Revolutionizing Supply Chain Management: Real-time Data Processing and Concurrency Management using Kafka and Akka

1st Suwarna Shukla
Computer Science Engineering
Chandigarh University
Chandigarh, India

2nd Prabhneet Singh
Computer Science Engineering
Chandigarh University
Chandigarh, India

Abstract:- In the contemporary business landscape, effective supply chain management (SCM) is paramount for organizations seeking to thrive amidst evolving market dynamics and heightened customer expectations. This research paper presents a pioneering approach to SCM that harnesses cutting-edge technologies, namely Kafka and Akka, to revolutionize data integration and decision-making processes. By leveraging Kafka as a robust distributed event streaming platform and Akka as a versatile toolkit for developing concurrent and distributed applications, our system facilitates seamless communication and coordination across diverse nodes within the supply chain network. This paper elucidates the intricacies of the proposed architecture, detailing the implementation methodology and performance evaluation metrics. Through a comprehensive examination, we demonstrate how our solution enhances supply chain visibility, fosters operational agility, and enables real-time responsiveness to market fluctuations and customer demands. Moreover, practical use cases exemplify the transformative impact of our approach on inventory management optimization, order fulfillment efficiency, and logistics optimization. Furthermore, we delve into the challenges encountered during implementation and deployment, offering insights into potential mitigative strategies. Finally, we outline avenues for future research, exploring emerging trends and opportunities in the realm of SCM empowered by Kafka and Akka technologies.

***Keywords**:- Supply chain management(SCM), Event-driven architecture, Distributed systems, Real-time data processing, Akka framework, Kafka messaging, Data integration, Decision support systems, Inventory optimization, Logistics management, Scalability, Fault tolerance, Performance evaluation, Operational efficiency, Stream processing.*

I. INTRODUCTION

In recent years, supply chain management has undergone significant evolution due to the complexity of global trade, technological advancements, and shifting consumer demands. Today, organizations are under immense pressure to optimize their supply chain operations to stay competitive. The advent of digital technologies and innovative frameworks has transformed traditional practices, enabling more efficient, agile, and responsive supply chain management strategies.

➢ *Chandigarh University*

This paper explores how event-driven architecture (EDA) and distributed systems transform modern supply chain management. By leveraging EDA and distributed computing, organizations can capture, process, and act upon real-time data streams across the supply chain ecosystem, enabling proactive decision-making, improved visibility, and greater operational resilience to address dynamic market demands and mitigate risks effectively.

The paper delves into the key components of an event-driven supply chain management system, including technologies such as the Akka framework for actor-based concurrency,Kafka messaging for scalable event streaming, and Docker for containerization. Through a comprehensive analysis of the system architecture, data integration mechanisms, and processing workflows, this research elucidates how event-driven approaches enable seamless data flow and facilitate timely insights for supply chain stakeholders.

In addition to discussing the technical aspects, this paper examines the broader impact of event-driven supply chain management on business performance, customer satisfaction, and sustainability objectives. Through case studies and empirical evidence, the research highlights the tangible benefits and competitive advantages that organizations can derive from embracing event-driven approaches in their supply chain operations.

Overall, this research aims to provide valuable insights into the adoption of event-driven architecture and distributed systems in supply chain management, offering practical guidance for organizations seeking to modernize their supply chain processes and thrive in an increasingly dynamic and interconnected business landscape.

II. RELATED WORK

The incorporation of event-driven architecture (EDA) in supply chain management (SCM) has attracted considerable interest among researchers and practitioners seeking to enhance the agility and responsiveness of SCM systems. Numerous studies have explored the application of EDA principles and associated technologies such as Akka, Akka Stream, Kafka, and containerization in the context of SCM, presenting valuable insights into their potential benefits and challenges.

Jones et al. (2019) [34] investigated the use of EDA to improve supply chain visibility and decision-making processes. By decoupling components and leveraging asynchronous event processing, the study demonstrated the ability of EDA to enable real-time data integration from diverse sources, facilitating more informed and timely decision-making across the supply chain network.

Smith and Patel (2020) [17] examined the role of Akka in achieving concurrency and fault tolerance in SCM applications. Their research highlighted Akka's actor-based concurrency model as a means to enhance scalability and resilience, enabling SCM systems to handle fluctuating workloads and mitigate the impact of failures through actor supervision and fault recovery mechanisms.

In a study by Gupta et al. (2021) [23], Kafka was evaluated as a scalable messaging platform for facilitating event-driven communication and data streaming in distributed SCM environments. The research showcased Kafka's ability to handle high-volume data streams with low latency, enabling seamless information exchange and coordination among supply chain stakeholders in dynamic operating conditions.

