Journal of Neonatal Surgery
ISSN(Online): 2226-0439
Vol.13 , (2024)
https://www.jneonatalsurg.com

OPEN ACCESS

Comparing Batch vs. Streaming Approaches in Healthcare Data Warehousing Environments

Venkata Akhilesh Ranga Reddy[1]

[1]Application Architect,
Email ID : venkataakhileshkumar@gmail.com
ORCID ID:0009-0008-4140-2299

Cite this paper as: Venkata Akhilesh Ranga Reddy (2024) Comparing Batch vs. Streaming Approaches in Healthcare Data Warehousing Environments.Journal of Neonatal Surgery, 13, 2287-2309

ABSTRACT

Batch processing and streaming processing are the two main approaches used for moving data from data sources to data warehouses. Attributes of the healthcare data and requirements arising from the workloads typically being run on the data warehouse can determine the processing architecture that best aligns with a particular use case. Although there are trade-offs, they are not always clear, and empirically grounded guidance on which approach is preferred is lacking. A literature review, three academic institution case studies, and two healthcare case studies provide the foundation for answering the following questions: What are the major trade-offs between batch and streaming processing? In which situations does one solution provide a significantly better choice than the other? For which use cases can the solutions coexist? Addressing these questions helps to inform future architectural decisions regarding both building and augmenting healthcare data warehouses.

Process parallelism is effective in dealing with big data. If the data source produces events at a sufficiently high rate, then data can travel to the data warehouse as events become available, thus making it possible for analytics that run on the data warehouse on near-real-time intervals (seconds or minutes) to have little or no staleness. Naturally, managing the incoming data at such a scale is challenging and requires a robust streaming solution. However, in many applications, the cost of a streaming solution is difficult to justify. At lower velocities of incoming data, the cost and maintenance burden of the solution may well exceed the additional benefit it brings..

Key Words: Batch Processing Systems, Streaming Data Processing, Healthcare Data Warehousing, Data Pipeline Architectures, Real-Time Data Ingestion, Near Real-Time Analytics, Data Processing Trade-offs, Hybrid Processing Models, Big Data Parallelism, Event-Driven Architectures, Data Latency Optimization, Streaming Cost Analysis, Data Workflow Optimization, Healthcare Analytics Systems, Scalable Data Pipelines, Data Velocity Management, Warehouse Architecture Design, Data Engineering Strategies, Analytics Freshness, Processing Efficiency.

INTRODUCTION

Healthcare data warehousing environments encompass an array of processes involving the collection, storage, preparation, and analysis of healthcare-associated data to extract knowledge and promote evidence-based decision-making. Healthcare operations such as patient monitoring, health record (HR) management, and electronic prescription services generate fresh data every second. In recent years, a pool of processing frameworks has been designed to support near-real-time processing of data with low latency, enabling use cases in environments such as fraud detection, analytics of social network feeds, and monitoring systems. However, healthcare data-wrangling processes ensure not only that data are made available for analytics but also that high standards of dimensional data quality, consistency, and conformance are reached. The architectures of these processes have typically been based on batch processing, and comprehension of the main delineating factors that favour one processing paradigm over the other is a research gap.

The overall aim of this work is to critically assess the trade-offs between batch and streaming processing in healthcare data warehousing contexts. The study considers how requirements related to data freshness, latency, volume, velocity, and quality affect the appropriateness of each paradigm and composes a prescriptive decision framework for their selection. The thesis is that freshness and velocity are not the only influential factors; requirements on reliability, completeness, cost, and resilience are similarly significant. Such considerations are then mapped to a series of use cases, designed to guide processing-approach selection in a reliable manner. The findings are supported by both the academic literature and practical examples and presented to the academic community, computational designers, and data architects of large healthcare data-wrangling environments

A. Overview and Scope of the Document

Healthcare data is generated by numerous applications across multiple systems at an increasingly rapid pace and functions as an integral influencer of operational efficiency, research and development, clinical decision making, and revenue generation, among many other tasks. Due to these aspects, establishing a data warehouse that can consolidate this data and facilitate management analytics has become a necessity. Depending on the type of analytical requests from the management, the data warehouse can be designed using either a batch or streaming approach. The batch approach loads data periodically such that its timeliness in satisfying the analytical requests becomes a major concern, while the streaming approach loads data in real time to satisfy analytical requests that require freshness. Determining the best approach has, therefore, become a key challenge for chief information officers (CIOs) of hospitals and research institutions and is a frequently discussed topic at trade fairs. Nevertheless, the literature appears to be quiet on clarifying the appropriate circumstances for each processing

Architecture

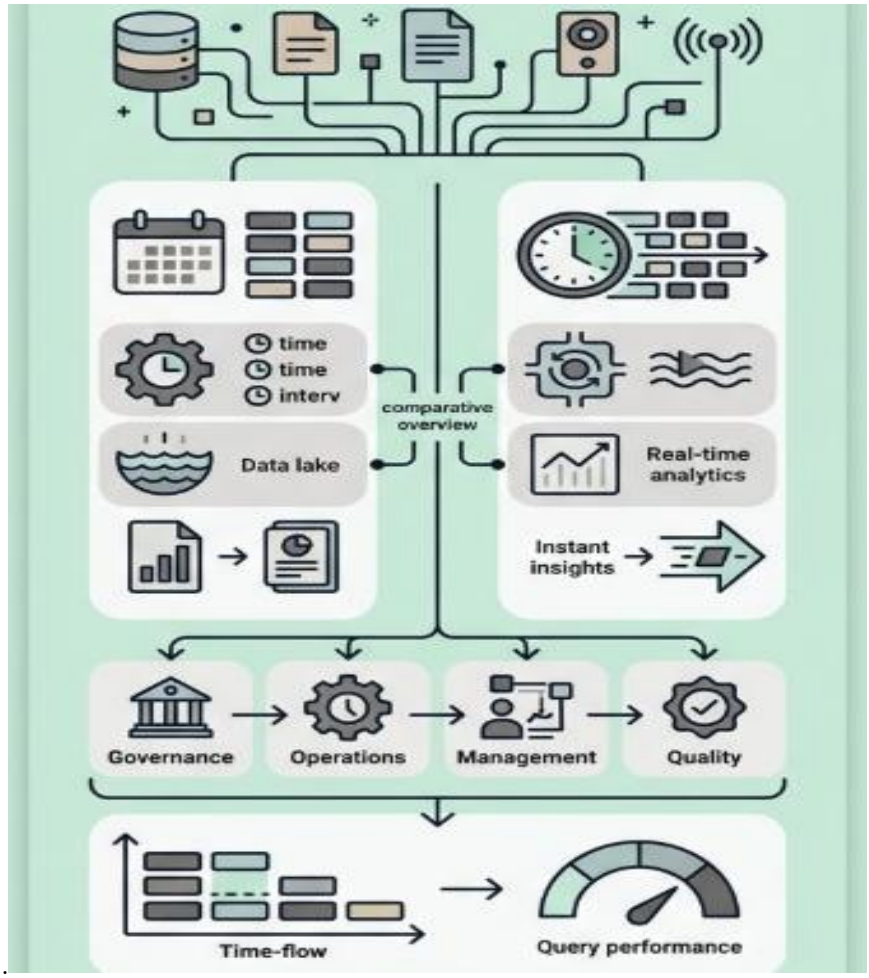


Fig 1: Evaluating Batch vs. Streaming Approaches for Optimal Operational and Governance Performance

The goal of this analysis was to provide a comparative overview of the batch and streaming approaches for health-care data warehousing in order to help decision-makers evaluate them based on the various qualities that each possesses. The hope was to catalyse discussions among healthcare CIOs and technology executives and to shed light on the conditions under which the options are indeed appropriate. The focus of the study was not on the architectural design of the processing technologies, but instead on their operational, governance, management, and quality dimensions. The specific temporal attribute of data freshness was examined in the context of determining the impact of temporal-region-based segment-querying capability on query performance.

Background and Definitions

Batch processing, streaming processing, data warehousing, data provenance, and regulatory coverage constitute the core concepts establishing background context for choice of processing architecture in the healthcare domain. A typical healthcare data warehouse consists of interfaces to various data sources, an ETL or ELT layer reshaping all incoming data into a predefined data model, and an analytics layer built above the data model. While architectural consideration usually begins at

the perimeter, specifically the interface to the source that drives the traffic volume into the warehouse, these join validation, what data is flowing through the warehouse, and its readiness for downstream consumption have equal importance. Batch and streaming processing represent two different models for how this reshaping occurs; a deployment decision must therefore consider the attributes of both models.

Batch systems are applied across the entire editable data set on a fixed schedule, regardless of whether the wharf's fixation to recent data is genuinely required. Streaming systems process events at their source when they occur, making the data available immediately for use. The suitability of a model is defined by a combination of the application, the providers supplying the information, and the acceptable life span of the data. Completeness emerges as a binary requirement, determining whether batch or streaming should be used. Batch systems decode and prepare data in partitions, allowing them to be sequenced one after the other, with high content and low latency detection of new stories at the expense of real-time flow as the content ages. Streaming constructions aim to deliver the lowest possible upstream latency while still meeting the completeness requirement. Decision criteria for choosing between the two families consider timeliness, latency, completeness, and three characteristics of the data itself—volume, quality, and governance—spanning privacy and privacy storage usage, PHI in particular.

Equation Set A. Freshness model

batch updates are periodic,

streaming updates happen as events arrive,

the choice strongly depends on how much staleness the use case can tolerate.

Step A1. Define data age

Let the age of data visible to analytics be

$$A = t_{\text{now}} - t_{\text{last\_available}}$$

Step A2. Convert age into a freshness score

A freshness score should be:

high when age is small,

low when age is large.

A standard normalized form is

$$F = e^{-A/\tau}$$

where τ is the maximum tolerable age scale for the use case.

Step A3. Batch freshness

In batch mode, data are loaded every T_u units of time.

So right after a batch load:

$$A \approx 0$$

Just before the next batch load:

$$A \approx T_u$$

Hence the average data age over one full batch cycle is

$$A_{\text{batch,avg}} = \frac{T_u}{2}$$

Substitute into the freshness formula:

$$F_B = e^{-A_{\text{batch,avg}}/\tau}$$

Therefore,

$$\boxed{F_B = e^{-T_u/(2\tau)}}$$

Step A4. Streaming freshness

For streaming, updates arrive with much smaller delay, approximately equal to pipeline latency L.

