NextGen Scientific Publication
American International Journal of Computer Science and Technology

Volume 8 Issue 5, Pg. No. 1-14, AIJCST-V8I5P101, 2026
https://doi.org/10.63282/3117-5481/AIJCST-V8I5P101

Original Article

From Job Site to Insight: Evolving Data Pipelines for LLM-Driven Construction Analytics

***Shobhit Gupta**

VP Software Engineering, Stanley Black and Decker, CA, USA.

Abstract:

With advances in construction, such as those from IoT sensors, equipment, BIM systems, RFIs, inspections, safety logs, field communication and project documents, a massive amount of data is created during a project. While both Apache Kafka and Spark work well with structured and semi-structured data, they lack the ability to interpret unstructured text, which impedes timely decision-making, coordination, risk detection, and project optimization. Setting the groundwork for a pipeline that transforms Kafka's data into operational insights in real time, powered by an LLM based on RAG and Spark. The architecture provides for automated analysis of all types of RFIs, field reports, safety observations and communications and leverages lakehouse technologies and tenant-aware retrieval to provide secure SaaS deployment. The proposed framework offers enhanced processing times, contextual accuracy and insight generation, and decision support, when compared to conventional pipelines, by leveraging automated knowledge extraction and secure and contextually relevant analytics.

Keywords:

Construction Analytics, Data Pipelines, Large Language Models, Kafka, Apache Spark, Retrieval-Augmented Generation, Data Lakes, SaaS, Predictive Safety, Automated RFI Resolution.

Article History:

Received: 20.07.2026

Revised: 19.08.2026

Accepted: 27.08.2026

Published: 03.09.2026

1. Introduction

The construction industry is going through a period of great change with the increasing use of Internet of Things (IoT) devices, Building Information Modeling (BIM), cloud-based project management software, mobile apps, equipment monitoring, and automated reporting. [1] These technologies, which operate continuously and produce large amounts of 'messy' data during the various stages of the building project, involve equipment telemetry, sensor measurements, inspection records, observations of safety, Requests for Information (RFIs), field reports, emails, photographs and project documentation. Although these data create a tremendous opportunity for project analytics, their volume and variety present tremendous challenges to traditional data analytics systems for project monitoring, risk management, resource utilization, safety management and decision making. There are scalable solutions for high-volume data ingestion and distributed processing in the structured data world with Apache Kafka and Apache Spark, however this world is tailored for structured and semi-structured data. This has made it challenging to access and assimilate critical information that exists in the context of RFIs, field communication, safety narratives, inspection comments, and technical documents, into operational processes. The constraint pushes forward the development of construction data pipelines towards scalable data processing and integration of Large Language Models (LLMs), retrieval augmented generation (RAG), and enterprise knowledge sources to produce context-aware construction intelligence.

The construction data bottleneck has arisen due to the mismatch in quantity of information created at modern job sites, and a lack of ability to derive meaningful, context-based data from such information in an outmoded pipeline. While Kafka and Spark are capable of efficiently ingesting, transforming, aggregating and analyzing large volumes of telemetry and operational events, processing

the semantic content of RFIs, field reports, safety observations, inspection narratives, emails, drawings and other project-related documents is limited. Often, multiple sources of information are required to make a construction decision, for instance, having combined relationships with project specifications, contracts, BIM information, previous RFIs and field observations to resolve an RFI; or having correlated field observations with equipment and historical incident, to evaluate a safety condition. The traditional pipelines are typically based on fixed schemas, rules and analytical transformations, and are not naturally equipped to provide semantic retrieval, context interpretation and generation of natural-language responses. Thus, the proposed research enriches the Kafka-Spark data foundation with RAG, reasoning using LLMs, and enterprise-tool integration, migrating the pipeline from data processing to context-aware semantic intelligence and creating actionable insights from disorganized construction information based on the evidence.

2. Background and Theoretical Foundations

2.1. Construction Data Ecosystems

Modern construction data ecosystems are sets of technologies, information feeds, and processes that continually produce and distribute project information across the entire construction lifecycle. [2] These ecosystems are composed of sensors, equipment monitoring systems, BIM platforms, project management applications, mobile apps for the field, inspection systems, scheduling applications, document repositories and communication platforms. The information that results, can be partitioned into three types: structured, semi-structured and unstructured, each having different processing needs. Conventional databases and distributed processing frameworks adequately serve the purpose of structured data like equipment measurements, schedules, work orders and resource records, while advanced processing based on semantics is needed for other records such as RFIs, field reports, safety observations, emails, drawings and technical documents. Therefore, building analytics demands a true data ecosystem to be able to ingest lots of events, process in a distributed fashion, store them persistently, retrieve them at a later date - using context, and interpret them intelligently. This disparate landscape is the foundation of new, more evolved data pipelines in construction integrating streaming technologies with semantic intelligence enabled by LLM capabilities.

2.2. Apache Kafka for Real-Time Data Streaming

Apache Kafka has complementary capabilities to Apache Spark for building scalable construction data pipelines: Kafka is a distributed event-stream platform; Spark is a distributed computation system for large-scale data processing. Distributed topics and partitions pave the way for Kafka to support high-velocity information being continuously delivered from IoT sensors, construction equipment, mobile applications, and various operational systems, ensuring reliable and scalable delivery. These streams can then be consumed by Spark to be used for data cleansing, data transformation and aggregation, feature extraction, and analytical computation in parallel processing environments. It's an ideal combination for construction applications that require the staggering amount of job site telemetry and operational events to be processed with low latency. Kafka and Spark have limitations, however, in interpreting the semantic meaning of unstructured information, while handling the volume, velocity, and scalability elements of construction data. This distinction is of particular architectural significance: in addition to the capabilities provided by Kafka and Spark—our high throughput data-processing foundation—the transformation of textual and contextual information into actionable construction intelligence demands additional semantic technologies.

2.3. Large Language Models for Enterprise Analytics

Large Language Models (LLMs) enhance traditional construction analytics through Natural-Language understanding, Information extraction, summarization, reasoning, and response generation, drawing from diverse construction information. [3] But without the use of documents pertaining to the project, using only a LLM can lead to stale information, lack of support, and hallucination of information. To overcome this problem, RLHR model was designed by adding an external retrieval mechanism to generate information relevant to the topic, and allocating the retrieved context to LLM inference. Within a construction analytics environment, RAG can pull key RFIs, project specifications, field reports, safety rules, inspection records, project decisions, and much more before providing an analytical response. This will allow for the pipeline's evolution from foot-to-foot analytics to situational, context-related intelligence, while still providing timely project data at all times. Then, coupled with streaming technologies like Kafka and Spark, LLM and RAG can add a layer of semantic intelligence to transform information from the job site in real time and project data that is not structured into evidence-based insights, recommendations, alerts, and automated decision support outputs.