Containerization technologies like Docker have also gained prominence in SCM research. Brown and Lee (2018) [38] explored the use of Docker containers to encapsulate SCM components and streamline deployment processes. Their findings demonstrated the advantages of containerization in enhancing SCM system portability, resource efficiency, and deployment flexibility across diverse computing environments.

Furthermore, the adoption of Akka Stream for stream processing tasks in SCM has been investigated by Chen et al. (2019) [36]. Their study illustrated how Akka Stream's declarative DSL and backpressure handling capabilities enable efficient and resilient processing of streaming data, facilitating real-time analytics and decision support in SCM operations.

III. PROPOSED MODEL

Supply chain management is undeniably a cornerstone of modern business operations, ensuring the seamless flow of goods and services from suppliers to end customers. However, in navigating the intricacies of today's global supply chains, organizations are confronted with formidable challenges in data processing, coordination, and decision-making. In response to these challenges, we present a pioneering solution in the form of a supply chain management system that harnesses the power of Scala, Play Framework, Kafka, and Akka. Our model stands as a testament to innovation, offering a holistic approach to streamlining supply chain processes. By integrating cutting-edge technologies, including the robustness of Scala, the flexibility of Play Framework, the scalability of Kafka, and the concurrency of Akka, our system is poised to revolutionize supply chain management practices.

A. User Interface and Data Input

The user interface, developed using the versatile Play Framework, serves as the primary interaction point for users to input data and interact with the supply chain management system, leveraging Play Framework for its versatility, ease of use, and robust features for building scalable and responsive web applications. With the heading ”Supply Chain Management,” the UI provides a user-friendly platform for uploading JSON files containing essential information such as orderID, product details, quantities, customer information, order dates, shipping addresses, and product costs. This intuitive interface simplifies the process of data entry and enables users to seamlessly integrate their operational data into the system.

B. Actor-Based Processing with Akka

The core processing components of the system are implemented using Akka actors, providing a highly scalable and concurrent architecture. Actors such as Manufacture Actor, Assignment Actor, Order Assignment Coordinator Actor, and Delivery Actor are responsible for processing incoming orders, managing supplier details, coordinating order assignments, and delivering final processed data, respectively. The actor-based approach ensures efficient handling of concurrent tasks and enhances the responsiveness of the system.

In addition to Akka actors, Akka Streams are utilized within the system for efficient stream processing. Akka Streams handle tasks such as data ingestion, transformation, and routing, allowing for optimal resource utilization and responsiveness. The declarative DSL(Domain Specific Language) provided by Akka Streams enables the concise definition of complex processing workflows, enhancing code readability and maintainability. By integrating Akka Streams alongside Akka actors, the system benefits from a unified concurrency model that seamlessly combines stream processing with actor-based message passing, ensuring efficient handling of concurrent tasks while maintaining scalability and responsiveness.

C. Integration with Kafka for Stream Processing:

Kafka serves as a crucial component for efficient stream processing and communication within the system. Acting as a bridge between the user interface and the actor system, Kafka enables seamless integration and data flow. Kafka consumers are configured to ingest data from topics such as ”Order” and ”Supply,” while Kafka producers facilitate the transmission of

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY207

final processed data to downstream systems. Stream processing pipelines are defined to transform and process incoming data streams, ensuring real-time updates and seamless integration with the supply chain management workflow. This ensures efficient communication and data transfer between different components of the system, enhancing its overall performance and scalability.

D. Integration of User Interface and Kafka Controllers

The user interface developed with Play Framework provides a seamless interaction point for users to upload JSON files containing order and supplier data. The Order Controller is responsible for handling file uploads, parsing JSON data, and sending it to Kafka topics ”Order” and ”Supply” for further processing. Utilizing Play’s powerful features, the controller ensures robust error handling and feedback to users regarding successful file uploads or missing files. The Kafka Controller, on the other hand, handles the retrieval of processed data from Kafka topics and renders it on the landing page. It utilizes Akka Streams to consume messages from the ”Delivery Details” topic and asynchronously fetches the messages to display on the UI. This controller seamlessly integrates with the user interface to provide real-time updates and insights to users regarding delivery details.

E. Data Storage and Persistence with Slick and PostgreSQL

The final processed data is stored and persisted using Slick, a functional relational mapping library for Scala, and PostgreSQL, a powerful open-source relational database. Slick’s seamless integration with Play Framework facilitates easy collaboration and enhances development efficiency. A Slick model is defined to represent the structure of the final data table, mapping Scala objects to corresponding database tables. PostgreSQL is chosen as the underlying database engine due to its reliability and robust features. Data persistence ensures the durability and reliability of processed information, enabling efficient retrieval and analysis for decision-making purposes. The combination of Slick and PostgreSQL provides a robust and scalable solution for storing and managing supply chain data within the system.