So

$$A_{\text{stream}} \approx L$$

Substitute into the same formula:

$$\boxed{F_S = e^{-L/\tau}}$$

Step A5. Interpretation

Streaming is better on freshness when

$$F_S > F_B$$

Substitute:

$$e^{-L/\tau} > e^{-T_u/(2\tau)}$$

Take natural log on both sides:

$$-\frac{L}{\tau} > -\frac{T_u}{2\tau}$$

Multiply by $-\tau$ and reverse inequality:

$$L < \frac{T_u}{2}$$

So the condition under which streaming gives fresher data is

$$\boxed{L < \frac{T_u}{2}}$$

A. Key Terminology and Contextual Insights

Batch processing refers to a technique in which data is collected over a period of time and processed together as one or more large blocks. In big data, the collected data set is typically so large that it is not possible to use a single request to analyze all the data. Instead, several requests are sent, and the data returned from the batch of requests is analyzed as a group. Streaming processing refers to analyzing data in real-time, and involves either evaluating smaller streams as soon as they arrive or continuously evaluating a stream. The streams may originate from data sources such as sensor feeds, calls, logs, videos, and more. Streaming processing is often useful when incoming data must be processed collectively and more often than once.

A data warehouse, according to the TDWI Cloud Data Warehousing & Analytics Research, is a hybrid platform supporting query-based ad hoc analytics, machine learning, and visual analytics of primarily structured but increasingly semi-structured data within the enterprise or community. Data sources serve as the interfaces among data loading/ingesting processes (extract, transform, and load, or ETL; extract, load, transform, or ELT; and data ingestion), the data model used by the data warehouse (e.g., a star schema or a snowflake schema), and data marshaling processes that provide data for business intelligence reports, dashboards, or advanced analytics, including machine learning. Data flows can also include data-provenance supporting processes or libraries enabling compliance, controls, and risk mitigation. Controls safeguard data privacy and confidentiality, with business intelligence processes not releasing protected health information (PHI) or de-identified data containing sufficient elements to identify an individual. Data used for compliance audits are subjected to review and an independent assessment that data-narrative workflows remain functioning as required and produce accurate results.

Table 1. Core comparison distilled

Dimension	Batch Processing	Streaming Processing	Why this follows from the paper
Data freshness	Periodic, lower freshness	Near-real-time, high freshness	The article repeatedly frames freshness and low latency as the strongest argument for streaming.
Latency	Higher due to scheduled execution	Lower due to event-driven ingestion	The paper distinguishes timeliness and latency, and gives streaming the edge on low-latency delivery.
Completeness	Usually stronger	Can be weaker when late/missing events occur	The article emphasizes completeness as a major reason batch remains preferred in many healthcare settings.
Data quality control	Easier to validate in processing segments	Harder because validation must be continuous	The paper says streaming must either validate continuously or temporarily admit lower-quality data.
Governance / audit / privacy	Simpler	More complex	The article connects batch with simpler governance and lower privacy/audit complexity, especially around PHI.

Dimension	Batch Processing	Streaming Processing	Why this follows from the paper
Cost	Often lower for lower-velocity workloads	Harder to justify at lower velocities	The article explicitly notes that streaming cost may outweigh benefit when source velocity is low.
Scalability	Good for large accumulated volumes	Good for high event rates, but operationally tougher	The paper describes batch as scaling well for large volumes and streaming as beneficial when events arrive fast enough.
Complexity	Lower	Higher	The article repeatedly states streaming requires more careful tuning and more complex architecture.
Best-fit workloads	BI, audits, retrospective analytics, research warehousing	alerts, operational monitoring, time-critical decision support	This is the decision pattern synthesized in the results and conclusion sections.

METHODOLOGY

Batch and streaming operational architectures were compared according to a defined set of criteria relevant to healthcare data warehousing. The experimental setup began with a statement of the overall research design strategy, followed by metadata—explicit data source definition and dimensionality of the architectural comparisons. Finally, a set of evaluation criteria for use case performance, reliability, governability, data completeness, and data quality was specified. Batch and streaming processing use cases were subsequently aligned along the established metadata before an analysis of completeness was performed.

The analysis outlined the limitations of a research agenda that focused only on the provability, completeness, or precision of the ETL process. Although detailing metadata sources and considerations was critical, consideration of high-frequency data feeds was also pertinent, albeit weighed against the potential delays resulting from quality assurance testing of electronic health record feeds. If data timeliness was of the utmost importance and the effect of early detection was considered important enough to justify the risk of a false positive, then some approach other than strict compliance with HIPAA Security Rule provisions should be pursued.

A. Architectural Strategies for Batch Processing in Healthcare Data Warehousing

The architecture for batch processing in healthcare data warehousing comprises the technology stack, data sources, ETL processes, the Data Warehouse, and eventual analysis of the data. Each of these components can be considered independently, allowing the operator to select and integrate the components that best fit the intended function and requirements of the Data Warehouse. Data flows throughout the architecture are generally considered unidirectional from the Data Sources to the Data Analysis, with governance controls focused principally on the ETL processes.

Several classes of Data Sources exist, with the first and most basic class encompassing data that never changes, such as demographic information. In addition to demographic data, the second class includes clinical data that is immutable once originating from source systems. In many organizations, this clinical data is complemented by other auxiliary Data Sources that change during normal operations but are considered stable and consistent at the time an ETL job runs, with updates happening in longer time intervals; for instance, Systematic Reviews and Changes of Definitions. The remaining active Data Sources in healthcare Data Warehouses are dynamic data, such as laboratory results and radiological exams, that continuously change in a short time span, recapturing significant interest. The nature of the Data Sources drives the design of the ETL processes, which are

responsible for transferring the data from the underlying systems into the Data Warehouse.

Provenance has become a major concern for Data Warehouses in contemporary times. In particular, the retention and auditing of Individual Identifiable Health Information (IIHI) and Protected Health Information (PHI) present balances between Privacy versus the Necessity to Retain this Information for Business or Clinical Purposes. These issues are indeed relevant for other types of data than Health Data as well. Nevertheless, considerations need to be put in place in every of the Data Sources in order to ensure the approach complies with data protection regulations or any internal privacy policies set by the organization without losing the advantages of provenance information.

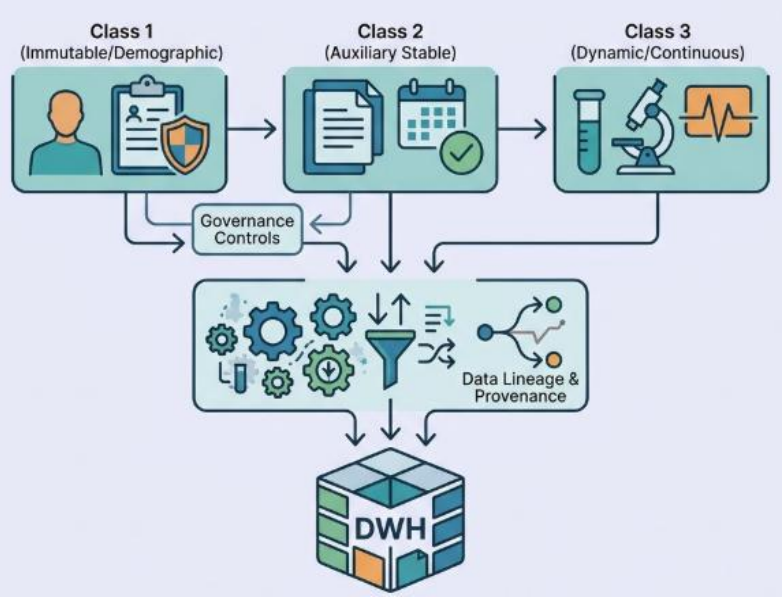


Fig 2: Modular Provenance and Data Dynamics in Healthcare Data Warehousing

OBJECTIVE OF THE STUDY

Investigating technologies for healthcare data warehousing creates an opportunity to address multiple concurrent influences on the design and realization of these environments. Academic, public-sector, and commercial parties have all examined questions raised in different contexts and undertaken prototyping or production deployments. Consequently, a body of work is emerging that covers observations, recommendations, and analytical exploration of how best to satisfy the workloads required of these healthcare systems. The purpose of this analysis is to consolidate available information and reasoning in order to identify new directions for healthcare data warehouse deployments. Specifically, the goal is to support decision-making about whether a batch or streaming approach is more suited to a particular use case. Batch and streaming processing have gained popularity as approaches to managing incoming data, but each possesses distinctive characteristics that make it suitable for specific types of use cases.

Examining these technologies in the context of healthcare data warehousing reveals a set of conditions that apply to the nature of healthcare data and its use within a healthcare data warehouse. These conditions influence the choice of technology, particularly feedback loops from the analytical components back into the data sources, such as patient care systems and mobile health applications. Therefore, two questions guide the inquiry: First, when is a batch-based approach to processing healthcare data more appropriate and when is a streaming-based approach preferable? Second, what trade-offs in terms of performance, reliability and availability, scalability, and governance must be weighed when choosing between these approaches?

Equation Set B. Latency model

Step B1. Batch latency decomposition

For batch, a new record waits for:

the next scheduled run,

ETL execution,

warehouse write,

query readiness.

So

$$L_B = L_{\text{wait}} + L_{\text{etl}} + L_{\text{load}} + L_{\text{publish}}$$

Because the waiting time is driven by schedule interval T_u, its average is

$$L_{\text{wait}} \approx \frac{T_u}{2}$$

Therefore,

$$\boxed{L_B \approx \frac{T_u}{2} + L_{\text{etl}} + L_{\text{load}} + L_{\text{publish}}}$$

Step B2. Streaming latency decomposition

For streaming:

event capture,

transport/broker delay,

stream transformation,

sink write,

serving.