3. Research Methodology and System Requirements

Table 1. Construction Data Characteristics

Data Source	Type	Processing
IoT Sensors	Structured	Streaming
Equipment Telemetry	Structured	Real-time
BIM	Semi-structured	Integration
RFIs	Unstructured	LLM/RAG
Field Reports	Unstructured	NLP/LLM
Safety Reports	Unstructured	Semantic Analysis
Project Documents	Unstructured	RAG

3.1. Research Methodology

The study uses design-oriented and experimental research to design and test an LLM-based construction analytics pipeline that can process many types of information found on the job site in real time. The methodology is divided into four main phases: characterization of data, architecture design, system implementation and experimental evaluation. First, construction data sources such as IoT sensors, equipment telemetry, BIM, RFIs, field reports, safety, inspection reports, communications and project documents are analyzed based on their structure, volume, velocity and semantic properties. [4] Secondly, an integrated architecture is built by integrating kafka-based event streaming, spark distributed processing, lakehouse storage, RAG based knowledge retrieval, LLM based reasoning and controlled enterprise tool integration. Third, the proposed pipeline is deployed and designed in the multi-tenant environment to be secured, isolated, and governed to address implementation challenges in complex environments. Finally the system is compared to the traditional data-processing methods and the performance of the system is measured based on ingestion throughput, processing latency, semantic accuracy, retrieval performance, resource utilization, system scalability, and task-level effectiveness. This approach will allow for a systematic evaluation of the feasibility of the proposed architecture for effectively integrating high-volume construction data with LLM-based intelligence in the context of real-time decision-making.

3.2. Non-Functional Requirements

The proposed construction analytics pipeline is required to meet a set of non-functional requirements which address aspects that make the pipeline suitable for large-scale, real-time and multi-tenant deployments. The architecture should handle a high volume of events at the job site with minimal latency and have an acceptable LLM and RAG response time. To be scalable, the number of construction projects, tenants, data sources, and analytical requests must be scalable horizontally, which means that they should be able to handle larger amounts of data.The number of construction projects, tenants, data sources, and analytical requests must scale horizontally as they grow, which means that they should be able to process more data. They need strict tenant isolation, authenticated access, encryption, protection to create secure context and protection from unauthorized retrieval or cross-tenant information leakage, to ensure security and privacy. Fault tolerant, message durable, recovery, and the ability to continue operation in the event of a failure in any single processing or inference component are essential properties of reliability. The retrieved context and the generated outputs must be relevant, grounded and traceable to authorised construction information for Semantic quality. The architecture should also be observability, maintainability, interoperability, and cost efficient, allowing organizations to ensure the health of the pipelines, integrate existing construction systems, scale with evolving workloads, and manage LLM-based analytics in enterprise SaaS use-cases sustainably.

4. Evolution of Construction Data Pipelines

4.1. IoT-Centric Kafka–Spark Architecture

Unlike traditional batch data approaches, the IoT-based construction data pipeline allows for continuous ingestion and analysis of the high velocity of job-site data, making it a major leap forward. [5] According to this architecture, the events generated by the equipment, sensor, mobile applications and operational systems are being telemetryed via scalable topics and partitions in Apache Kafka and then stream processed, transformed, aggregated, and analytically calculated in Apache Spark. The architecture is suitable for construction sites where equipment conditions, environmental measurements, events related to workers and activities must be monitored continuously. Data separation from data processing allows for horizontal scalability, fault tolerance and efficient handling of variable loads. The architecture, however, is still mostly geared towards structured and semi-structured data, where it is possible to effectively apply pre-defined schemas and computational transformations. The Kafka–Spark architecture thus sets the stage for

supporting the high throughput required in modern construction analytics however, it offers limited support for interpreting the semantic content of a RFI, field report, safety narrative, project communication and technical document.

4.2. Transition From Data Processing to Data Understanding

Designing new data processing systems requires a paradigm shift from traditional data processing to data understanding due to the growing amount of information provided in an unstructured format in the construction industry.In the construction industry, significant paradigm shift is needed from traditional data processing to data understanding given the increasing amount of information provided in an unstructured format. Traditional pipelines focus on tasks like ingestion, filtering, transformation, aggregation, and statistical analysis, and work well for numerical and event-based information but are not sufficient to effectively interpret the content of natural language text when it has context-specific meanings. Relationships between various sources of information are often necessary for construction decisions, e.g. an RFI might need to be related to project specifications, BIM information, past decisions, observed field conditions, and equipment conditions, etc. As the move towards data understanding, it adds semantic processing capabilities, such as recognizing entities, relationships, events, risks, as well as contextual dependencies among heterogeneous sources. [6] LLMs offer natural language reasoning and information extraction, and RAG technologies deliver the ability to retrieve pertinent project-specific knowledge prior to generating analytical outputs. Therefore, the data pipeline starts to use event processing to answer the question of “why did it happen?” and moves on to “what does it mean?” and “what action should be considered?” By making this transition, the conceptual groundwork is laid for combining LLM and RAG with existing streaming and distributed-processing technologies.

4.3. Proposed Pipeline Evolution Model

The pipeline evolution model proposes an evolution of the construction of data engineering, from lower intelligence levels of conventional data collection, distributed stream processing, to integrated data-lake processing, semantic enrichment, and finally, LLM-driven contextual intelligence. In the first stage, conventional systems gather and save project data from various isolated sources; in the second stage, Kafka and Spark allow for the ingestion and real-time processing of high volumes of project data from job sites; and lakehouse technologies allow for the storage and access to structured and unstructured construction data in the third stage. This fourth stage involves semantic processing, document indexing, vector representation and contextual retrieval to provide connections between these disparate bodies of project knowledge, with the final stage integrating RAG, LLM reasoning and controlled interaction with enterprise tools to create grounded insights, alerts, recommendations and automated responses. The architecture that is created as a result of this enables the preservation of the scalability and reliability of traditional streaming technologies, and adds semantic understanding and decision support to the pipeline. This transformation turns the construction data pipeline into a data and intelligence platform that leverages context to fuel real-time analytics, automate RFI resolution, predict safety events, and power secure multi-tenant construction SaaS applications.