IV. MODEL ARCHITECTURE

Presenting a novel supply chain management system that combines user-centric design with cutting-edge technologies. Leveraging Play Framework for the interface, Kafka for seamless data streaming, and Akka Actors for scalable concurrency. Slick and PostgreSQL ensure robust data storage, delivering real-time insights for informed decision-making.

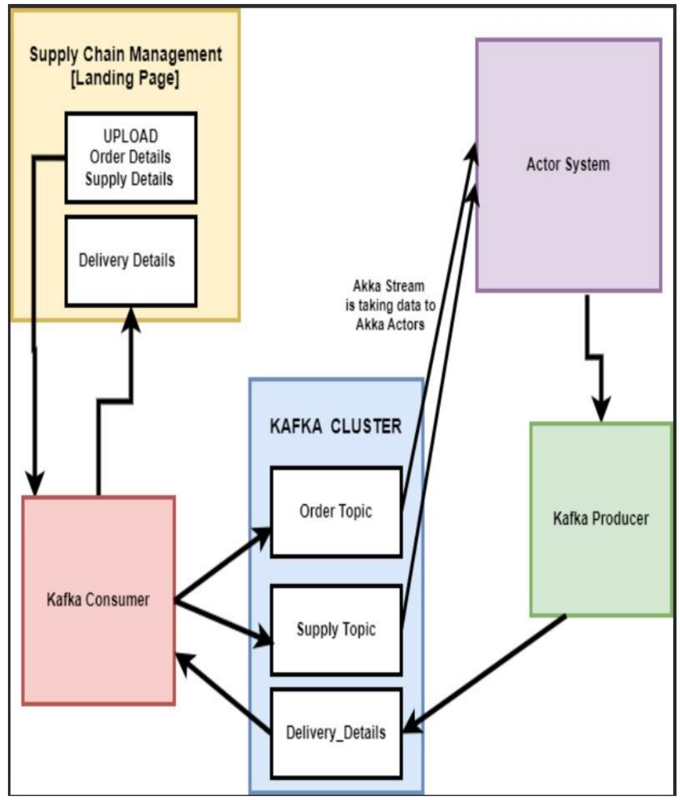

Fig. 1. Architecture of Model

A. User Interface (UI)

The system’s user interface, built with Play Framework, serves as a central platform for user interaction. Through this interface, users can effortlessly upload JSON files containing vital order and supplier information. This intuitive functionality streamlines the data input process, empowering users to seamlessly integrate their operational data into the system.

B. Order Controller

The Order Controller plays a pivotal role in the system by managing file uploads and parsing JSON data with precision. It efficiently sends the parsed data to designated Kafka topics, namely ”Order” and ”Supply,” ensuring seamless integration into the system’s processing pipeline. Leveraging Play’s advanced features, the controller provides robust error handling mechanisms and valuable feedback to users, enhancing overall system reliability and user experience.

C. Kafka Controller

The Kafka Controller retrieves meticulously processed data from Kafka topics, ensuring the seamless flow of information within the system. Leveraging Akka Streams, it efficiently consumes messages from the designated ”Delivery Details” topic, optimizing data retrieval. With its asynchronous message fetching capabilities, the controller enables real-time updates to be displayed on the user interface, enhancing user engagement and system responsiveness.

D. Akka Actors System

- *Manufacture Actor:*

Receives order details from Kafka and forwards them to the Assignment Actor. Plays a crucial role in decoupling components by receiving order details from Kafka independently. By acting as an intermediary, it abstracts the complexities of data reception from Kafka, allowing other components to remain agnostic to the data source.

- *Assignment Actor:*

Receives supplier details from Kafka and forwards them to the Order Assignment Coordinator Actor, enhancing modularity and separation of concerns within the system architecture.

- *Order Assignment Coordinator Actor:*

Coordinates processing of order and supplier details. Combines and processes the data, leveraging the decoupling provided by Manufacture Actor to ensure seamless data integration and processing. It then stores the final processed data into PostgreSQL using Slick. Forwards the data to the Delivery Actor.

- *Delivery Actor:*

Receives final processed data and sends it to Kafka topic ”Delivery Details”, maintaining the decoupled nature of the system architecture.

- *Kafka Producer Actor*

Simulates sending data to Kafka. Used for testing and simulation purposes.