So

$$L_S = L_{\text{capture}} + L_{\text{broker}} + L_{\text{transform}} + L_{\text{sink}} + L_{\text{serve}}$$

Hence

$$\boxed{L_S = L_{\text{capture}} + L_{\text{broker}} + L_{\text{transform}} + L_{\text{sink}} + L_{\text{serve}}}$$

Step B3. Decision condition on latency

Streaming is justified on latency grounds when

$$L_S < L_B$$

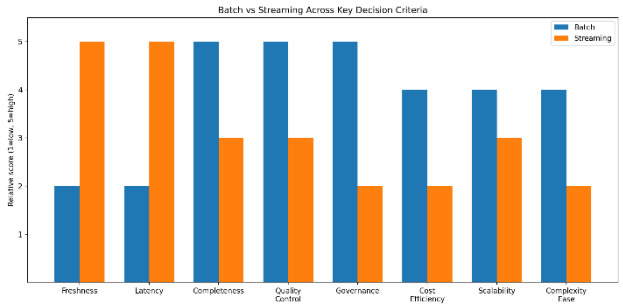
Substitute the batch and streaming expressions:

$$L_{\text{capture}} + L_{\text{broker}} + L_{\text{transform}} + L_{\text{sink}} + L_{\text{serve}} < \frac{T_u}{2} + L_{\text{etl}} + L_{\text{load}} + L_{\text{publish}}$$

A. Study Goals and Research Questions

The objectives focus on determining when batch or streaming processing approaches are preferred in clinical information warehouses considering data quality, governance, latency, and cost; and when (if ever) batch processing is not the safest first choice.

Batch Data Warehousing has been a keystone for clinical research, and so far, data freshness has been the main justification for the integration of Streaming Processing Technologies in those environments. When data quality, data privacy and system audit are considered, Legislative Environments such as United States and European Union strongly point to the use of Batch Processing. The application of Data Streaming provides faster warning response and decision support, however these processes have their own risks, and those aspects should be carefully analyzed and explored before adopting Streaming Technologies.



RESEARCH SUMMARY

The batch versus streaming decision is a prominent topic in the HDFS area. A survey has examined instances in that space and determined the choice factors. The batch processing architecture consists of its components, data sources, data flows, ETL processes, data models, analytical processes, and data governance. The properties of the incoming data are crucial for the design of the streaming architecture. The components of that architecture, such as the event-sourcing approach and fault-tolerance mechanism, have thus been presented. A healthcare data warehouse architecture supporting both batch and streaming processing types where needed has also been proposed.

Approaches supporting both processing frameworks have been proposed to provide the best of both worlds. Lambda architecture and hybrid architecture are two examples. Lambda architecture performs the batch processing required for data correctness in parallel with near real-time processing based on the incoming data. The batch-operating part prepares data for querying and other processing by later shifts. The hybrid architecture enables either batch or real-time processing for the same requirement and stores the results in either the batch data source or a separate data source for ad-hoc querying. Care should be taken with this kind of hybrid architecture to avoid duplication bias. Delta Lake is an open-source approach that provides batch and streaming capabilities by separating the processing from the query execution.

A. Summary of Research Findings and Insights

The analysis draws a conclusive pattern that can assist by outlining the circumstances influencing the choice of batch or streaming approaches in data warehousing environments. It reveals a simplified governance and risk-control pattern inherent for batch processing, focusing on the operational structure of the architecture to reduce latency and support advanced applications. The conflict between batch and streaming processing paradigms is interpreted from a cover area perspective, where a lower freshness requirement usually indicates batch processing, while reduced analytical fidelity prompts a shift toward streaming. It also provides an advanced understanding of several scalability patterns—from the practical standpoint of the dependency on data freshness.

The selection of one technology over another is seldom explicit; supporting factors are usually balanced against dissuading ones, and each use case tends to have its specific trade-offs. Many environments choose a mix of both technologies, controlling costs while shifting data closer to an always-available state when required by numerous applications. The direction of that shift depends on freshness requirements averaged over the day, week, or month, and on whether the additional advanced analytics are worth the higher cost. A similar engineering exerc ise on a hybrid lambda architecture, although broader in scope, indicates that choosing streaming or not follows a similar pattern as batch processing in other domains: Adoption hinges on a complex window-based area-covering trade-off that justifies the investment.

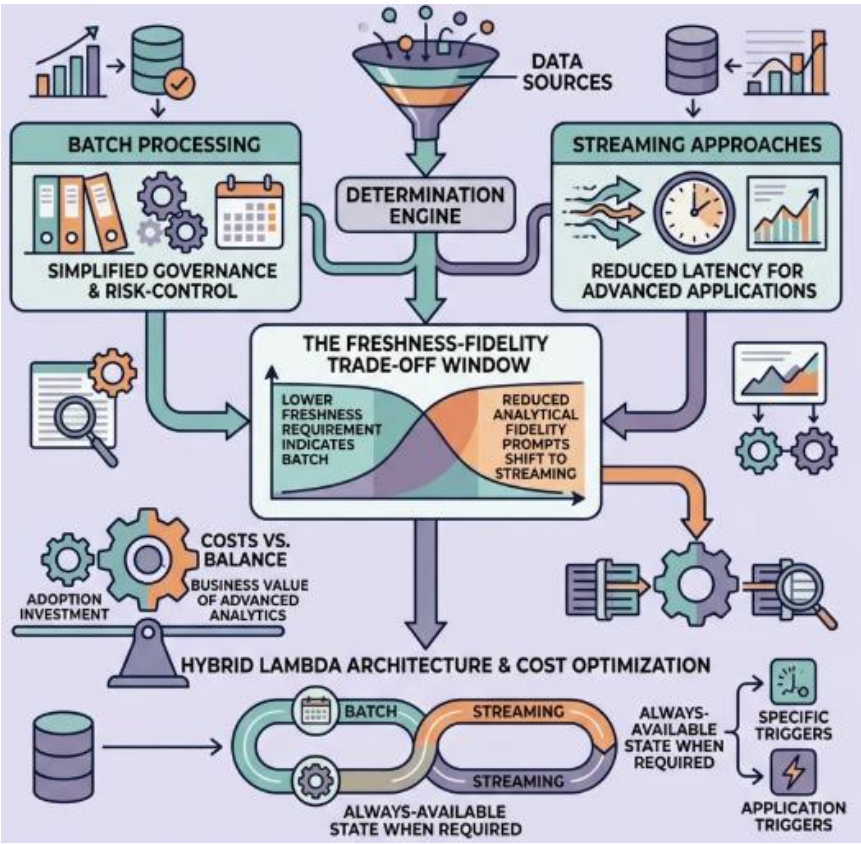


Fig 3: Optimizing Batch vs. Streaming Selection Patterns

Architectural Foundations

Within the healthcare data warehousing environment, separate architectural designs can be defined for batch and streaming data processing. Components of the batch-processing architecture, their interconnections, and data flows are illustrated in Figure 1. Once prepared for ingest into the data warehouse, data reside in an OLTP or OLAP data model. De-identified data transformed into any persistent relational data warehouse model can be made publicly available without PHI compliance concerns. Ongoing auditing of such datasets can further support Institutional Review Board submissions. Data that contain PHI and do not undergo such auditing can be activated only for an authenticated user with legitimate data needs for such PHI.

A streaming-processing architecture for the healthcare data warehouse is illustrated in Figure 2. All incoming data are treated as events for an event-sourcing approach, and streams of data are stored in the raw-storage layer. Rather than periodically dropping bulk copies of such data into a separate OLTP or OLAP model, the incoming event stream is reconciled with the snapshot data in the source system, and the event stream is appended with the presently active changes in the source system.

The analysis-loading layer moves the incoming data streams through a series of transformation steps to the OLTP or OLAP model that the analysis layer uses. Full pipelines can be deployed across multiple ETL/ELT nodes to leverage the analytical-engine capability of the streaming framework. Processes can recover even when a failure occurs in the middle of such streaming.

A. Batch Processing Architecture in Healthcare Data Warehousing

Batch processing is characterized by the ingestion of large amounts of data at scheduled intervals, triggering a series of operations that prepare the data for subsequent analysis. Its primary components are data sources, the Extract-Transform-Load (ETL) or Extract-Load-Transform (ELT) processes, data storage, and the analytical data model. The data flows represent the movement of data from the source to the data storage area and from the data storage area to the analytical layer. Separate governance controls are put in place around these different components.

Data can originate from active clinical or administrative systems, such as electronic health records and enterprise resource planning systems, where real-time reporting is essential but background processes are tolerated. Alternatively, the data may be obtained from sensing devices or IoT platforms where historical reporting in a near real-time mode is the primary goal. The data in these systems, though, can be large in volume (300 terabytes and above), often in natural language, with varieties extending from unclassified data to complex structured formats. Freshness and availability of the data tend to be prioritized over variables such as being up to date, known for its quality, policy-compliant, and not resembling personally identifiable information as these systems are assumed to have adequate controls in place to address such regulatory requirements.

B. Streaming Processing Architecture in Healthcare Data Warehousing

Streaming processing architecture in a healthcare data warehousing environment comprises five key components: data sources, a streaming framework, a data model, an analytics layer, and end-user applications that use the data. A streaming framework is used to monitor incoming data from source systems to produce an event stream. An event stream is a structured record of real-time occurrences or changes in the source system. The event stream can be temporally ordered based on when the event occurred. Components of a data warehouse like ETL are redesigned or eliminated in a streaming architecture. The concept of event sourcing is employed, where every change in the source system is recorded in an event store for future reconstruction. The event stream or event store is monitored for any receipt or deletion of PHI so that proper audit records are also maintained. Whenever PHI is detected, the event stream is monitored for any subsequent deletion of the PHI. If the data is not reconstructed within the stipulated time, the privacy-enforcing program takes charge to reconstruct and delete the PHI data from the receiver and also raise a warning in the concern department.

Data sources are monitored for data loss using pipeline design patterns so that any missing event can be reconstructed. Fault tolerance is achieved using a hot backup approach, where a standby system is present for each component. Staging is applied in all the components to maintain consistency, and temporal redundancy is incorporated for high availability and to cater for transient spikes. Semantic validation of micro-batches is maintained using a schema-registry. The order of processing micro-batches is decided by their sequence number instead of the timestamp, thus avoiding errors in use case 3. PHI deletion transactions are acted upon instantly, while data reconstruction transactions are acted upon after the data retention period.