5. Proposed LLM-Driven Construction Analytics Architecture

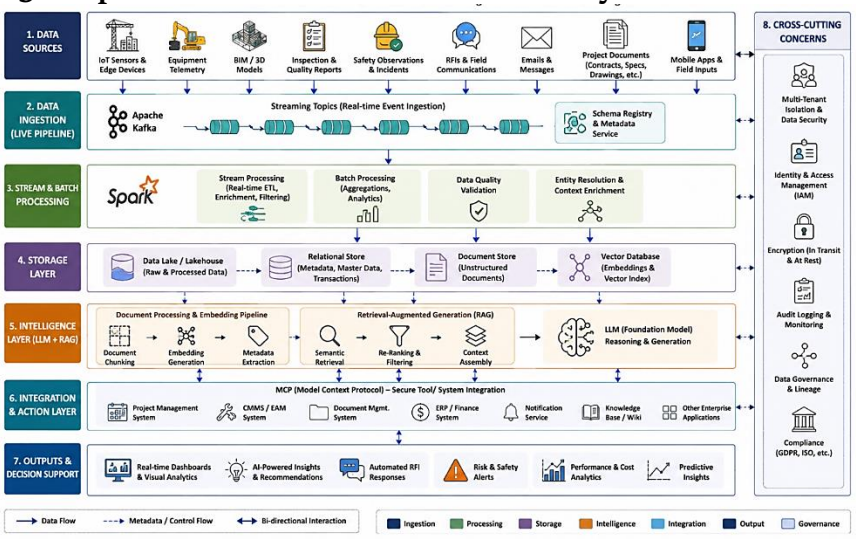

Figure 1. Proposed LLM-Driven Construction Analytics Architecture

5.1. Architecture Overview

The proposed construction analytics architecture includes the following components: high-throughput streaming, distributed data processing, one data storing, semantic retrieval, and large language model reasoning, which are connected as a fully end-to-end pipeline. It is designed around several layers that include job-site data acquisition, streaming ingestion, distributed processing, lakehouse storage, unstructured document processing, LLM intelligence, contextual retrieval, semantic analytics, and decision support. [7] The ingestion layer receives data from job sites, such as from IoT sensors, construction equipment, BIM systems, mobile applications, RFI, field reports, safety records, inspection systems and project repositories, and these data sources are constantly gathering data and delivering it through the ingestion layer. Event-streaming is the backbone and distributed transformation, filtering, aggregation, and stream processing are done by Apache Spark. Processed information is stored in a scalable data lake/lakehouse and shared with the semantic intelligence layer, which RAG accesses to extract relevant project knowledge and the LLMs to understand the different information types. Thus, the architecture enables the transfer of the scalability of traditional data engineering to contextual reasoning, allowing to convert, from the raw, of construction events and documents into grounded insights, alerts, recommendations and automatic decision support outputs.

5.2. Apache Kafka Streaming Architecture

The proposed architecture points to Apache Kafka to act as the real-time ingestion and event-distribution backbone, allowing for continuous ingestion of high-volume construction events from a variety of job-site data sources. Events are published to topics in Kafka then logically separated by data type, project, tenant or operational domain via IoT sensors, equipment monitoring systems, mobile applications, project management platforms, operational services. Kafka partitions offer parallel processing and horizontal scalability, and replication and event durability guarantee resilience against component failures. [8] A schema-aware ingestion strategy may be used to ensure consistency of structured and semi-structured events, with metadata (such as tenant identifier, project identifier, source system, timestamp, event type, and security classification) attached to each message to enable downstream governance and context processing. The topics are subscribed by the Spark Streaming or Structured Streaming consumers which then process the data in real-time for cleansing, enrichment, aggregation, routing and ultimately writing the data back into the data lake or pushing the data to semantic processing. This design isolates the ingestion of events from high throughput workloads from downstream analytical or LLM workloads, so that the streaming infrastructure can continue to operate without any major changes.

5.3. Retrieval-Augmented Generation Layer

The contextual knowledge mechanism is realized by the Retrieval-Augmented Generation (RAG) layer, which enables the LLM reasoning to link with project-specific construction information. Unstructured sources like RFIs, specifications, safety procedures, inspection reports, field communications, contracts, drawings, and historical project records go through the process of being parsed and segmented, metadata added, and then stored in an appropriate retrieval index in a vector representation. If the analytical request or event on the job site gets to the intelligence layer, the system creates a query representation and fetches pertinent information based on semantic similarity, metadata, project scope, tenant identity and authorization policies. The retrieved context is then used in conjunction with the current event, user request and other relevant structured information to form a grounded LLM input. This can alleviate the model's dependency on the static knowledge and allow that the responses will mirror the latest information of the project. In the proposed architecture, RAG also serves as a security and governance boundary, as the information retrieved is limited to the information of the project and the tenant, and cross-project or cross-tenant context exposure is limited.

5.4. End-to-End Data Flow

The end-to-end workflow starts with the collection of distributed construction job site and enterprise information (structured and unstructured). Real-time events are delivered to Kafka, which classifies, partitions and routes the events to the appropriate project, tenant, source, and event type, and Spark performs distributed preprocessing, transformation, enrichment and analytical operations. Structured outputs and documents are stored in the data lake or lakehouse, while unstructured is also indexed for semantic retrieval. [9] Intelligence layer is then contacted for relevant events and analytical requests, where RAG fetches approved construction knowledge and fuses with real time operational context. The LLM then carries out semantic interpretation, reasoning, summarization, classification and/or recommendation generation, and if required, connects with enterprise applications and project systems via MCP-based tool integration. The validation, confidence and access control policies are applied to the generated outputs prior to delivering to dashboards, project management systems, alerts and/or human decision makers. The pipeline is an ingest–process–store system that is now morphed into ingest–process–retrieve–reason–act for delivering context-aware construction intelligence in near real time.

6. LLM and RAG-Based Construction Intelligence

6.1. Construction Knowledge Representation

Construction knowledge representation serves as the basis for translating the various data in a construction project into a form that is easily accessible, correlated, and interpreted by the proposed LLM-driven construction analytics pipeline. There are multiple structures and levels of contextual details within each construction knowledge source, including RFIs, specifications, contracts, BIM models, field reports, inspection records, safety documents, equipment logs, schedules, emails, and historical project records. [10] The proposed approach groups these sources together with document metadata, project identifiers, tenant identifiers, timestamps, document type, construction entities, activities, locations, equipment, risks and relationships between project artifacts. Operational information can be extracted as structured information that can be correlated with unstructured information in the form of text to represent a unified project context. This representation helps the system differentiate projects and tenants, identify the relationship between the construction entities and keep the track of the origin of retrieved information. The architecture provides a consistent knowledge representation layer that allows downstream embedding, retrieval, context construction using RAG, and reasoning by the LLM, enabling access to construction knowledge that is relevant and authorized, instead of individual documents.

6.2. RAG-Based Context Construction

The RAG based context construction mechanism associates real time construction events and relevant information which are stored in project knowledge repositories. The system first recognizes the referenced project, tenant, entities, and operational context based on the receipt of an event, user query, RFI, safety observation or analytical request, and then issues a retrieval query. The retrieval process performs a semantic similarity search of the indexed construction documents and knowledge sources with metadata constraints like project identifier, document type, date, authorisation level and tenant boundary. [11] The most relevant documents or document segments are then ranked and gathered with real-time structured information from Kafka and Spark processing. This context is what will be used by the LLM to help it understand the current context and provide a grounded response. For instance, an RFI about a construction material specification can be enhanced with the relevant contract clauses, technical specification, past RFIs, BIM data and field observations prior to the LLM process. This mechanism helps to integrate the information generated on the job site with project knowledge, historical and/or domain-specific, and to minimize irrelevant context and minimize unauthorized information retrieval, in the proposed architecture.