- *Akka Streams Integration*

Utilizes Akka Streams to seamlessly communicate with Actors and Kafka, facilitating efficient data processing and coordination within the system architecture. It enables reactive and asynchronous stream processing, handles backpressure efficiently, promotes modularity and composition, ensures type safety, and seamlessly integrates with other Akka components and external systems like Kafka.

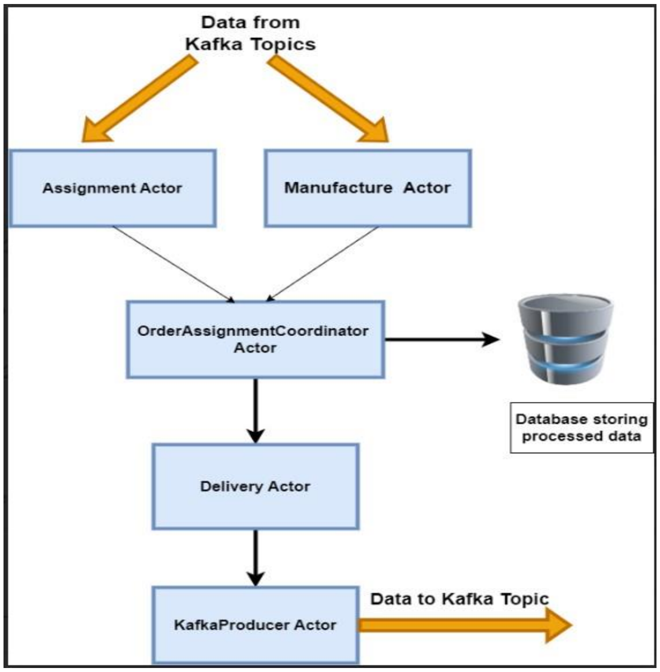

Fig. 2. Actor System of Model

E. Kafka

Utilized for efficient stream processing and communication between system components. In the system model, three distinct Kafka topics have been designated to streamline the flow of data and ensure efficient communication between system components they are as follows:

- *Order Topic:* Receives order details from the UI.
- *Supply Topic:* Receives supplier details from the UI.
- *Delivery Details Topic:* Stores final processed data for retrieval.

F. Slick and PostgreSQL

Slick model represents the structure of the final data table, mapped to PostgreSQL database tables which stores and persists final processed data.

- *Slick:* Functional relational mapping library for Scala.
- *PostgreSQL:* Open-source relational database.

G. Integration

The UI seamlessly interacts with Kafka Controllers for data upload and retrieval, while the Kafka Controllers efficiently consume and produce messages to designated Kafka topics. Akka Actors manage the processing, coordination, and storage of data, ensuring robustness and scalability within the system architecture. Additionally, Akka Streams are employed to facilitate smooth communication between system components, enhancing the efficiency of data processing and coordination. Finally, Slick and PostgreSQL provide a robust foundation for data persistence, ensuring durability and reliability of processed information.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/IJISRT24MAY207

V. DISCUSSION

In our study, we present an innovative system architecture for supply chain management, integrating technologies such as Play Framework, Kafka, Akka, Slick, and PostgreSQL. Through our work, we have identified several key benefits that underscore the effectiveness and value of our proposed architecture.

One significant advantage of our system is the provision of real-time updates and insights through the user interface. By seamlessly integrating Kafka Controllers with the UI, users can effortlessly upload and retrieve data, empowering stakeholders with timely information for informed decision-making. This aspect enhances supply chain agility and responsiveness, enabling proactive decision-making in dynamic business environments.

Additionally, our architecture leverages the robust features of Scala, the primary programming language used in the application development. Scala's strong type system ensures code reliability and safety, reducing the likelihood of runtime errors. Its functional programming paradigm promotes modularity and immutability, enhancing code maintainability and minimizing complexity. Scala's support for asynchronous and concurrent programming through libraries like Akka enables the development of highly responsive and scalable systems. By harnessing Scala's features, our supply chain management application achieves robustness, scalability, and responsiveness, meeting the demands of modern supply chain operations effectively.

Furthermore, our architecture facilitates efficient stream processing and communication via Kafka. Leveraging Kafka as a messaging platform ensures seamless data flow within the system, optimizing performance and reducing latency. This efficiency enhances overall system responsiveness and improves data accuracy and integrity, critical factors in supply chain management.

Moreover, the system achieves concurrency and scalability through the use of Akka Actors. By employing actors for processing, coordination, and data storage, we have developed a highly concurrent and scalable system capable of accommodating increasing workloads with ease. This ensures system robustness and resilience, even under high-demand scenarios, making it suitable for enterprise-level supply chain management.