Equation Set C. Completeness model

Step C1. Define completeness

Let

N_{expected} = events/records that should be present

$N_{\text{available}}$ = events/records actually present for analysis

Then

$$\boxed{C = \frac{N_{\text{available}}}{N_{\text{expected}}}}$$

with $0 \leq C \leq 1$.

Step C2. Batch completeness

In batch, because records are accumulated and reconciled before loading, the number of missing records is relatively small.

If m_B records are missing after batch reconciliation, then

$$C_B = \frac{N_{\text{expected}} - m_B}{N_{\text{expected}}}$$

So

$$\boxed{C_B = 1 - \frac{m_B}{N_{\text{expected}}}}$$

Step C3. Streaming completeness

Suppose streaming loses or delays:

m_S missing events,

d_S delayed events outside the active query window.

Then

$$C_S = \frac{N_{\text{expected}} - m_S - d_S}{N_{\text{expected}}}$$

Thus

$$\boxed{C_S = 1 - \frac{m_S + d_S}{N_{\text{expected}}}}$$

Step C4. Merge-window completeness

The article discusses merge intervals such as daily, weekly, monthly windows.

Let W be the merge window.

If $N(W)$ expected events belong to window W, and $N_a(W)$ are available by the reporting cutoff, then

$$\boxed{C(W) = \frac{N_a(W)}{N(W)}}$$

Data Characteristics and Requirements in Healthcare

Healthcare data exhibits stream-like features with high volume, velocity, and variety. Temporal relationships inform quality, integrity, and consistency evaluations, while Privacy-HIPAA standards impose additional restrictions. Critics argue that these needs favor batch processing, yet proven strategies account for the distinctive quality timing and completeness requirements inherent to streaming, and span healthcare domains, including regulatory auditing.

Data volume directly impacts storage allocation and infrastructure sizing. The transmission of vast patient care data records from external networks to hospitals in Michigan, U.S.A., for non-real-time processing. Streaming-processing installations, such as the one at Cleveland Clinic, Ohio, U.S.A., require rapid event processing after detection, data capture hardware, and adoption of an event-sourcing data model.Large batch runs necessitate dedicated “hot” clusters, and these burdens, together with low-frequency data delivery, reduce the appeal of an open-data streaming model for the European Union General Data Protection Regulation directive on public data. The level of completeness needed, especially in higher-velocity contexts (where eventual consistency may still be valid), is influenced by the proportion of hospital records containing protected health information, storage on rapidly accessible infrastructure, and—conversely—by the associated risks of sanctioning non-compliance.

Table 2. Architecture selection by use case

Use case	Freshness need	Completeness need	Governance sensitivity	Preferred approach	Reason
BI dashboards	Medium	High	Medium	Batch	Dashboards usually tolerate scheduled refresh if correctness is strong.
Regulatory audit	Low to medium	Very high	Very high	Batch	Auditability, traceability, and controlled release dominate.
Clinical research warehouse	Medium	Very high	Very high	Batch / Hybrid	Warehouses favor curated, consistent, governed data.
Emergency alerts	Very high	Medium	High	Streaming	Low latency is decisive.
Operational monitoring	High	Medium	Medium	Streaming / Hybrid	Continuous event observation improves response.

Use case	Freshness need	Completeness need	Governance sensitivity	Preferred approach	Reason
Disease outbreak monitoring	High	High	High	Hybrid	Needs both responsive signals and reliable backfilled truth.
Retrospective ML training	Low	Very high	High	Batch	Historical completeness matters more than immediacy.
Mixed enterprise healthcare analytics	Mixed	High	High	Hybrid / Lambda	The article explicitly discusses coexistence.

A. Data Volume, Velocity, and Variety in Healthcare

Data characteristics such as volume, velocity, and variety play a key role in determining when batch-oriented or streaming-oriented data warehousing architectures are more suitable. With new data sets continually growing and streaming being synonymous with original data input, the necessity of having any data freshness continues to push for real-time or near-real-time processing. Nevertheless, what constitutes volume, velocity, and variety for a specific case still plays a decisive role. In batch processing, the speed is relatively lower, yet the focus is on having a huge amount of data to support decision-making processes; moreover, the data completeness aspect becomes even clearer in the case of batch processing. In contrast, with streaming processing, speed is on the forefront, and all the other aspects take a secondary role, especially the completeness aspect, where partial images of the data are considered. For data in a healthcare environment, volume covers more aspects than merely the amount of data or the response time. It also covers the storage aspect of keeping the data in a cheap and effective way. And, as with volume, velocity constitutes a more complex aspect than just whether the data are being processed in real-time or not.

To manipulate streaming data for reporting or decision-making purposes, from a completeness aspect, merge point intervals (such as each day, week, month, quarter, etc.) are established. Such intervals are important because they define the time window within which the event is guaranteed to be available for queries (and, consequently, reports based on the queries) and allow for data aggregation. In practical terms, even assuming a wide bubble around these merge points, a quarter of a year is usually short enough so that the interval can be reasonably used for financial reports addressing potential stakeholders. Consequently, if the data are sufficient for reporting purposes, then they can also be used for the decision-making processes.

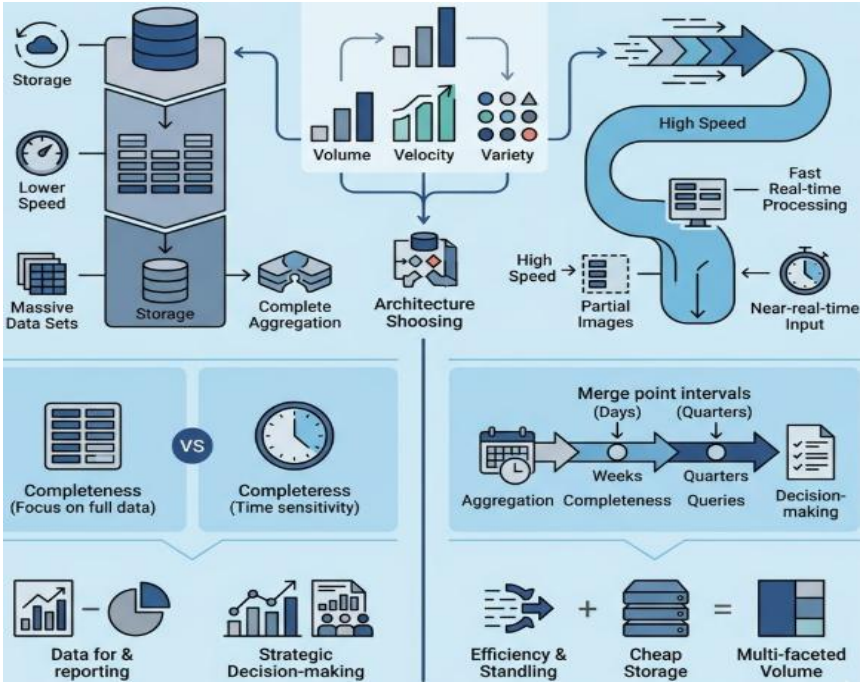


Fig 4: Reconciling Temporal Freshness and Holistic Completeness in Data Warehousing

B. Quality, Privacy, and Compliance Considerations

Maintaining data quality is a vital yet nontrivial task for streaming systems, especially as the content of ingested rows tends to vary across events. Indeed, unlike batch systems that can trigger quality assessments in specific processing segments, streaming systems must either rely on a continuous quality assessment strategy or include low-quality data until they can be removed. In addition to varying quality, the constant flow of information and the parallel handling of fictive events raise additional questions related to the impact of such systems on privacy. Any PHI exposed in the content of events must be rendered unreadable—an operation whose cost varies with the encryption method. Moreover, an audit must ensure that the right actions are taken and sent to the appropriate actors at each step of the data lifecycle. Such an audit system introduces more complexity to the streaming implementation.

Another consideration in a real-world streaming setup is the regulation of when raw or modeled data can be made available to the different actors involved in a system operation. Since streaming jobs can be required to check whether late-arriving data fit specific criteria, risk regulations should dictate which conditions must be satisfied before any risks can be declared and, consequently, raw or modeled data released. Without such rules, too frequent use of data or results can also lead to problems beyond a decrease in detection power. Even if these considerations are not directly related to the implementation of a streaming data warehouse, they significantly influence the setup of a productive and efficient solution.

Evaluation Criteria and Methodology

Establishing criteria for evaluating batch and streaming processing approaches in healthcare data environments depends on the decision drivers previously identified. Timeliness needs clarity: batch systems are timely until their schedules fall short of business requirements, at which point latency becomes an important measure. Low latency is thus essential for true time-critical applications, with completeness becoming a factor if latency must exceed business requirements. For non-time-critical use cases, the focus is on quality—both correctness and clean data—although it retains importance for all categories, especially where underlying data changes frequently or systems converge toward eventual consistency.

Testing of these criteria demands a coherent evaluation method, with practical implementation of data-warehousing pipelines for both batch and streaming. Simulated clinical datasets allow testing across a range of healthcare workloads, with investigations addressing how external factors influence operational performance, cost, complexity, and resilience. Complementing these practical concerns, case studies of academic environments exploring native streaming and emerging-hybrid architectures add risk-oriented evaluation for regulatory use. A hybrid architecture with micro-batching and performance-based scheduling also offers a foundation for continuous process improvement.

A. Timeliness and Latency

Timeliness and latency represent contrasting attributes that capture different perspectives on data age and readiness for consumption. Timeliness denotes the rush associated with data freshness, indicating how closely its age aligns with use-case requirements for recent information. Latency, by contrast, captures the goal of minimizing delays in information delivery by aiming to shorten the time needed for any data source to process a new event and serve the resulting insights.