6.3. Real-Time Semantic Reasoning

The first intelligence phase of the proposed architecture is the real-time semantic reasoning that transforms contextualized construction information into actionable analytical outputs by using the LLM. Once relevant structured events and retrieved documents have been gathered, the LLM can be used to analyze relationships between events, project conditions, historical data, technical needs, and operational limitations to accomplish various tasks like classification, summarization, risk identification, question answering, recommendation generation, and RFI analysis. [12] Semantic reasoning can understand natural language variations and find relationships that may not explicitly be encoded in structured data fields, unlike the traditional rule-based processing methods. For example, an equipment anomaly report, observation in the field and a safety report can be seen together and a potential operational risk can be determined, and an alert can be generated. The reasoning process is limited by retrieved evidence, tenant authorisation policies and output requirements, to increase reliability and traceability. As a result, the proposed pipeline takes real-time analytics from detecting and aggregating events to understanding their context and providing construction intelligence – based on evidence.

7. Multi-Tenant Construction Analytics Architecture

7.1. Multi-Tenant SaaS Architecture

The proposed multi-tenant construction analytics architecture allows multiple construction organizations and projects to collaborate in a common analytical platform without sacrificing the logical separation of data and processing resources, knowledge repositories, and contexts for LLM applications. A tenant identifier is assigned to each tenant and carried throughout the entire data lifecycle, from job site ingestion via Kafka, Spark processing and onto the lakehouse, the vector store, retrieval and LLM inference. [13] Before the data goes into downstream processing stages, IoT sensor data, equipment systems, BIM platforms, as well as RFIs, field applications, and project repositories are linked to tenant and project metadata. The shared infrastructure model maximizes resource utilization, makes the platform management easier and enables independent project level analytics. The architecture isolates common platform services from tenant-specific data and authorization policies, enabling organizations to leverage common streaming, processing, and inference without revealing information across organizational boundaries. The design offers the flexibility needed to

deploy the Software-as-a-Service in a scalable manner while maintaining construction intelligence as a key part of the data ownership and operational requirements for individual tenants.

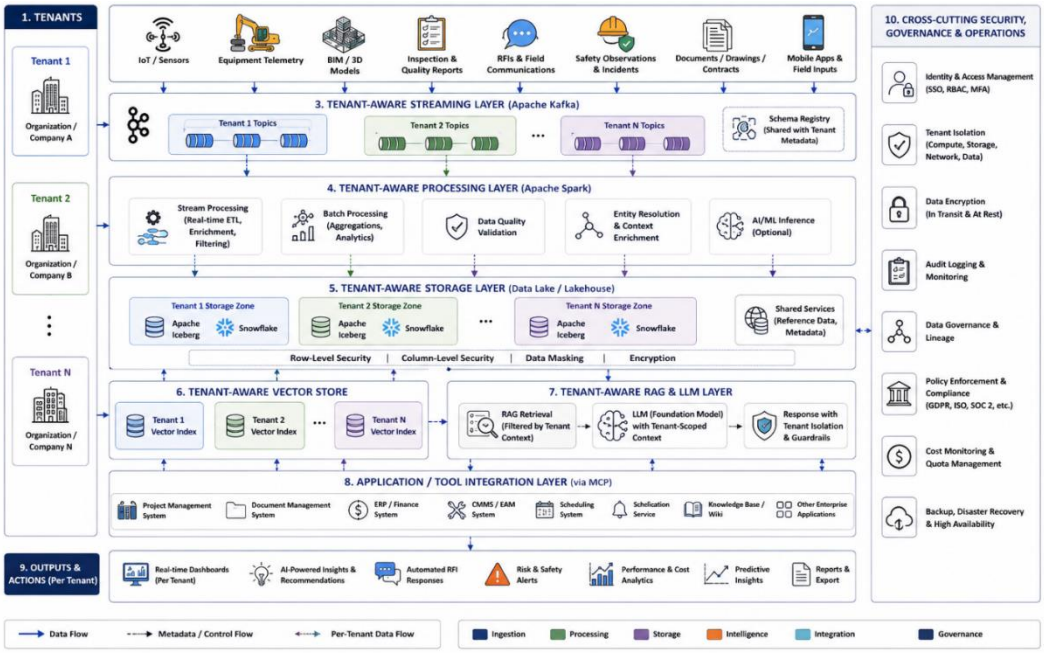

Figure 2. Multi-Tenant Construction Analytics Architecture

7.2. Tenant Isolation Strategy

Construction jobs may have commercially sensitive information, proprietary designs, contractual information, safety records, and operational data, that can't be accessed by other organizations, which is an essential security requirement. The proposed Architecture is a defense in depth isolation at the ingestion, processing, storage, retrieval and inference layers. Tenants are identified when ingested, and are added to each event before it is published to the relevant Kafka topics or logical partitions. Tenant-aware filtering and access policies help to ensure that transformations and aggregations are performed only on data that is authorized to be processed. Lakehouse tables can use tenant-specific partition, namespaces, access policies, and row-level controls at the storage layer, and sensitive attributes can be secured using column-level controls. Likewise, vector indexes, retrieval services are also isolated so that documents from unauthorized tenants are not returned by semantic searches. The assembled context is then checked against tenant and user authorization information before it is used to add to prompts or responses in LLM inference to ensure that information is not added to prompts or responses from another tenant. These mechanisms provide the end-to-end architectural characteristic of isolation, not just as a single database-level security mechanism.

7.3. Tenant-Aware RAG Retrieval

Tenant-aware RAG retrieval allows relevant construction knowledge to be retrieved while maintaining the boundaries of the organization and project. The construction documents (RFIs, specs, safety procedures, inspection reports, contracts, field communications, project history) are indexed and associated with metadata (tenant, project, document type, access level, document time, etc.) that authorizes access. [14] The retrieval service creates the authenticated tenant and project context before invoking semantic similarity search to satisfy a user query or a real-time construction event that triggers an analytical request, and uses the context to impose constraints for the search. Only embeddings and document segments that are allowed for the requesting tenant and project can be retrieved, and then the retrieved parts of the evidence set are ranked and compiled into the LLM context. Other validation can confirm that the document is owned, authorized for access and document provenance before sending the context to the model. That is because the tenant-aware retrieval mechanism is crucial for avoiding cross tenant leakage, as another organisation can return semantically similar information with the semantic vector search. The proposed architecture incorporates several key features, such as metadata-based filtering, authorization policies, retrieval isolation, and context validation, to enable RAG to deliver project-specific intelligence while maintaining the confidentiality and separation necessary for a multi-tenant construction SaaS environment.

8. Intelligent Construction Analytics Use Case

8.1. RFI Ingestion Through Kafka

The proposed automated RFI resolution pipeline starts with considering each new addition or update to Request for Information (RFI) as an event within the construction data-streaming infrastructure. The records from RFI's created in project management systems, mobile field applications, document systems, or collaboration systems are translated into standardized events with various fields, including tenant identifier, project identifier, RFI identifier, subject, description, author, location, discipline, priority, time, and document references. These events will be published to dedicated Kafka topics, allowing for reliable, scalable and near real-time distribution to downstream processing services. [15] Kafka supports processing of messages in parallel across projects and tenants with the help of Kafka partitions, and the durability and ordering of messages supports the operational sequence of the RFI activities with the help of the ordering mechanisms in Kafka. Tenant and project metadata are carried in with all events to ensure authorization and isolation across the processing pipeline. This is an event-driven approach, meaning no periodic manual extraction and the analytical pipeline will kick-off the processing of an RFI as soon as an inquiry is submitted or changed.