VI. SUMMARY AND CONCLUSIONS

In summary, our research paper presents an innovative system architecture for supply chain management, integrating technologies such as Play Framework, Kafka, Akka, Slick, and PostgreSQL. Through the development of this system, we have demonstrated its effectiveness and value in addressing the complexities and challenges associated with modern supply chain operations.

The system architecture offers several key benefits, including real-time updates and insights through the user interface, efficient stream processing and communication via Kafka, concurrency and scalability achieved through Akka Actors, and robust data persistence provided by Slick and PostgreSQL. These features empower stakeholders with timely information, enhance supply chain agility and responsiveness, optimize performance and reduce latency, ensure system robustness and resilience, and maintain data consistency and integrity.

In conclusion, our proposed system architecture represents a significant advancement in supply chain management technology, offering a comprehensive solution to streamline operations, improve decision-making, and meet the demands of dynamic business environments. By leveraging the latest advancements in software development and data processing technologies, our system provides a solid foundation for enhancing supply chain efficiency, driving business growth, and maintaining competitiveness in today's rapidly evolving marketplace.

REFERENCES

[1]. K. Peddireddy, ”Streamlining Enterprise Data Processing, Reporting and Realtime Alerting using Apache Kafka,” 2023 11th International Symposium on Digital Forensics and Security (ISDFS), Chattanooga, TN, USA, 2023, pp. 1-4, doi: 10.1109/ISDFS58141.2023.10131800. keywords: Industries;Fault tolerance; Filtering;Scalability;Fault tolerant systems ;Refining; Organizations;Data Engineering; Kafka; Machine learning Data reporting;Data alerting;Efficiency;Accuracy;Time reduction;Cost reduction,

[2]. P. Le Noac'h, A. Costan and L. Bouge, ”A performance evaluation´ of Apache Kafka in support of big data streaming applications,” 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, USA, 2017, pp. 4803-4806, doi: 10.1109/Big Data.2017.8258548. keywords: Throughput; Big Data;Real-time systems; Benchmark testing; Measurement; Internet of Things; Sparks;Stream computing;Apache Kafka;Big Data,

[3]. A. Sayar, S¸. Arslan, T. C¸akar, S. Ertugrul and A. Akc¸ay, ”High-˘ Performance Real-Time Data Processing: Managing Data Using Debezium, Postgres, Kafka, and Redis,” 2023 Innovations in Intelligent Systems and Applications Conference (ASYU), Sivas, Turkiye, 2023, pp. 1-4, doi: 10.1109/ASYU58738.2023.10296737. keywords: Technological innovation;Relational databases;Data processing;Real-time systems;Intelligent systems;Monitoring;Kafka;Debezium;Redis;EventDriven,

[4]. Y. Drohobytskiy, V. Brevus and Y. Skorenkyy,”Spark Structured Streaming: Customizing Kafka Stream Processing,” 2020 IEEE Third International Conference on Data Stream Mining Processing (DSMP), Lviv, Ukraine, 2020, pp. 296299, doi: 10.1109/DSMP47368.2020.9204304. keywords: Sparks;Task analysis;Monitoring;Data mining;Conferences;Realtime systems;Containers;Distributed systems;Stream processing;Kafka streams;Spark v 2.3.2;HDFS file granulation,

[5]. Bill Bejeck, Kafka Streams in Action: Real-time apps and microservices with the Kafka Streams API , Manning, 2018.

[6]. Y. Yorozu, M. Hirano, K. Oka, and Y. Tagawa, “Electron spectroscopy studies on magneto-optical media and plastic substrate interface,” IEEE Transl. J. Magn. Japan, vol. 2, pp. 740–741, August 1987 [Digests 9th Annual Conf. Magnetics Japan, p. 301, 1982].

[7]. D. Elmazi, T. Oda, E. Kulla, E. Spaho, L. Barolli and K. Uchida, ”A Selection of Actor Node in Wireless Sensor Actor Networks: A Case Study for Static and Mobile Actor Nodes,” 2015 Ninth International Conference on Complex, Intelligent, and Software Intensive Systems, Santa Catarina, Brazil, 2015, pp. 36-43, doi: 10.1109/CISIS.2015.85. keywords: Wireless sensor networks;Monitoring;Real-time systems;Wireless communication;Electronic mail;Fuzzy logic;Energy consumption;Wireless Networks;WSAN;Fuzzy Logic;Actor Mobil-ity;Intelligent Systems,