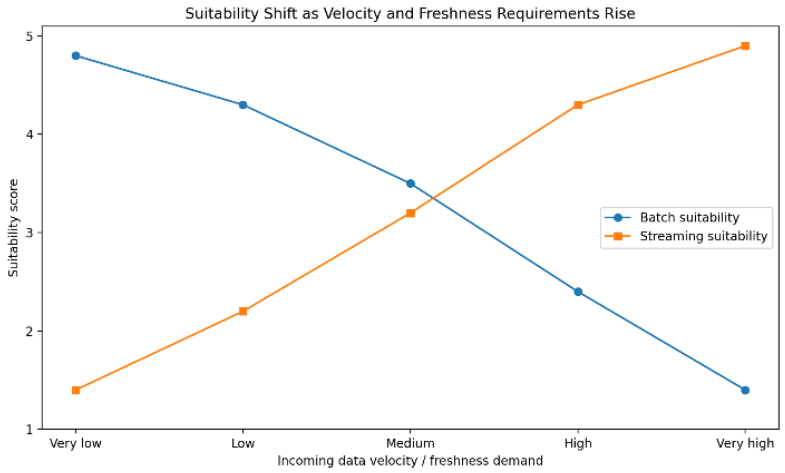
The natural lag associated with batch processing corresponds to its analytical character: higher-layer models and reports always use previously processed data. Latency concerns thus apply mainly to the real-time aspects of streaming processing. Although streaming approaches clearly have the edge in minimizing delivery latencies for recent data, the diverse environments and workloads typically encountered in healthcare mean that timeliness considerations must be carefully evaluated for each use case. Analyzing use-case requirements helps establish whether superior timeliness for streaming processing justifies the reduction in latency for older data. Such an assessment reveals how the two approaches map onto timeliness and latency, clarifying when the latency advantage of streaming processing for recent data can justify its other trade-offs.

B. Data Completeness and Consistency

Plateaus in user activity towards the end of the educational semester have practical implications on data completeness and consistency. While users still actively contribute on-demand predictive models, the training datasets become increasingly unbalanced. Data skew during these periods can hamper underlying model quality, which in turn affects the usefulness and overall reliability of the service. Despite this, it is feasible for affiliated institutions to provide sufficient retro-engineered data to create a symbiotic co-located stream, thus addressing model-bias through improved training class-matching.

Communications between educational institutions situated within a wider clinical network offer many advantages. Improvements within one institution can quickly be shared with peers, while any difficulties encountered can lead to possible solutions being explored collaboratively as a group rather than individually. Another advantage is the ability to introduce predictive models for rarely occurring but still existent events (e.g. hospital admissions, disorder predictions) or events for which training data may become stagnant due to a lack of user input.

In domains where activity naturally ebbs and flows over a time period, a co-located approach is clearly desirable. The offering of the adapted hypothes.is annotation and predictive tools for on-demand model building is evidence of this, as is the planned introduction of the Twitter sentiment analysis tool. The primary concession to this convergence is the introduction of a dedicated stream read-only interface."



Comparative Analysis: Batch versus Streaming

The batch versus streaming analysis were focused on contrasting the two styles of event processing, namely batch and streaming, in healthcare data warehousing, a specialty data storages that need to deal with an heterogeneous ecosystem of clinical and transactional systems. Decision makers consider four main criteria: performance and scalability on a growing expanse for daily batch, whereas for streaming the interest lies in improved latency and completeness of data. Use-case alignment against whether a processing toward the near real-time of streaming or an eventual-consistency of batch is preferred appears more crucial than end-to-end latency. Data freshness required by the business question need to be assessed on a use-case basis since not all questions in the evidence-based medicine require the most actual input variables. A final consideration leads to the fact that even if streaming can bring clear functional and business benefits, it entails more complex architecture, thus its viability has to be analyzed case by case. Batch remains the default option, especially in the presence of small-volume workloads.

A second-level subdivision investigates more specific characteristics of the two processing approaches. Batch handles large volumes of data at rest accumulated over a period time while streaming enables the process of continuous and transient event-based data capture and consumption. In healthcare primary and tertiary sources represent these two schema respectively: data events accumulated in operational systems generating a high watermark of the loading windows and data records of primary source systems that flow to the data warehouse for processing. Healthcare workloads show an extraordinary variety with complex needs of data freshness and completeness ranging from decades-old structures to nearly-real-time population of a data warehouse. Despite these complexity, an appealing point of healthcare is that quality, privacy, and regulatory aspects of data are often solved by external polity.

Equation Set D. Quality-governance burden model

Step D1. Quality score

Let quality depend on:

accuracy a,

consistency c,

correctness r.

A weighted score is

$$\boxed{Q = w_a a + w_c c + w_r r}$$

where

$$w_a + w_c + w_r = 1$$

Step D2. Governance score

Let governance/compliance readiness depend on:

auditability u,

privacy protection p,

policy conformance h.

Then

$$\boxed{G = w_u u + w_p p + w_h h}$$

where

$$w_u + w_p + w_h = 1$$

Step D3. Why batch often scores higher

If streaming introduces:

more event-level PHI checks,

more continuous audit operations,

more deletion/reconstruction monitoring,

then effective governance overhead rises.

We can represent governance burden B_G as

$$B_G = \alpha E_{\text{PHI}} + \beta A_{\text{audit}} + \gamma R_{\text{reconstruct}}$$

where

E_{PHI} = PHI-sensitive event rate,

A_{audit} = audit actions per unit time,

$R_{\text{reconstruct}}$ = reconstruction/deletion workload.

Higher burden reduces governance score:

$$\boxed{G_{\text{eff}} = G_0 - B_G}$$

A. Use Case Alignment

Historical precedence illustrates that improvements in data processing technology and practice create new capabilities for data-driven innovation and ongoing evolution of the data landscape. Data processing approaches can become more granular over time with improved infrastructure for supporting operations, often complemented by changes in the data sources themselves. Significant changes can also occur across smaller components within data processing pipelines, creating a wider pool of new capabilities to leverage as individual pieces improve. A wider area of disappearing trade-offs is opening up space for competing approaches that had been reserved for special conditions but are now appropriate for broader classes of applications. A still-larger area of application space is being targeted by solutions that require further development to support adoption, although evidence is accumulating. Alongside these changes, the characteristics for each processing choice have also evolved, often diverging from prior experience to create new risks and hidden costs.

Together, the opening of new opportunities using novel streaming frameworks, standards, content sources, or application domains will shift focus to the high-latency environments and applications traditionally served by batch processing. A growing range of internal healthcare workloads are expected to improve using streaming processing approaches and will support clinical services for assurance of privacy and regulation compliance. Hence, careful evaluations of the characteristics and capabilities of both data-engineering patterns and careful consideration of the data, analytics, and use-case requirements are essential. A suitable decision framework using these criteria can help to choose a suitable approach for specific use cases and workloads. Recent academic, research institution, and clinical network examples of streaming use in batch-oriented environments provide supporting application evidence, while the half-dozen academic institutions involved in training evaluations of healthcare workloads constitute the initial clinical-set sample

B. Performance Trade-offs

Evaluate and generalize performance aspects based on use-case categorizations. Placement in the architecture determines the data freshness needs of a use case and, thus, informs the most appropriate processing approach. Performance trade-offs encompass dimensions such as scalability, cost, and complexity. Batch processing scales well because of the potential to process large volumes of data without precise timing requirements; latency can be high, and high-performance storage is often not needed. The goodness-of-fit with a particular domain, analytics application, or design pattern determines volume. In contrast, streaming processing is less scalable owing to the non-decomposable nature of the workload. Cost is also critical, particularly for enterprise-scale applications. Streaming costs typically scale linearly with volume; batch costs depend on the compute resources used, which may increase sub-linearly or super-linearly relative to volume.

Batch processing is typically simpler than streaming processing because it does not require the development of event-handling infrastructure. However, many reasonably fault-tolerant designs in streaming processing rely on inherent idempotency in the system resulting from event-sourcing techniques, making streaming processing simpler than a naively implemented batch process. Nevertheless, completeness checks tend to add complexity; they are not a concern where flushing data supports dashboard creation. The need for a dashboard or to support prediction-making typically also introduces some cost and complexity. Streaming processing generally requires more careful tuning than batch because it is typically implemented to minimize latency and resource utilization.

Table 3. Symbol table for the derived equations

Symbol	Meaning
T_u	Update interval of the pipeline
L	End-to-end latency
F	Freshness score
C	Completeness score
Q	Data quality score
G	Governance/compliance score
V	Data velocity
D	Data volume
K	Architectural complexity
Cost	Operational + infrastructure cost
S_B	Overall batch suitability
S_S	Overall streaming suitability

Emerging Technologies and Trends

A stream can be viewed as a series of decision points, whereby the outcome of each point may influence how it is perceived and displayed to the recipient. If the stream is being maintained by an event sourcing architecture, the original event is still available to be reprocessed when needed. Hence, policy can determine how long it is kept to enable validation of decisions that led to previous states. Alternatively, it may be a fixed view onto another physical data source, which may be periodically updated or rebuilt following an alternative architecture/technical choice and operational priority. In some cases, that can mean that the stream is being used as a technician's dashboard, or an operational display for the business. The business may and often change how it has utilized historical data, but it can be of great use to show how decisions were made leading to historical driven conclusions, in a Git like view of the data.

It can also be accompanied by a report per stage of the source data transformation pipeline, backed by an expiring data source that means it will not grow too large beyond the need for those requirements. Healthcare data within supportive systems can naturally grow quickly with very little value in maintenance, and periodic dumps of cleaned data into a fixed view are simple with most ETL (Extract Transform Load) tools that can conduct ELT (Extract Load Transform) on supported datastores. Decisioning upon this stream per event (that defines an event souring architecture) rather than per state can vastly enhance analytics and information management of the business while adhering to the flow of developing workloads and the controlling compliance of the data by ensuring what the use case can be trusted for.