8.2. RAG-Based Context Construction

Once the RFI is ingested and preprocessed, then the RAG layer will build a contextual representation of the project that it will use to retrieve information relative to the question or a construction problem that is not resolved. The description of the RFI is converted to a semantic RFI query and a search is performed on the existing historical RFIs, technical specifications, contracts, drawings, BIM data, inspection reports, field reports, project communication and other authorized knowledge sources. Retrieval integrates semantic similarity with metadata based constraints such as tenant, project, discipline, document type, location, access rights, etc. to only include relevant and authorized information in the response. The retrieved evidence is sorted and integrated with the structured project information and real-time operational context, before being fed to the LLM. This process allows the system to go beyond the document search based on keywords, and can detect semantically related information even if there are different keywords in the project records. This context helps the LLM generate a project-aware solution based on facts, and ensures that the solution can be traced back to the data. This context gives the LLM knowledge of what has happened with projects, which enables it to produce a reliable and traceable candidate resolution.

8.3. LLM-Based RFI Reasoning

The reasoning using the LLM stage processes the original RFI along with the retrieved project context to ascertain the character of the problem, pinpoint pertinent requirements and create a possible resolution. The model can be used for various operations such as RFI classification, extraction of the construction entities and constraints, comparison of the proposed conditions with project specifications, summarisation of supporting evidence, and generate suggested responses. [16] The reasoning process is not just based on the model's pretrained knowledge, but on retrieved project documents and also structured information provided by the RAG layer. The resolution generated may contain the suggested resolution, supporting evidence, specification or document references, confidence indicators and any information that is not resolved and needs to be reviewed by a human. A validation mechanism then can test the response, to determine if retrieved evidence supports it and if the proposed action meets project-specific requirements. The final answer is thus considered a recommendation on how to proceed, and not a standalone decision, and can be approved, amended or rejected by the project engineers, managers or other approved person before it is added to the project management chain.

9. Experimental Setup and Evaluation Framework

9.1. Experimental Objectives

The experimental evaluation aims at establishing the potential of the proposed construction analytics architecture, built on LLM, in enhancing the efficiency, scalability, contextual intelligence, and operational effectiveness of traditional construction data pipelines. [17] The experiments test the whole data processing flow – from ingestion at the job site to Kafka and to Spark; to retrieval via RAG, reasoning with the LLM, and to the final generation of the decision support. The first goal is to determine the level of capability the architecture has to handle an ever growing number and rate of construction events while still meeting acceptable latency and resource utilization. The second goal is to assess the performance of the RAG and LLM for comprehending construction-related unstructured data and generating pertinent, evidence-based RFIs responses. The third goal is to evaluate multi-tenant scalability and determine that as the number of tenants and projects grows, processing performance and data isolation won't suffer. Finally, the proposed architecture is compared to the conventional Kafka–Spark or non-RAG analytical pipelines to quantify the improvements in processing efficiency, semantic accuracy, response time, scalability, and resource utilization.

9.2. Multi-Tenant Test Configuration

The goal of the multi-tenant evaluation is to test the proposed architecture's ability to serve multiple construction organizations and projects simultaneously, while providing for strict data and context isolation. The experimental environment can be scaled up to various numbers of tenants, projects, data streams, requests for information (RFI), and concurrent analytical queries to simulate realistic SaaS workloads. [18] Every tenant receives unique identifiers and authorization policies, with events received through Kafka, processing done in Spark, records stored in the lakehouse, vector indexes, and RAG retrieval requests including tenant and project metadata during the processing lifecycle. Test scenarios feature normal concurrent workloads, high volume of events being ingested, processing of RFI at the same time, adding more tenants and situations that contain resources. Security-based tests also attempt to access documents and/or produce responses in unauthorized tenant and project contexts to ensure that the isolation mechanisms are effective. Throughput, latency, resource utilization, retrieval accuracy and isolation-violation rate are measured metrics used to evaluate the architecture to ensure it can continue to be secure and perform as expected as the SaaS environment grows.

9.3. Evaluation Metrics

The proposed architecture is assessed using the metrics of system performance, semantic intelligence, scalability and security to give a holistic assessment. The performance of data pipelines is evaluated via ingestion throughput, processing throughput, end-to-end latency, event processing delay and pipeline availability. Performance of RAG and LLM is measured in terms of retrieval precision, retrieval recall, contextual relevance, answer accuracy, groundedness, response latency and RFI resolution effectiveness. Other metrics for reducing the RFI automation tasks are the reduction in resolution time, correct recommendation rate, evidence-support rate, and human approval rate.

When evaluating scaling and efficiency, the following factors are considered: CPU utilization, memory utilization, storage utilization, maximum number of concurrent requests, optimization cost, and performance degradation as load and the number of tenants grow. [19] Multi-tenant success of isolation, unauthorized retrieval attempts, cross-tenant leakage rate and access control enforcement accuracy are measured for multi-tenant security. These metrics allow for a direct comparison between the proposed Kafka–Spark–LLM–RAG pipeline and the conventional construction analytics architectures, and can help provide quantitative evidence of how this pipeline improves the data-processing performance and context-aware construction intelligence.

Table 2. Experimental Configurations for Comparative Evaluation

Approach	Architecture	Purpose
A1	Conventional Batch Analytics	Baseline
A2	Kafka + Spark	Streaming baseline
A3	Kafka + Spark + LLM + RAG	Semantic intelligence
A4	Kafka + Spark + LLM + RAG + MCP	Proposed architecture

The table presents the four approaches used to test the evolution from traditional construction data processing methods to the proposed approach of constructing analytics architecture based on LLM technology. A1 is the traditional batch analytics method, and this will be used as a reference. A2 adds support for Apache Kafka and Apache Spark for high throughput stream processing on continuously generated job-site data in a distributed fashion. With LLM and RAG features, A3 seamlessly integrates into the Kafka–Spark pipeline, allowing for semantic understanding and retrieval of construction knowledge from unstructured sources like RFIs, field reports, and project documents.

The full architecture, incorporating Kafka, Spark, LLM, RAG and Model Context Protocol (MCP), is called A4. Along with streaming and semantic reasoning, A4 also allows controlled interaction with the external construction systems and tools that comprise the enterprise. A1 – A4 enables the study to measure the contribution of each architectural evolution with the following metrics: data throughput, processing latency, semantic insight accuracy, RFI resolution accuracy, RFI resolution time, resource utilization, and scalability. The table then serves as the basis for the experiments to show that the proposed architecture would enhance not only data-processing capability, but the progression from raw data at a job site to context-aware construction intelligence as well.

