science and Its Applications (ICoDSA)*, IEEE, Jul. 2025, pp. 649–654. doi: 10.1109/ICoDSA67155.2025.11157424.

[13] H. B. Dama, "A Survey of MySQL Database Administration Techniques and Best Practices," *ESP J. Eng. Technol. Adv*, vol. 6, no. 1, pp. 89–98, 2026.

[14] H. Ravilla, "Building Scalable Applications with Heroku and Salesforce Integration," *Am. J. Technol.*, vol. 4, no. 3, Dec, pp. 15–36, 2025, doi: 10.58425/ajt.v4i3.454.

[15] S. Kilaru, V. S. R. Yarlagadda, and M. B. Nagaraja, "AI-Augmented Block-Sketch Hybrid Compression: Enabling Semantic Query Processing on Compressed Data," in *2026 IEEE 16th Annual Computing and Communication Workshop and Conference (CCWC)*, Las Vegas, NV, USA: IEEE, 2026, pp. 0966–0972, Jan. doi: 10.1109/CCWC67433.2026.11393750.

[16] V. Sharma, "Cloud-Native 5G Deployments: Kubernetes and Microservices in Telco Networks," *Int. J. Innov. Res. Eng. Multidiscip. Phys. Sci.*, vol. 10, no. 3, pp. 1–8, May 2022, doi: 10.37082/IJIRMPS.v10.i3.232706.

[17] B. Krishnan, A. Thaneeru, R. Lingam, and S. K. Kaata, "The Future of Cloud Data Engineering: Multi-Tenant, Multi-Region Pipelines Leveraging LLM-Powered Data Governance," in *2025 1st International Conference on Advancement in Futuristic Technologies (ICAFT)*, 2025, pp. 1–8. doi: 10.1109/ICAFT66710.2025.11453308.

[18] B. P. Singh, "Convergence of AI and Zero Trust: Enabling Continuous Verification Across Hybrid Cloud Environments," *Int. J. Intell. Syst. Appl. Eng.*, vol. 14, no. 1s, pp. 339–348, 2026.

[19] M. R. C. Mukkolakkal, "IntelliStore: An Intelligent AI Agent Framework for Autonomous Storage and Database Optimization in Cloud-Native Microservices," *Int. J. Sci. Res. Mod. Technol.*, vol. 3, no. 12, pp. 243–250, Dec, 2024, doi: 10.38124/ijsrmt.v3i12.1024.

[20] A. K. Padhy, C. Medicherla, B. Vulugundam, C. Kulkarni, T. P. Patel, and S. Shivam, "Latency-Optimized Microservices Orchestration for Real-Time E-Commerce in Multi-Cloud Environments," in *2025 International Conference on Computer and Applications (ICCA)*, IEEE, Dec. 2025, pp. 1–6. doi: 10.1109/ICCA66035.2025.11430930.

[21] R. N. Rajendran, D. K. Rai, S. K. Anumula, and S. Agrawal, "Zero Trust Security Model Implementation in Microservices Architectures Using Identity Federation," in *2025 2nd International Conference on Recent Trends in Electrical, Electronics and Computing Technologies (ICRTEECT)*, 2025, pp. 1–6. doi: 10.1109/ICRTEECT67512.2025.11448625.

[22] A. Naresh, R. Rao Thallada, and K. Nallabothu, "Risk-Based Governance for Autonomous Decision Systems," *J. Bus. Manag. Stud.*, vol. 8, no. 6, pp. 69–73, 2026, doi: 10.32996/jbms.2026.8.6.5.

[23] P. Naayin and S. Kamatal, "Automating Infrastructure Platforms with Cloud, Kubernetes, and Site Reliability Engineering," *Int. J. Comput. Tech.*, vol. 8, no. 6, 2021.

[24] S. Deng *et al.*, "Cloud-Native Computing : A Survey from the Perspective of Services," pp. 0–32, 2023, doi: 10.36227/techrxiv. 23500383. v1.

[25] V. Besozzi, M. Della Bartola, P. Dazzi, and M. Danelutto, "High-Performance Serverless Computing: A Systematic Literature Review on Serverless for HPC, AI, and Big Data," *IEEE Access*, vol. 13, pp. 195611–195656, 2025, doi: 10.1109/ACCESS.2025.3633989.

[26] S. Muvva, "Cloud-Native Data Engineering: Leveraging Scalable, Resilient, and Efficient Pipelines for the Future of Data," *ESP J. Eng. \& Technol. Adv.*, vol. 1, no. 2, pp. 287–292, 2021.

[27] C. K. Gomathy, V. R. K. E. C, and S. Ghiridharan, "Big Data Computing : Architectures , Technologies , and Future Perspectives," *Int. J. Sci. Res. Eng. Trends Vol. 11, Issue 6, Nov-Dec-2025, ISSN 2395-566X*, vol. 11, no. 6, pp. 1–8, 2025.

[28] B. F. More and S. Pawar, "Smart Home System using IoT and AI," *Int. J. Manag. Technol. Eng.*, vol. 8, no. XI, pp. 2241–2246, 2018.

[29] S. Bhat, S. R. Sirikonda, V. Katoch, and R. Jain, "Carbon-Kube: A Kubernetes-Native Framework for Multi-Objective Carbon-Aware Scheduling of Big Data Pipelines," in *2026 9th International Conference on Electronics, Materials Engineering & Nano-Technology (IEMENTech)*, 2026, pp. 1–6. doi: 10.1109/IEMENTech202669403.2026.11434192.

[30] H. Raza, T. Erdenetsogt, A. Singh, M. Farooq, M. M. Kabeer, and M. S. Aslam, "A Comprehensive Review on Data Science Frameworks for Big Data Analytics," *PERFECT J. Smart Algorithms*, vol. 3, no. 1, pp. 1–10, 2026, doi: 10.62671/perfect.v3i1.217.

[31] W. Elouataoui and Y. Gahi, "Rethinking Big Data Value Chains: A Comparative Framework Across Hadoop, Modern, and Cloud Stacks," in *2025 11th International Conference on Optimization and Applications (ICOA)*, 2025, pp. 1–6. doi: 10.1109/ICOA66896.2025.11236898.

[32] H. B. Abdalla, Y. Kumar, Y. Zhao, and D. Tosi, "A Comprehensive Survey of MapReduce Models for Processing Big Data," *Big Data Cogn. Comput.*, vol. 9, no. 4, 2025, doi: 10.3390/bdcc9040077.

[33] A. I. Abueid, "Big Data and Cloud Computing Opportunities and Application Areas," *Eng. Technol. Appl. Sci. Res.*, vol. 14, no. 3, pp. 14509–14516, Jun. 2024.

[34] S. Ponnusamy and P. Gupta, "Scalable Data Partitioning Techniques for Distributed Data Processing in Cloud Environments: A Review," *IEEE Access*, vol. 12, pp. 26735–26746, 2024, doi: 10.1109/ACCESS.2024.3365810.

[35] W. Qi, M. Sun, and S. R. A. Hosseini, "Facilitating big-data management in modern business and organizations using cloud computing: a comprehensive study," *J. Manag. Organ.*, vol. 29, no. 4, pp. 697–723, 2023, doi: 10.1017/jmo.2022.17.