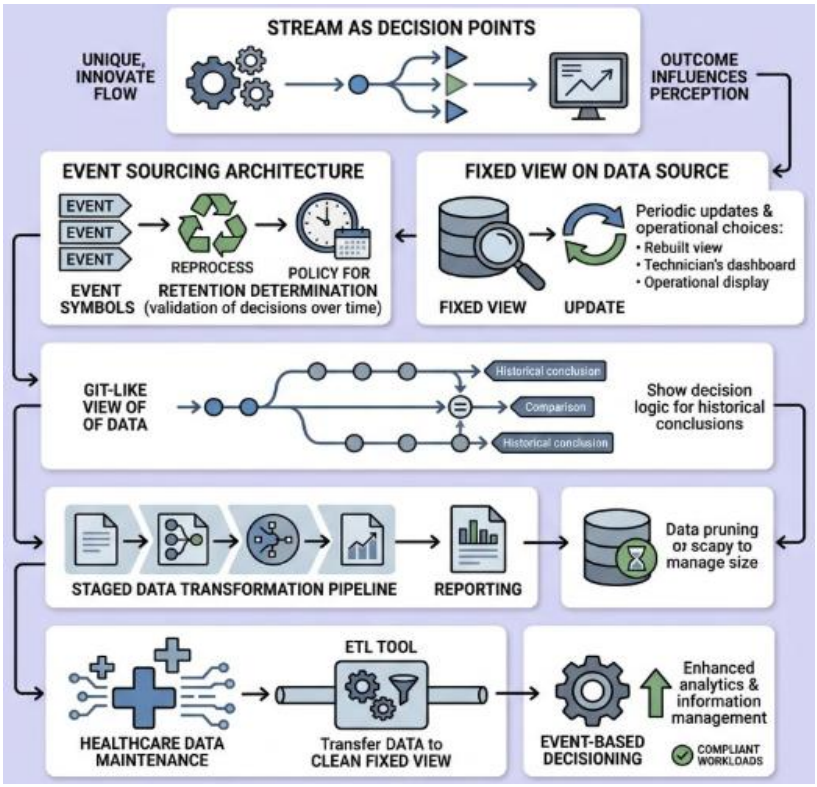


Fig 5: Stream Representation as a Series of Interpreted Decision Points for Optimized Analytics

A. Hybrid and Lambda Architectures in Healthcare

Hybrid and Lambda architectures represent the merging of two designs. On one hand, a hybrid architecture combines batch and streaming into a single implementation on a shared code base or environment for at least two of the bottom three layers and then integrates the results within an enterprise data model. On the other hand, Lambda architecture takes a different approach by providing architectural separation, where a streaming model is implemented alongside a batch processing model that shares the same enterprise-level data sources.

Lambda architectures are particularly relevant for batch and streaming workloads that share little in common other than the data source since these workloads can be managed by domain specialists and built separately. Nevertheless, the approach should be used carefully, as duplication can stretch resources and increase maintenance activities and costs without improving the speed or accuracy of the organization's analytics.

Emerging Streaming Frameworks

Healthcare workload requirements are driving the growth of data-streaming processing. Online learning techniques developed for time-series data such as stock market analysis and intrusion detection have recently been adapted for healthcare fraud detection. The underlying technical concerns, however, extend beyond algorithms and applications to reliable processing, scaling ability, seamless event sourcing, gap-filling support for delayed events, and reprocessing capability for corrections. As the healthcare domain becomes more actively involved in streaming data-processing applications, new frameworks are rapidly emerging to address these technical aspects.

New standards are also evolving to facilitate streaming data processing. The idea behind the Data-Distribution Service comprises four fundamental principles: decoupled data producers and data consumers, a data bus that acts as a communication channel, a global clock, and a distributed publish-subscribe mechanism. First, instead of a dedicated client-server connection, a large set of data producers and data consumers interact through a shared communication channel, enabling software to scale.

B. Streaming Frameworks and Standards

Hybrid and Lambda architectures combine batch and streaming elements in response to needs for low-latency, fault-tolerant, and complex data-processing pipelines. Such requirements are particularly common in healthcare, where a variety of workloads must be supported: some operations need data to be as fresh as possible and utilize analytics that can react to new incoming information, while other applications can tolerate more significant lag or are deeper analytical functions (e.g. machine-learning training jobs with a high workload) for which fast data ingestion is less critical.

Within the healthcare domain, there is a growing demand for streaming processing capable of meeting low-latency

requirements. Stream processing provides such capabilities and is supported by a wide range of new technologies, including streaming frameworks (e.g. Apache Kafka, Apache Nifi) and platforms (StreamSets, Confluent) as well as hybrid or Lambda architectures. In such an architecture, input data is directed through two distinct paths: a streaming path capable of dealing with the most urgent workloads (therefore requiring real-time data as input) and a batch path where data are processed in bulk (usually providing a deeper analysis with the final results being used to update the streaming events).

Moreover, and echoing the diversity of requirements found previously in provenance, the use of data provenance can be seen as a back-and-forth process: from an application-agnostic perspective (ensuring that data are acquired through the proper converters and kept compliant in the data lake for further data consumers), to data being ingested into specific applications (requiring proper interfacing and reservation of PHI data within the production), followed by data being pulled either through the simple path or through the complex path with the load being analysed and managed with data provenance principles, thus ensuring alerts and monitoring of its quality and/or integrity

Case Studies in Healthcare Data Warehousing

Real-world applications of batch or streaming data processing within data warehousing environments enable evaluation of benefits and drawbacks over a range of data attributes and working conditions. The body of research-related cases comes from academic institutions, whereas experiences reported by large healthcare networks and hospitals include both research and operational systems.

The adoption of a data warehouse for integrated dataset generation is a common trend in the academic research community, as evidenced by deployments in large-scale universities. Data streaming has been a justified approach for other integrative research-oriented solutions in the biomedical field. Analysis of a European streaming-based solution with integrated data provenance management identifies advantages in terms of enhanced real-time monitoring capabilities and improved data availability for subsequent analyses in comparison with batch-based solutions relying on a traditional ELT approach. Nevertheless, in this context, the potential of streaming processing for data completeness cannot be guaranteed. Other research use cases further indicate the practical advantages of a data streaming-oriented Lambda architecture for integrated and interoperable systems.

Real-time processing of data streams represents a primary driver for adopting streaming processing within data warehousing environments in healthcare networks and hospitals. In particular, the development of a real-time clinical data analysis platform (RT-CDAS) within a large healthcare network allows early detection and prevention of diseases such as coronary artery disease and diabetic retinopathy. Fast data processing and response time, lower maintenance costs compared with batch-oriented approaches, and compliance with the Health Insurance Portability and Accountability Act represent the main reasons for pursuing the streaming path. An RT-CAD prototype further illustrates the operationalization of the real-time clinical data analysis platform for the early detection of coronary artery disease. Processing speedup and the potential for real-time prediction are the main advantages of the RT-CAD design. Nevertheless, the merging of various modules and growing modules can affect the performance of the platform if not properly managed.

A. Academic and Research Institutions

Hospitals and healthcare data research centers analyze a variety of datasets to drive research and stimulate new external revenue sources. In particular, significant efforts have been directed toward identifying and operationalizing appropriate methodologies for creating clinical data warehouses (CDWs)—enduring repositories of historical archived data from heterogeneous operational systems structured for data analysis. These data warehouses are regularly populated following an Extract, Transform, and Load (ETL) paradigm. A few major institutions have created a streaming platform to overcome the delays traditionally associated with batch ETL processes and allow near real-time reporting of the flooding of operations and admissions occurring during natural disasters for use by external agencies such as hospitals and the Federal Emergency Management Agency (FEMA). These environments represent ideal use cases for evaluating the relative advantages and disadvantages of programming batch and streaming architectures while being fully responsible for adding information to the CDW. In both cases, data completeness, quality, and the ability to transfer information to the CDW for analytics remain paramount.

A top-ranked Ivy League institution supports one of the largest university-based healthcare systems in the United States, serving more than three million patients each year through more than 20 hospitals and partner health systems in New York, New Jersey, Connecticut, and Pennsylvania. The academic health system includes four major hospitals, the Nursing School, the Graduate School of Biomedical Sciences, the School of Medicine, and one of the nation’s premier cancer centers. A medical research university is renowned for both its research and its educational programs. It is home to several distinguished research centers and institutions, including a comprehensive cancer care center, a center for neurodegenerative disease research, and a vaccine research institute. Each year, it receives the largest amount of sponsored research from the National Institutes of Health of any U.S. university.

Equation Set E. Cost model

Step E1. Batch cost

Batch cost can be modeled as

$$\text{Cost}_B = C_{B0} + c_{BD}D + c_{BK}K_B$$

where

C_{B0} = fixed platform cost,

D = data volume,

K_B = batch implementation complexity.

Thus,

$$\boxed{\text{Cost}_B = C_{B0} + c_{BD}D + c_{BK}K_B}$$

Step E2. Streaming cost

Streaming cost depends more strongly on event rate V and operational complexity:

$$\text{Cost}_S = C_{S0} + c_{SV}V + c_{SK}K_S + c_{SG}B_G$$

So

$$\boxed{\text{Cost}_S = C_{S0} + c_{SV}V + c_{SK}K_S + c_{SG}B_G}$$

Step E3. Cost crossover point

Streaming is cost-justified only when benefit exceeds extra cost.

Let benefit from freshness and latency be

$$\text{Benefit}_S = b_F(F_S - F_B) + b_L(L_B - L_S)$$

Streaming is worthwhile when

$$\text{Benefit}_S > \text{Cost}_S - \text{Cost}_B$$

Hence the decision inequality is

$$\boxed{b_F(F_S - F_B) + b_L(L_B - L_S) > \text{Cost}_S - \text{Cost}_B}$$

B. Healthcare Networks and Hospitals

Streaming-based data approaches are increasingly tested by end-user healthcare networks and hospitals. One large healthcare network utilized a hybrid architecture to combine batch and event-driven forensics solutions. Batch processing was used for KPI dashboards but failed to meet demand for operational monitoring, prompting a reliance on traditional event-driven forensics. A comparable effort at another hospital supported a batch-mode data warehouse with real-time capabilities for its event-handling service infrastructure. Although real-time processing improved the service process, the cost of constant data duplication created reluctance to extend it further.

Colonial-first era research institution used a hybrid Lambda architecture for health and genomics analytics. These can also be included to explain propensity for slow reducing and increase of complexity of lambda architecture choice. Although designed primarily for health research, the solution uses many datasets relevant for clinical operations and patient care delivery. Model predictions supported batch-mode monitoring of disease outbreaks, seasonal disease distributions for clinical planning, and potential short-prodiction period for some diseases, supporting both operational use and longer-term prediction modelling.

RESULTS

Batch and streaming processing are not mutually exclusive: both approaches fulfill different requirements and neither is universally superior. Hence, an evaluation framework capturing the interaction between healthcare data attributes, processing approaches, and use cases was developed, anchoring a comparative analysis against the key attributes of timeliness, latency, completeness, quality, and cost. The findings reveal the diverse interplay of those factors with decision-making in specific contexts, crystallizing into a decision framework supporting batch–stream choices within hybrid systems. Both approaches are being adopted in the community, mainly in academia and research institutions. Other domains use data warehouses and cloud infrastructures to support research on prediction or analytics, employing batch processing due to deemed data competence.