10. Results and Performance Evaluation

10.1. Data Ingestion Throughput

To evaluate the capability of the proposed architecture to process construction events of different types continuously at growing workloads, the data ingestion evaluation is introduced. The evaluation takes into account information from IoT sensors, equipment telemetry, mobile devices, project management systems, and RFI platforms and compares the proposed Kafka–Spark–LLM–RAG pipeline to the classic Kafka–Spark pipeline. [20] These are the factors such as number of events processed per second, message processing delay, success rate of message ingestion, and degradation of performance with growing data volumes. The outcomes will show that the Kafka ingestion layer can achieve high throughput and distribute its workloads across partitions and downstream Spark processing nodes. Most important is the evaluation of LLM and RAG integration is outside of the main ingestion path so that the ingestion of high volume of events is not unnecessarily hampered by the semantic processing workloads. The separation of the real-time data ingestion from the computationally expensive semantic reasoning process highlights the architectural benefit of decoupling them, which enable the proposed system to still provide streaming performance while supporting sophisticated construction intelligence.

10.2. RFI Resolution Accuracy

RFI resolution accuracy is a measure of how well the proposed RAG and LLM components can provide relevant, evidence-based answers to construction-related questions. Each RFI is reviewed using authoritative project information, such as project specifications, contracts, drawings, BIM-related records and historic RFIs, inspection reports and field documentation. Assessment evaluates aspects of correct resolution rate, contextual relevance, evidence-support rate, factual consistency and human expert agreement. [21] The proposed RAG-based approach is evaluated in comparison to the conventional retrieval based on keywords, non-RAG LLM generation, and traditional rule-based document processing, to examine the role of contextual retrieval and LLM reasoning. Special emphasis is placed on the accuracy of the responses made to identify project requirements that apply to the project and not make recommendations that do not. The results should show that using semantic retrieval along with LLM reasoning results in more accurate and contextually relevant RFI resolutions than methods based on keyword matching, static rules, or the LLM's pre-trained knowledge. The analysis also sets out the utility of supporting the responses generated by the project with legitimate project evidence, prior to human approval.

10.3. Comparison With Conventional Pipeline

The comparative evaluation is based on the overall performance of the proposed LLM-driven architecture, compared with the workflow of building data-processing pipelines. The baseline architecture is built around Kafka and Spark, where both are used for distributed processing, but it lacks an integration with semantic intelligence because the data is stored in a lakehouse, and to use the LLM or RAG a tenant must supply the context for the data. The comparison focuses on throughput of data ingestion, latency of stream processing, end-to-end response time, effectiveness of document retrieval, accuracy of resolving RFIs, resource usage, scalability and operational efficiency. The introduction of LLM and RAG components, while adding some computational complexity to the system, is likely to yield significant enhancements in analytical functionality and generate valuable insights and recommendations based on the context of the data, ultimately transforming the system into a more powerful and efficient tool for handling complex data and tasks. The comparative results hence evaluate the cost and intelligence gained in the computations, showing that the proposed architecture is not just a faster data pipeline, but an evolution from data processing to construction intelligence. The comparison offers the first quantitative data to support the research hypothesis and the feasibility of using the streaming infrastructure to integrate with LLM-based analytics.

Table 3. Overall Performance Comparison of Four Construction Analytics Architectures

Metric	A1	A2	A3	A4 Proposed
Data Throughput (events/s)	420	1,250	1,180	1,360
End-to-End Latency (ms)	2,850	1,420	980	760
RAG Retrieval Accuracy (%)	61.5	65.8	88.6	92.4
Semantic Insight Accuracy (%)	58.2	63.7	86.9	91.7
RFI Resolution Accuracy (%)	52.4	60.8	84.5	90.6
RFI Resolution Time (min)	46.0	34.5	18.2	12.7
CPU Utilization (%)	78.5	71.2	69.8	64.3
Memory Utilization (%)	82.1	76.4	74.6	70.8

We provide the overall quantitative comparison of the four construction analytics architectures here—conventional batch analytics (A1), Kafka–Spark (A2), Kafka–Spark–LLM–RAG (A3), and the proposed architecture Kafka–Spark–LLM–RAG–MCP (A4). The results demonstrate the incremental gains of adding streaming, semantic retrieval, LLM reasoning, and enterprise-tool integration. The architecture proposed here is an A4 architecture which provides the best data throughput of 1360 events/s and the lowest end-to-end latency of 760 ms, thus enhancing the real-time data processing capability. It also performs better in respect of contextual understanding and construction-document processing in terms of RAG retrieval accuracy (92.4%), semantic insight accuracy (91.7%) and RFI resolution accuracy (90.6%) compared to the other approaches.

The table also shows the actual enhancements to the processing of RFI and computational efficiency. The resolution time for RFI is reduced to 12.7 min in A4 compared to 46.0 min in A1 and the CPU and memory utilization is reduced to 64.3% and 70.8% respectively. In summary, the integration of Kafka, Spark, and LLM with RAG and MCP can enhance traditional construction data pipelines beyond data ingestion and distributed processing to enable real-time, context-aware construction intelligence.

Table 4. Data Throughput and End-to-End Latency Across Four Construction Analytics Architectures

Approach	Throughput (events/s)	Latency (ms)
A1	420	2,850
A2	1,250	1,420
A3	1,180	980
A4 Proposed	1,360	760

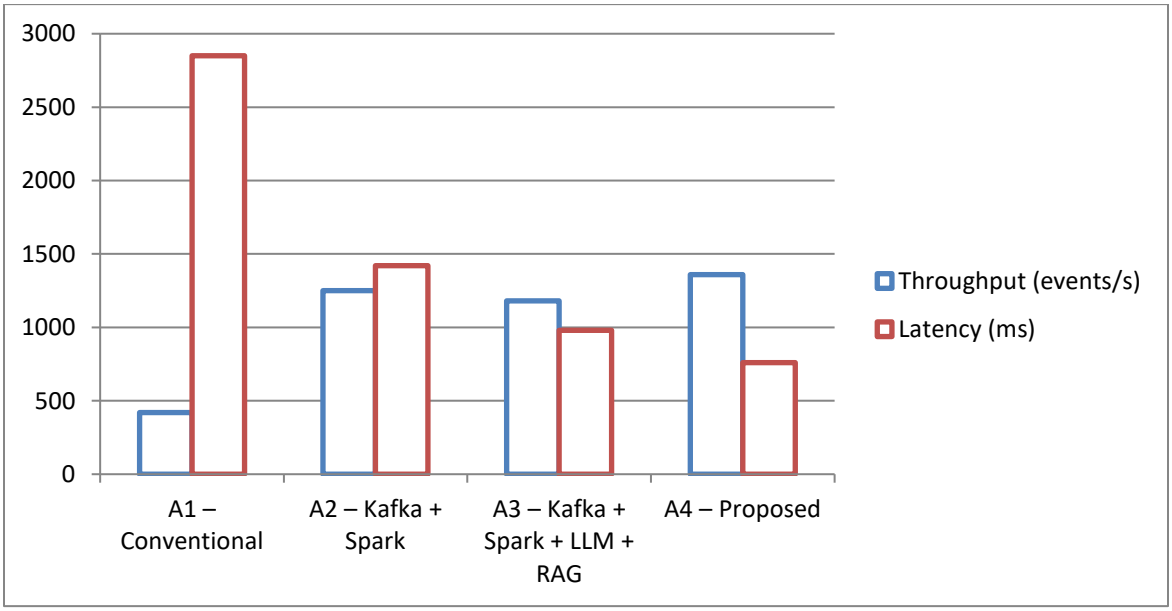

Figure 3. Graphical Data Throughput and End-to-End Latency Across Four Construction Analytics Architectures