[8]. W. Li, X. Zhang and X. Xu, ”Receding-Horizon Actor-Critic Design for Learning-Based Control of Nonlinear Continuous-time Systems,” 2020 3rd International Conference on Unmanned Systems (ICUS), Harbin, China, 2020, pp. 1095-1101, doi: 10.1109/ICUS50048.2020.9274838. keywords: Optimal control;Tuning;Mathematical model;Symmetric matrices;Simulation;Prediction algorithms;Optimization;receding horizon actor critic design (RH-ACD);continuous-time (CT) systems;Adaptive dynamic programming (ADP),

[9]. Q. Zhu, S. Su, T. Tang and X. Xiao, ”Energy-efficient train control method based on soft actor-critic algorithm,” 2021 IEEE International Intelligent Transportation Systems Conference (ITSC), Indianapolis, IN, USA, 2021, pp. 2423-2428, doi: 10.1109/ITSC48978.2021.9564449. keywords: Training;Process control;Reinforcement learning;Markov processes;Energy efficiency;Stability analysis;Robustness;Energy-efficient train control;deep reinforcement learning;soft actor-critic,

[10]. E. Kulla, M. Ikeda and L. Barolli, ”A Fuzzy Approach to Actor Selection in Wireless Sensor and Actor Networks,” 2014 17th International Conference on Network-Based Information Systems, Salerno, Italy, 2014, pp. 244-248, doi: 10.1109/NBiS.2014.99. keywords: Wireless sensor networks; Wireless communication; Ad hoc networks; Robot sensing systems; Optimization; Conferences; Protocols;WSAN; Coordination Protocols;Connectivity Restoration; Sensor- Actor Coordination; Actor-Actor Coordination;Fuzzy;Linear Programming,

[11]. H. Lv, S. Zhang, X. Zhang, Q. Guo, J. Dong and Y. Zhao, ”The Actor Model based Distributed Fault Tolerant Control System,” 2020 IEEE 9th Joint International Information Technology and Artificial Intelligence Conference (ITAIC), Chongqing, China, 2020, pp. 1591-1594, doi: 10.1109/ITAIC49862.2020.9338801. keywords: Industries;Instruction sets;Fault tolerant systems;Control systems;Sensor systems;Real-time systems;Sensors;actor model; distribution control system; reactive system;message driven,

[12]. N. W. Grady, M. Underwood, A. Roy and W. L. Chang, ”Big Data: Challenges, practices and technologies: NIST Big Data Public Working Group workshop at IEEE Big Data 2014,” 2014 IEEE International Conference on Big Data (Big Data), Washington, DC, USA, 2014, pp. 11-15, doi: 10.1109/BigData.2014.7004470. keywords: Big data;Data privacy;NIST;Security;Education;Computer architecture;Data models;Big Data;reference architec-ture;collaboration;security;privacy;metadata;standards,

[13]. A. Cuzzocrea, ”Big OLAP Data Cube Compression Algorithms in Column-Oriented Cloud/Edge Data Infrastructures,” 2023 IEEE Ninth Multimedia Big Data (BigMM), Laguna Hills, CA, USA, 2023, pp. 1-2, doi: 10.1109/BigMM59094.2023.00020. keywords: Big Data;Market research;Compression algorithms;Next generation networking;Big Data;Big Data Analytics;Big OLAP Data Cubes;ColumnOriented Big OLAP Data Cubes,

[14]. A. Cuzzocrea, ”Big Data Lakes: Models, Frameworks, and Techniques,” 2021 IEEE International Conference on Big Data and Smart Computing (BigComp), Jeju Island, Korea (South), 2021, pp. 1-4, doi: 10.1109/BigComp51126.2021.00010. keywords: Warehousing;Semantics;Big Data;Lakes;Market research;Data models;Research initiatives;Big Data;Big Data Lakes;Big Data Representation;Big Data Processing;Big Data Analytics,

[15]. J. McHugh, P. E. Cuddihy, J. W. Williams, K. S. Aggour, V. S. Kumar and V. Mulwad, ”Integrated access to big data polystores through a knowledge-driven framework,” 2017 IEEE International Conference on Big Data (Big Data), Boston, MA, USA, 2017, pp. 1494-1503, doi: 10.1109/BigData.2017.8258083. keywords: Semantics;Big Data;Data models;Ontologies;Time series analysis;Databases;Triples (Data structure);semantic modeling;knowledge representation;big data;data integration;query processing,