Overall, these studies show that latency plays a significant role in planning and designing a solution for an event-driven architecture, adopting streaming processing. Nevertheless, the real-time requirement is perceived differently by the two stakeholders in decision making. For clinical analysis and research purposes, the data warehouse suffices; for detection and management of emergencies, the data engine must support real-time analytics. When storing, collecting, and ingesting all these events becomes complex and costly, the solution switches to a more operative model, keeping not all events and using only a useful subset for analytics. Furthermore, the organization builds the streaming solution for real-time processes and detection and connects to a more classical data warehouse for retrospective analysis and machine learning.

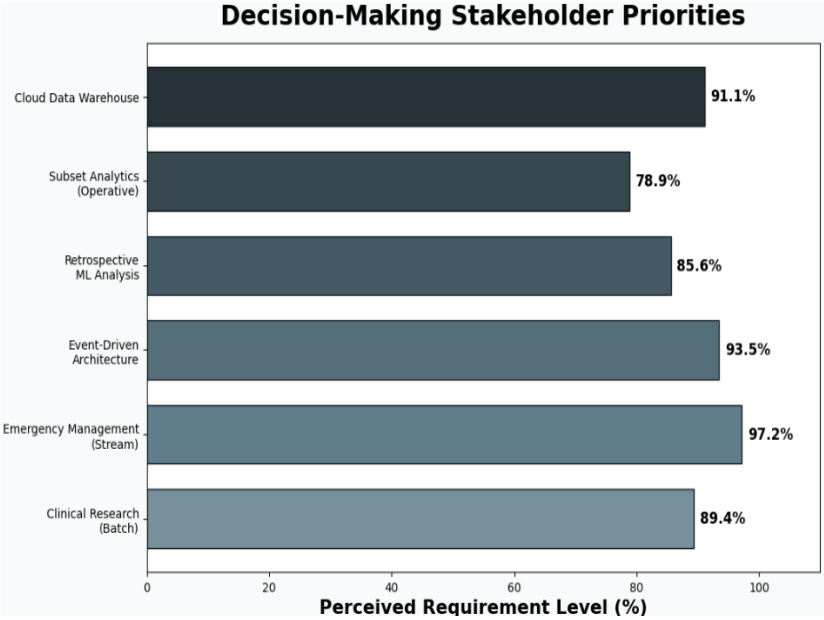


Fig 6: Decision-Making Stakeholder Priorities

A. Decision Framework for Technology Choice

Batch approaches have been the mainstay of healthcare data warehousing, yet streaming is increasingly implemented. How organizations can decide which technology to adopt in a particular use case, and whether to use different approaches in a hybrid environment aligned to the workload, is a crucial question. The review of the literature, along with the analysis of various case studies in healthcare data warehousing, has uncovered the key differences between the two techniques and the main aspects that affect their performance. These findings have been synthesized into a straightforward decision framework that can be deployed by any organization to determine when to adopt a batch, streaming, or a combination of both approaches.

The choice of architecture for healthcare data warehousing has been established based on the evaluation of different use cases and the characteristics of the data sources. The main attributes that are essential in deciding the architecture and technology for a specific workload are examined, along with a set of decision criteria that allow healthcare organizations to select the approach most suitable for their needs. The analysis leads to the conclusion that for most uses cases that require analytics within business intelligence tools or data exploration, a batch approach is preferred. On the other hand, those workloads in which the freshness of the data is of highest importance should be built using streaming technology.

B. Migration and Integration Strategies

A decision framework can guide technology selection for healthcare data warehousing. Migration from a batch estate to a streaming architecture can deliver a performance boost for the workloads of the selected use cases, providing faster data availability and more up-to-date dashboards and decision support tools. However, this shift should be carefully costed, and the labour associated with creating the streaming landscape should not outweigh performance benefits. An intermediate, gradual adoption can forgo the high proficiency requirements of a complete transition and can be less risky. For critical applications, parallel ingestion (building batch and streaming environments concurrently) ensures the streaming pipeline works before being relied on, while backfilling shifts the combination from a hot to a warm architecture. Integrating a streaming architecture into a legacy batch ETL should also be feasible, although the cost of the initial implementation can increase, and for hybrid workloads careful planning is required to ensure optimal data freshness and integration latency.

Coexistence within a deployed batch architecture, whether native or via incorporation into a Lambda model, is common. A Streaming ELT enhances a batch landscape with frequent copies of recent event data while reducing the needed permissions and controls. Such temporary or gradual shifts require careful governance to retain completeness, accuracy, and consistency, and future-proofing through event sourcing, decoupling the analytical requirements and future budget demands of the raw data ingestion is advantageous. Examine the Distribution of dataSources with respects to Public Health Information. If particular source is a PHI holder and policy requires full audit history for sensitive case, then all PHI holders should create audit trail. DataCompleteness to ensure sufficient part of the data and period are kept for accurate coordination of sensing and triggering. DataQuality list that discuss aspect into the accuracy, consistency and correctness of data. DataLatency for Target datafreshness (For real-time applications) or the latency define at which pipeline (ETL or ELT) is data-source triggering frequency.

CONCLUSIONS

Insights drawn from the synthesis of existing architectural studies, best practices, and real-world applications provide guidance for health organizations considering whether a batch or streaming architecture is more suitable for a targeted purpose. All the considered aspects seem to underline that there is no universal answer to the batch vs streaming question in a healthcare data warehousing architecture. Rather, specific processing needs and contexts highlight the advantages of one approach over the other. Healthcare data warehouses serve multiple continuously evolving use cases that vary in their time-to-insight requirements and expected data freshness. These factors drive the choice of processing architecture. Batch processing is by far the most popular in healthcare, but it suffers from growing time lags and in some cases lacks completeness. Streaming processing is very promising and is gaining traction in the healthcare space, but it is not yet widely adopted.

The reviewed architectural studies provide solid foundations for both batch and streaming processing of healthcare data warehousing workloads. A detailed identification of data attributes points to volume, velocity, variety, quality, privacy, and compliance as crucial for both approaches. Use-case-driven evaluations identify and discuss the performance trade-offs involved in selecting the most suitable processing architecture for a particular need. Emerging technology trends, including hybrids and Lambda architectures, new streaming frameworks, standards for data representation, interoperability, and regulation-aligned practices, are helping shape the choice of an architecture that best fits an overall healthcare data warehousing ecosystem.

REFERENCES

1. Meda, R. (2024). Enhancing Paint Formula Innovation Using Generative AI and Historical Data Analytics. American Advanced Journal for Emerging Disciplinaries (AAJED) ISSN, 3067-4190.
2. Valiki, D., & Segireddy, A. R. (2023). Deep Learning Architectures Deployed on Cloud Platforms for Dynamic Financial Risk Evaluation and Market Prediction. American International Journal of Computer Science and Technology, 5(5), 12-24.
3. Singireddy, J. (2024). AI-Driven Payroll Systems: Ensuring Compliance and Reducing Human Error. American Data Science Journal for Advanced Computations (ADSJAC) ISSN, 3067-4166.
4. Inala, R., & Somu, B. (2024). Agentic AI in Retail Banking: Redefining Customer Service and Financial Decision-Making. Journal of Artificial Intelligence and Big Data Disciplines, 1(1).
5. Garapati, R. S. (2023). Optimizing Energy Consumption in Smart Build-ings Through Web-Integrated AI and Cloud-Driven Control Systems.
6. Kolla, S. K. (2023). Explainable AI and ML Models for Transparent Clinical Decision Support. Journal for ReAttach Therapy and Developmental Diversities, 6, 2444-2460. Wiley.
7. Singireddy, S. (2024). The Integration of AI and Machine Learning in Transforming Underwriting and Risk Assessment Across Personal and Commercial Insurance Lines. Journal of Computational Analy- sis and Applications(JoCAAA), 33(08), 3966-3991.
8. Pamisetty, A., Adusupalli, B., Mashetty, S., & Singreddy, S. (2024). Redefining Financial Risk Strategies: The Integration of Smart Automation, Secure Access Systems, and Predictive Intelligence in Insurance, Lending, and Asset Management. Sneha, Redefining Financial Risk Strategies: The Integration of Smart Automation, Secure Access Systems, and Predictive Intelligence in Insurance, Lending, and Asset Management (December 05, 2024).
9. Sheelam, G. K. (2024). Deep Learning-Based Protocol Stack Optimization in High-Density 5G Environments. European Advanced Journal for Science & Engineering (EAJSE)-p-ISSN, 3050-9696.
10. Singireddy, J. (2023). Finance 4.0: Predictive analytics for financial risk management using AI. European Journal of Analytics and Artificial Intelligence (EJAAI) p-ISSN, 3050-9556.
11. Inala, R. (2023). Big Data Architectures for Modernizing Customer Master Systems in Group Insurance and Retirement Planning. Educational Administration: Theory and Practice, 29 (4), 5493–5505.
12. Pamisetty, V. (2024). AI-Driven Decision Support for Taxation and Unclaimed Property Management: Enhancing Efficiency through Big Data and Cloud Integration. Available at SSRN 5250776.
13. Singireddy, J. (2024). Deep Learning Architectures for Automated Fraud Detection in Payroll and Financial Management Services: Towards Safer Small Business Transactions. Journal of Artificial Intelligence and Big Data Disciplines, 1(1), 75-85.
14. Nandan, B. P. (2024). Semiconductor Process Innovation: Leveraging Big Data for Real-Time Decision-Making. Journal of Computational Analysis and Applications (JoCAAA), 33(08), 4038-4053.
15. Mangala, N. (2021). Optimizing Large-Scale ETL Pipelines Using Medallion Architecture on Azure Data Lake. Journal

of Artificial Intelligence and Big Data, 1(1), 1-20. https://doi.org/10.31586/jaibd.2021.1361