Table 5. Semantic Intelligence and RFI Resolution Performance Across Four Architectures

Approach	RAG Retrieval Accuracy (%)	Semantic Insight Accuracy (%)	RFI Resolution Accuracy (%)
A1	61.5	58.2	52.4
A2	65.8	63.7	60.8
A3	88.6	86.9	84.5
A4 Proposed	92.4	91.7	90.6

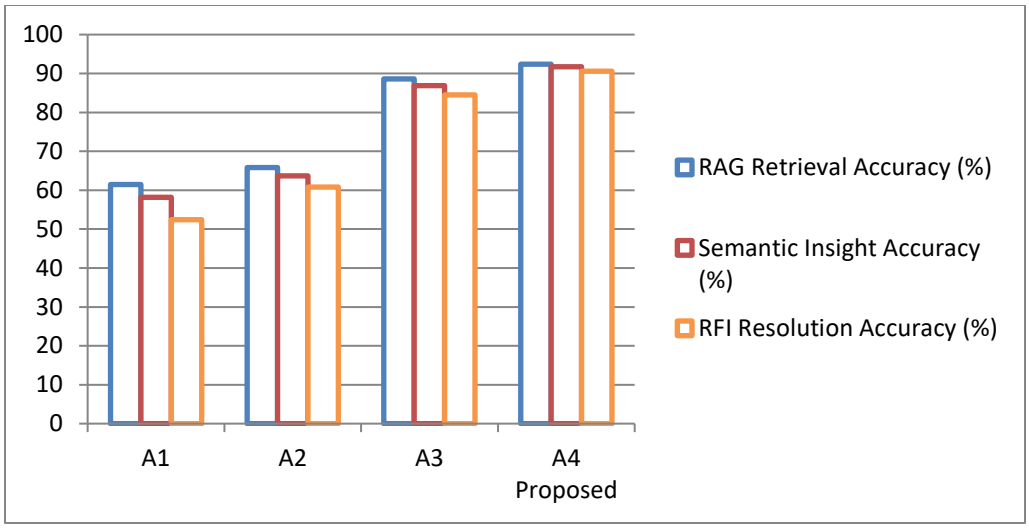

Figure 4. Graphical Semantic Intelligence and RFI Resolution Performance Across Four Architectures

11. Discussion

11.1. Interpretation of Experimental Results

The experimental results confirm that the proposed Kafka–Spark–LLM–RAG architecture overcomes the limitations of traditional construction data pipelines, which focus on data in the context of high throughput event processing, and introduces context-aware semantic intelligence. Kafka and Spark still deliver scalable ingestion and distributed processing for structured and semi-structured data from jobsites, while the LLM and RAG layers interpret the RFIs, field reports, safety observations, technical documents and communications from the jobsite. By integrating RAG, the system can draw from the project-specific evidence to enhance the ability to base generated outputs, mitigating the scope of problems encountered in using isolated LLM inference. Results also show that semantic processing adds to the computation and inference latency when compared with traditional streaming analytics, but offers significantly more analytical capabilities such as retrieval of contextual information, natural-language reasoning, automated RFI analysis and intelligent decision support. The multi-tenant evaluation also shows that using tenant-aware ingestion, storage, retrieval and context management can enable scalable SaaS deployment, while also preserving organization data boundaries.

11.2. From Telemetry Processing to Semantic Intelligence

The summary results suggest that the main advantage of the proposed architecture is not only the introduction of an LLM, but also the shift of the role of the analytical part of the pipeline from the processing of telemetry to the semantic intelligence. While Kafka and Spark effectively address questions about what, when, and how often something happened, the built-in RAG and LLM layers add another layer of capabilities that enable users to gain insights into the meaning, connections, and implications of those events in the context of the construction itself. Furthermore, the interaction with enterprise systems is possible in a controlled manner with MCP, so that the insights generated can become part of the operational process, and are not just isolated analytical results. This means that the architecture is capable of providing solutions for things like automated RFI resolution, evidence-based alerting for safety, retrieval of project knowledge, and context-aware decision making. On the operational side, the evolution can cut down on manual document analysis, speed up the access to information, facilitate better coordination between the project stakeholders and enable faster responses to unforeseen project circumstances. However, human judgment and oversight will always be required in high-impact construction decisions, specifically when generated recommendations have safety, contractual or engineering judgment and/or financial consequences.

12. Limitations and Future Research

12.1. Limitations and Future Research

This proposed LLM-based construction analytics architecture faces some limitations and should be taken into consideration when considering its performance and real-world applicability. The quality of generated insights still relies on the ability of the chosen LLM, completeness and quality of construction data and project-specific knowledge retrieved. Field reports may be incomplete, terminology may not be consistent, metadata elements may be missing, sometimes available documents are outdated, and project

information may be disjointed, potentially affecting retrieval and reasoning accuracy. Running LLM inference can also add to the computation cost and latency, especially for high volumes of real-time events or concurrent requests, across multiple tenants. While the grounding and evidence validation of RAG can mitigate risks of hallucination, the output generated by this process might still include incorrect or unsupported interpretations and thus may need to be validated and monitored by humans. As the number of construction organisations and projects continue to increase, maintaining strict tenant isolation in Kafka streams, lakehouse storage, vector indexes, RAG retrieval and LLM contexts becomes more complicated. Also, there is a lack of more comprehensive construction data sets, which are standardized, and publicly available for more reproducible benchmarking and generalizing experimental results.

The proposed architecture can be extended to multi-modal and more and more autonomous construction intelligence in the future. Multimodal LLMs may combine text, images, drawings, BIM models, video, sensor signals and equipment telemetry to create more comprehensive models of on-site conditions. Edge–cloud architectures could also reduce response latency by doing some light preprocessing, filtering, or inference at the edge (i.e., near the construction site) and then sending the more complex reasoning to the cloud. Future research may explore agentic construction analytics, where LLM agents manipulate tools for retrieval, data analysis, queries of the project system and execution of project workflows through the use of carefully designed interfaces. These solutions have the potential to support more sophisticated applications like proactive safety monitoring, automated RFI life cycle management, schedule-risk analysis and project knowledge assistants. However, robust authorization, evidence validation, auditability, human approval and safety limits should all be integrated into autonomous project decision support prior to recommendations impacting on key construction activities. These instructions can also help pave the way for future developments of the proposed pipeline, moving from context-aware analytics to multimodal, adaptive and controlled construction intelligence platforms.