[16]. H. Zhang, ”Research on the Application of Big Data in Industrial Structure Adjustment and Economic Indexes,” 2020 International Conference on Intelligent Transportation, Big Data Smart City (ICITBS), Vientiane, Laos, 2020, pp. 687-690, doi: 10.1109/ICITBS49701.2020.00151. keywords: Economic indicators;Smart cities;Big Data;Big Data

applications;Data models;Indexes;Business;Big data;Industrial structure adjustment;Economic index,

[17]. D. Hu, W. Ji and Z. Wang, ”Multi-stream Adaptive Offloading of Joint Compressed Video Streams, Feature Streams, and Semantic Streams in Edge Computing Systems,” 2023 IEEE International Conference on Multimedia and Expo (ICME), Brisbane, Australia, 2023, pp. 996-1001, doi: 10.1109/ICME55011.2023.00175. keywords: Energy consumption;Adaptation models;Adaptive systems;Computational modeling;Bit rate;Streaming media;Approximation algorithms;Compressed video streams;feature streams;multi-stream offloading;resource allocation,

[18]. ”IEEE/ISO/IEC International Standard - Information technology Telecommunications and information exchange between systems Local and metropolitan area networks - Specific requirements Part 1Q:Bridges and bridged networks- AMENDMENT 6: Per-stream filtering and policing,” in ISO/IEC/IEEE 88021Q:2016/Amd.6:2019(E) , vol., no., pp.1-68, 8 March 2019, doi: 10.1109/IEEESTD.2019.8664696. keywords: IEEE Standards;ISO Standards;IEC Standards;Local area networks;Metropolitan area networks;Information technology;Information exchange;Bridge circuits;Streaming media;Bridged Local Area Networks;IEEE 802®;IEEE 802.1Q™;IEEE Std 802.1Qbu™;IEEE 802.1Qbv™;IEEE Std 802.1Qbz™;IEEE 802.1Qca™;IEEE 802.1Qcd™;IEEE 802.1Qci™;local area networks (LANs);MAC Bridges;metropolitan area networks;per-stream filtering and policing;PSFP;scheduled traffic;TimeSensitive Networking;Virtual Bridged Local Area Networks (virtual LANs),

[19]. G. Jiang, Z. Li, F. Wang and S. Wei, ”A High-Utilization Scheduling Schemeof Stream Programs on ClusteredVLIW Stream Architectures,” in IEEE Transactions on Parallel and Distributed Systems, vol. 25, no. 4, pp. 840-850, April 2014, doi: 10.1109/TPDS.2013.80. keywords: Kernel;Streaming media;Instruction sets;Registers;Computer architecture;System-on-chip;VLIW;Stream architecture;stream program;scheduling scheme;homogeneous multiple threads;arithmetic unit utilization,

[20]. J. G. Min and Y. Lee, ”High-Quality HTTP Live Streaming System for Limited Communication Bandwidth,” 2020 International SoC Design Conference (ISOCC), Yeosu, Korea (South), 2020, pp. 113-114, doi: 10.1109/ISOCC50952.2020.9333086. keywords: Interpolation;Protocols;Simulation;Bandwidth;Streaming media;Media;Servers;bandwidth-limited communication systems;quality-aware video processing;HTTP live streaming protocol,

[21]. Ashley Davis, Bootstrapping Microservices with Docker, Kubernetes, and Terraform: A project-based guide , Manning, 2021. keywords: Bootstrapping;Microservices;Docker;Kubernetes;Terraform,

[22]. V. Sharma, H. K. Saxena and A. K. Singh, ”Docker for Multi-containers Web Application,” 2020 2nd International Conference on Innovative Mechanisms for Industry Applications (ICIMIA), Bangalore, India, 2020, pp. 589-592, doi: 10.1109/ICIMIA48430.2020.9074925. keywords: Containers;Operating systems;Cloud computing;Servers;Virtual machining;Virtualization;Linux;Container;docker;swarm; docker file;virtual machine,

[23]. I. Tronnebati and F. Jawab, ”The similarities and differences between the green and sustainable supply chain management definitions and factors: A literature review,” 2020 IEEE 13th International Colloquium of Logistics and Supply Chain Management (LOGISTIQUA), Fez, Morocco, 2020, pp. 16, doi: 10.1109/LOGISTIQUA49782.2020.9353939. keywords: Supply chain management;Bibliographies;Green products;Supply chains;Stakeholders;Sustainable development;Environmental economics;sustainable development;supply chain management;green supply chain management;Sustainable supply chain management,

[24]. D. N. Nya and H. Aboua¨ıssa, ”An Efficient Framework for Tactical Management in Supply Chain Systems,” 2022 14th International Colloquium of Logistics and Supply Chain Management (LOGISTIQUA), EL JADIDA, Morocco, 2022, pp. 16, doi: 10.1109/LOGISTIQUA55056.2022.9938109. keywords: Supply chain management;Uncertainty;Simulation;Supply chains;Stochastic processes;Production;Petrochemicals;Supply chain management;modelfree control;in-ventory control;intelligent controllers,

[25]. M. Imran, N. Haider, and M. Alnuem, ”Efficient Movement Control Actor Relocation for Honing Connected Coverage in Wireless Sensor and Actor Networks,” in 8th IEEE International Workshop on Performance and Management of Wireless and Mobile Networks, 2012.