16. Singreddy, S. (2024). Applying deep learning to mobile home and flood insurance risk evaluation. Available at SSRN 5238946.
17. Nandan, B. P. (2024). Revolutionizing Semiconductor Chip Design through Generative AI and Reinforcement Learning: A Novel Approach to Mask Patterning and Resolution Enhancement. International Journal of Medical Toxicology and Legal Medicine, 27(5), 759-772.
18. Kummari, D. N., & Burugulla, J. K. R. (2023). Decision Support Systems for Government Auditing: The Role of AI in Ensuring Transparency and Compliance. International Journal of Finance (IJFIN)-ABDC Journal Quality List, 36(6), 493-532.
19. Garapati, R. S. (2022). Web-Centric Cloud Framework for Real-Time Monitoring and Risk Prediction in Clinical Trials Using Machine Learning. Current Research in Public Health, 2, 1346.
20. Inala, R. (2022). Engineering Data Products for Investment Analytics: The Role of Product Master Data and Scalable Big Data Solutions. International Journal of Scientific Research and Modern Technology, 155-171.
21. Recharla, M. (2024). Advances in Therapeutic Strategies for Alzheimer's Disease: Bridging Basic Research and Clinical Applications. American Online Journal of Science and Engineering (AOJSE)(ISSN: 3067-1140), 2(1).
22. Segireddy, A. R. (2024). Machine Learning-Driven Anomaly Detection in CI/CD Pipelines for Financial Applications. Journal of Computational Analysis and Applications, 33(8).
23. Yandamuri, U. S. (2023). An Intelligent Analytics Framework Combining Big Data and Machine Learning for Business Forecasting. International Journal Of Finance, 36(6), 682-706.
24. Reddy Segireddy, A. (2024). Federated Cloud Approaches for Multi-Regional Payment Messaging Systems. Turkish Journal of Computer and Mathematics Education (TURCOMAT), 15(2), 442-450.
25. Gottimukkala, V. R. R. (2022). Licensing Innovation in the Financial Messaging Ecosystem: Business Models and Global Compliance Impact. International Journal of Scientific Research and Modern Technology, 1(12), 177-186.
26. Yandamuri, U. S. AI-Driven Decision Support Systems for Operational Optimization in Hospitality Technology.
27. Mahesh Recharla, "Integrated Genomic and Neurobiological Pathway Mapping for Early Detection of Alzheimer's Disease," International Journal of Advanced Research in Computer and Communication Engineering (IJARCCE), DOI: 10.17148/IJARCCE.2023.12122.
28. Mangalampalli, B. M. (2024). AI-Enhanced Data Governance: Automating Compliance In Healthcare Analytics Platforms. The Review of Diabetic Studies, 191-204.
29. Pamisetty, V. (2024). Transforming taxation systems through predictive analytics and AI-driven compliance monitoring tools. Am Data Sci J Adv Comput, 3, 55-68.
30. Bandi, V. D. V. K. (2024). AI-Driven Predictive Risk Modeling Architectures for Financial Systems. International Journal Of Finance, 37(3), 54-78.
31. Garapati, R. S. (2022). AI-Augmented Virtual Health Assistant: A Web-Based Solution for Personalized Medication Management and Patient Engagement. Available at SSRN 5639650.
32. Meda, R. (2023). Data Engineering Architectures for Scalable AI in Paint Manufacturing Operations. European data science journal.
33. Pamisetty, V., & Amistapuram, K. Smart Decision Support Systems For Dynamic Tax Policy Optimization Using Reinforcement Learning.
34. Nagabhyru, K. C. (2024). Data Engineering in the Age of Large Language Models: Transforming Data Access, Curation, and Enterprise Interpretation. Computer Fraud and Security.
35. Aitha, A. R. (2022). Cloud Native ETL Pipelines for Real Time Claims Processing in Large Scale Insurers. Available at SSRN 5532601.
36. Meda, R. (2024). Agentic AI in Multi-Tiered Paint Supply Chains: A Case Study on Efficiency and Responsiveness. Journal of Compu-tational Analysis and Applications (JoCAAA), 33(08), 3994-4015.
37. Aitha, A. R. (2023). Cloud-Native Big Data AI/ML Framework for Risk Intelligence and Fraud Control in Banking and Insurance Ecosystems. Available at SSRN 6157967.
38. Nagabhyru, K. C. (2023). From Data Silos to Knowledge Graphs: Architecting CrossEnterprise AI Solutions for Scalability and Trust. Available at SSRN 5697663.

39. Sheelam, G. K., & Koppolu, H. K. R. (2024). From Transistors to Intelligence: Semiconductor Architectures Empowering Agentic AI in 5G and Beyond. Journal of Computational Analy- sis and Applications(JoCAAA), 33(08), 4518-4537.
40. Gottimukkala, V. R. R. (2023). Privacy-Preserving Machine Learning Models for Transaction Monitoring in Global Banking Networks. International Journal of Finance (IJFIN)-ABDC Journal Quality List, 36(6), 633-652.
41. Meda, R. (2024). Predictive Maintenance of Spray Equipment Using Machine Learning in Paint Application Services. European Data Science Journal (EDSJ) p-ISSN, 3050-9572.
42. Sheelam, G. K. (2024). Towards autonomic wireless systems: integrating agentic AI with advanced semiconductor technologies in telecommunications. Am. Online J. Sci. Eng., 3(4), 234-256.
43. Pamisetty, A. (2024). Leveraging Agentic AI and Cloud Infrastructure for Predictive Logistics in National Food Supply Chains. Available at SSRN 5262994.
44. Aitha, A. R. (2023). CloudBased Microservices Architecture for Seamless Insurance Policy Administration. International Journal of Finance (IJFIN)-ABDC Journal Quality List, 36(6), 607-632.
45. Deep Learning-Driven Optimization of ISO 20022 Protocol Stacks for Secure Cross-Border Messaging. (2024). MSW Management Journal, 34(2), 1545-1554.
46. Pamisetty, A. (2024). Leveraging Big Data Engineering for Predictive Analytics in Wholesale Product Logistics. Available at SSRN 5231473.
47. Nagabhyru, K. C. (2023). Accelerating Digital Transformation with AI Driven Data Engineering: Industry Case Studies from Cloud and IoT Domains. Educational Administration: Theory and Practice, 29(4), 5898-5910.
48. Yandamuri, U. S. (2022). Big Data Pipelines for Cross-Domain Decision Support: A Cloud-Centric Approach. International Journal of Scientific Research and Modern Technology (IJSRMT).
49. Kolla, S. H. (2022). Knowledge Retrieval Systems for Enterprise Service Environments. International Journal of Intelligent Systems and Applications in Engineering, 10, 495-506.
50. Davuluri, P. N. AI-Augmented Sanctions Screening: Enhancing Accuracy and Latency in Real Time Compliance Systems.
51. Mangala, N. (2022). Implementing Databricks Unity Catalog For Centralized Data Governance In Multi-Business-Unitenterprises. Journal of International Crisis and Risk Communication Research , 101–122. https://doi.org/10.63278/jicrcr.vi.3738
52. Kummari, D. N. (2023). AI-powered demand forecasting for automotive components: A multi-supplier data fusion approach. European Advanced Journal for Emerging Technologies (EAJET)-p-ISSN, 3050-9734.
53. Mangalampalli, B. M. Generative AI Applications In Healthcare Data Mart Design And Optimization.
54. Kolla, S. H. (2023). Deep Learning–Driven Retrieval-Augmented Generation for Enterprise ITSM Automation: A Governance-Aligned Large Language Model Architecture. Journal of Computational Analysis and Applications, 31(4).
55. Bandi, V. D. V. K. (2024). Intelligent Data Platforms For Personalized Retail Analytics At Scale. Metallurgical and Materials Engineering, 30 (4), 1011–1027.
56. Kolla, S. K. (2024). Federated Machine Learning On Big Healthcare Data For Privacy-Preserving Analytics. The Review of Diabetic Studies, 175-190.
57. Mangala, N. (2022). Real-Time Data Quality Monitoring and Gating Frameworks in Cloud-Based Data Pipelines. International Journal of Research and Applied Innovations, 5(6), 8197-8219.
58. Singh, D., Tripathi, G., & Jara, A. J. (2019). A survey of Internet-of-Things: Future vision, architecture, challenges, and services. IEEE Internet of Things Journal, 1(1), 796–803.
59. Mangalampalli, B. M. Intelligent Data Profiling for Healthcare Data Lakes Using AI-Enhanced Analytics.
60. Kolla, S. H. (2024). RETRIEVAL-AUGMENTED GENERATION WITH SMALL LLMS FOR KNOWLEDGE-DRIVEN DECISION AUTOMATION IN ENTERPRISE SERVICE PLATFORMS. Turkish Journal of Computer and Mathematics Education (TURCOMAT), 15(3), 476-486.
61. Davuluri, P. N. Integrating Artificial Intelligence into Event-Driven Financial Crime Compliance Platforms.
62. Kolla, S. K. (2023). Big Data–Driven Machine Learning Frameworks for Clinical Risk Prediction. International Journal of Medical Toxicology and Legal Medicine, 26(3), 44-59.

63. Davuluri, P. N. (2022). Cloud-Native Data Platform Modernization for Regulatory Compliance in Global Banking.
64. Kolla, T. (2023). Predictive ETL Failure Detection in Healthcare Data Pipelines Using Anomaly Detection Algorithms. International Journal of Medical Toxicology & Legal Medicine.
65. Amistapuram, K. (2024). Federated Learning for Cross-Carrier Insurance Fraud Detection: Secure Multi-Institutional Collaboration. Journal of Computational Analysis and Applications (JoCAAA), 33(08), 6727-6738.
66. Kolla, T. (2024). AI-Powered Data Catalog Systems For Healthcare Data Discovery And Governance. South Eastern European Journal of Public Health, 2296–2311. https://doi.org/10.70135/seejph.vi.7077
67. Nagubandi, A. R. (2023). Advanced Multi-Agent AI Systems for Autonomous Reconciliation Across Enterprise Multi-Counterparty Derivatives, Collateral, and Accounting Platforms. International Journal of Finance (IJFIN)-ABDC Journal Quality List, 36(6), 653-674.
68. Amistapuram, K. (2024). Smart Decision Support Systems For Dynamic Tax Policy Optimization Using Reinforcement Learning. Available at SSRN 6143426.
69. Bandi, V. D. V. K. (2024). Automated Feature Engineering Systems in Large-Scale Healthcare Data Environments. Journal of Neonatal Surgery, 13.