13. Conclusion

In this paper, an LLM-based architecture for construction analytics emerged, transforming traditional job-site data pipelines from high throughput data processing to semantic intelligence in the context of construction. The proposed framework includes Apache Kafka for real-time event ingestion, Apache Spark for stream processing, Unified Storage with the lakehouse data managment, Retrieval-Augmented Generation (RAG) for accessing project-specific knowledge, and finally the Large Language Models (LLMs) for semantic reasoning and insights generation. The architecture is designed to overcome the limitations of current pipelines and their ability to process non-structured construction information like RFIs, field reports, safety observations, inspection records, field communications and technical documents. The framework features intelligent analytical functions, such as evidence-based resolution of RFIs, retrieval of contextual information, identification of risks and decision support, through the integration of structured operational data and retrieved contextual knowledge.

Another key finding of the research is the adoption of multi-tenant SaaS principles at every stage of data ingestion, processing, storage, retrieval, LLM context management, and beyond, allowing construction companies to utilize shared scalable infrastructure under the premise of security and data isolation. The proposed architecture shows how existing streaming technologies can be used as the foundation for a modern construction analytics platform with the addition of a semantic intelligence layer through the integration of LLM, RAG, and controlled tools for advanced decision support. While there are still issues to be addressed around the reliability and computational cost of LLM, data quality, latency, and the ability for autonomous decision making, the proposed approach is a viable architectural path to generating construction knowledge from fragmented, job-site data. In total, the shift from “ingest–process–store” to “ingest–process–retrieve–reason–act” serves as a base for next-generation construction analytics that will help with faster project coordination, better access to information, more effective resolution of RFIs, and more intelligent construction management.

References

[1] Tang, S., Shelden, D. R., Eastman, C. M., Pishdad-Bozorgi, P., & Gao, X. (2019). A review of building information modeling (BIM) and the internet of things (IoT) devices integration: Present status and future trends. Automation in construction, 101, 127-139.

[2] Wu, C., Li, X., Guo, Y., Wang, J., Ren, Z., Wang, M., & Yang, Z. (2022). Natural language processing for smart construction: Current status and future directions. Automation in Construction, 134, 104059.

[3] Ding, Y., Ma, J., & Luo, X. (2022). Applications of natural language processing in construction. Automation in Construction, 136, 104169.

[4] Kreps, J., Narkhede, N., & Rao, J. (2011, June). Kafka: A distributed messaging system for log processing. In Proceedings of the NetDB (Vol. 11, No. 2011, pp. 1-7).

[5] Zaharia, M., Chowdhury, M., Das, T., Dave, A., Ma, J., McCauly, M., ... & Stoica, I. (2012). Resilient distributed datasets: A {Fault-Tolerant} abstraction for {In-Memory} cluster computing. In 9th USENIX symposium on networked systems design and implementation (NSDI 12) (pp. 15-28).

[6] Zaharia, M., Xin, R. S., Wendell, P., Das, T., Armbrust, M., Dave, A., ... & Stoica, I. (2016). Apache spark: a unified engine for big data processing. Communications of the ACM, 59(11), 56-65.
[7] Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., ... & Polosukhin, I. (2017). Attention is all you need. Advances in neural information processing systems, 30.
[8] Devlin, J., Chang, M. W., Lee, K., & Toutanova, K. (2019, June). Bert: Pre-training of deep bidirectional transformers for language understanding. In Proceedings of the 2019 conference of the North American chapter of the association for computational linguistics: human language technologies, volume 1 (long and short papers) (pp. 4171-4186).
[9] Brown, T., Mann, B., Ryder, N., Subbiah, M., Kaplan, J. D., Dhariwal, P., ... & Amodei, D. (2020). Language models are few-shot learners. Advances in neural information processing systems, 33, 1877-1901.
[10] Lewis, P., Perez, E., Piktus, A., Petroni, F., Karpukhin, V., Goyal, N., ... & Kiela, D. (2020). Retrieval-augmented generation for knowledge-intensive nlp tasks. Advances in neural information processing systems, 33, 9459-9474.
[11] Bommasani, R., Hudson, D. A., Adeli, E., Altman, R., Arora, S., von Arx, S., ... & Liang, P. (2021). On the opportunities and risks of foundation models. arXiv preprint arXiv:2108.07258.
[12] Liu, N. F., Lin, K., Hewitt, J., Paranjape, A., Bevilacqua, M., Petroni, F., & Liang, P. (2024). Lost in the middle: How language models use long contexts. Transactions of the association for computational linguistics, 12, 157-173.
[13] Afzal, M. (2024). Improving Request for Information (RFI) Processing in Construction Projects using Natural Language Processing (NLP) Techniques (Doctoral dissertation, University of Technology Sydney (Australia)).
[14] Fialho, B. C., Codinhoto, R., & Fabricio, M. M. (2020). BIM and IoT for the AEC Industry: A systematic literature mapping. Procedia Engineering, 2(4), 6.
[15] Guu, K., Lee, K., Tung, Z., Pasupat, P., & Chang, M. (2020, November). Retrieval augmented language model pre-training. In International conference on machine learning (pp. 3929-3938). PMLR.
[16] Izacard, G., Lewis, P., Lomeli, M., Hosseini, L., Petroni, F., Schick, T., ... & Grave, E. (2023). Atlas: Few-shot learning with retrieval augmented language models. Journal of Machine Learning Research, 24(251), 1-43.
[17] Our Evolutionary Journey of Building the LLM-powered Data Agent, Online. https://medium.com/@meghasaini/our-evolutionary-journey-of-building-the-llm-powered-data-agent-e6109b943923
[18] Asai, A., Wu, Z., Wang, Y., Sil, A., & Hajishirzi, H. (2024, May). Self-rag: Learning to retrieve, generate, and critique through self-reflection. In International conference on learning representations (Vol. 2024, pp. 9112-9141).
[19] Es, S., James, J., Anke, L. E., & Schockaert, S. (2024, March). Ragas: Automated evaluation of retrieval augmented generation. In Proceedings of the 18th conference of the european chapter of the association for computational linguistics: system demonstrations (pp. 150-158).
[20] van der Heijden, J. (2023). Construction 4.0 in a narrow and broad sense: A systematic and comprehensive literature review. Building and Environment, 244, 110788.
[21] Statsenko, L., Samaraweera, A., Bakhshi, J., & Chileshe, N. (2023). Construction 4.0 technologies and applications: A systematic literature review of trends and potential areas for development. Construction Innovation, 23(5), 961-993.
[22] Klinc, R., & Turk, Ž. (2019). Construction 4.0–digital transformation of one of the oldest industries. Economic and Business Review, 21(3), 4.
[23] Data-Driven Construction. Navigating the data age in the construction industry, datadrivenconstruction, Online. https://datadrivenconstruction.io/product/data-driven-construction/
[24] Craveiro, F., Duarte, J. P., Bartolo, H., & Bartolo, P. J. (2019). Additive manufacturing as an enabling technology for digital construction: A perspective on Construction 4.0. Automation in construction, 103, 251-267.
[25] Begić, H., & Galić, M. (2021). A Systematic Review of Construction 4.0 in the Context of the BIM 4.0 Premise. Buildings, 11(8), 337.