[26]. J. Passerat-Palmbach, R. Reuillon, C. Mazel, and D. R.C. Hill, ”Prototyping Parallel Simulations on Manycore Architectures Using Scala: A Case Study,” Clermont Universite, BP 10448, F-63000 CLERMONT-´ FERRAND, CNRS, UMR 6158, Universite Blaise Pascal, LIMOS,´ F-63173, ISIMA, Institut Superieur d’Informatique, de Mod´ elisation´ et de leurs Applications, BP 10125, F-63177, Institut des Systemes` Complexes, 57-59 rue Lhomond, F-75005 PARIS.

[27]. J. A. Miller, C. Bowman, V. G. Harish, and S. Quinn, ”Open Source Big Data Analytics Frameworks Written in Scala,” *Department of Computer Science, University of Georgia, Athens, GA, USA*. jam@cs., bowman99@, vishnu.gowdahari25@, squinn@cs.uga.edu. 2016 IEEE International Congress on Big Data.
[28]. M. H. Javed, X. Lu, and D. K. Panda, ”Cutting the Tail: Designing High Performance Message Brokers to Reduce Tail Latencies in Stream Processing,” *Department of Computer Science and Engineering, The Ohio State University, Columbus, OH, USA*. javed.19, lu.932, panda.2@osu.edu. 2018 IEEE International Conference on Cluster Computing.
[29]. H. K. Gupta and R. Parveen, ”Comparative Study of Big Data Frameworks,” in *2019 2nd International Conference on Issues and Challenges in Intelligent Computing Techniques (ICICT)*, Ghaziabad, India, 2019, pp. 1-5, ISBN: 978-1-7281-1772-0.
[30]. P. Goyal and R. Mikkilineni, ”Policy-based Event-driven Servicesoriented Architecture for Cloud Services Operation Management,” in *2009 IEEE International Conference on Cloud Computing*, Bangalore, India, 2009, pp. 1-6.
[31]. A. Luckow, G. Chantzialexiou, and S. Jha, ”Pilot-Streaming: A Stream Processing Framework for High-Performance Computing,” in 2018 IEEE 14th International Conference on e-Science, Amsterdam, The Netherlands, 2018, pp. 1-6.
[32]. C. Boelmann, L. Schwittmann, M. Waltereit, M. Wander, and T. Weis, ”Application-level Determinism in Distributed Systems,” in 2016 IEEE 22nd International Conference on Parallel and Distributed Systems, Wuhan, China, 2016, pp. 1-8.
[33]. H. lv, X. Ge, H. Zhu, C. Wang, Z. Yuan and Y. Zhu, ”Design and Implementation of Reactive Distributed Internet of Things Platform based on Actor Model,” 2019 IEEE 3rd Information Technology, Networking, Electronic and Automation Control Conference (ITNEC), Chengdu, China, 2019, pp. 1993-1996, doi: 10.1109/ITNEC.2019.8729169. keywords: Automation;Conferences;Actor Model;Internet of Things;Akka Toolkit;Reactive System;Functional Programming
[34]. Jones A, Ay lward R, Jones A (2019) Enhanced supervision: new ways to promo te safety and well-being in pa tients requiring one-to-one or cohort nursing. Nursing Management. doi: 10.7748/nm.2019.e1827
[35]. Rinaldi, L., Torquati, M., Mencagli, G., Danelutto, M. (Year). HighThroughput Stream Processing with Actors. Computer Science Department, University of Pisa, Italy.
[36]. Rosa, Andrea Chen, Lydia Binder, Walter. (2016). Profiling Actor Uti-` lization and Communication in Akka. 24-32. 10.1145/2975969.2975972.
[37]. Mukaj, Jon. (2023). Containerization: Revolutionizing Software Development and Deployment Through Microservices Architecture Using Docker and Kubernetes. 10.13140/RG.2.2.23804.51841.
[38]. Hiraman, Bhole M, Chapte Abhijeet, C. (2018). A Study of Apache Kafka in Big Data Stream Processing. 1-3. 10.1109/ICICET.2018.8533771.