



e-ISSN: 2319-8753 | p-ISSN: 2347-6710

# IJIRSET

International Journal of Innovative Research in  
**SCIENCE | ENGINEERING | TECHNOLOGY**



# INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

Volume 11, Issue 11, November 2022

**ISSN** INTERNATIONAL  
STANDARD  
SERIAL  
NUMBER  
INDIA

**Impact Factor: 8.118**



9940 572 462



6381 907 438



ijirset@gmail.com



www.ijirset.com

# Engineering Real-Time Supply Chain Visibility and ETA Intelligence Platforms for Large-Scale Retail Distribution Networks

Naga Bhuvaneswari Chinnam

Senior Software Engineer, Sam's Club, USA

**ABSTRACT:** Large-scale retail distribution networks operate across highly dynamic environments in which transportation events, warehouse activities, inventory movements, carrier operations, customer demand, and external conditions continuously influence fulfillment performance. Traditional supply chain visibility platforms often depend on periodic data synchronization, manually maintained status information, and isolated operational systems, limiting their ability to provide timely and reliable insight into shipment progression and estimated time of arrival (ETA). This paper presents a generalized architectural approach for engineering real-time supply chain visibility and ETA intelligence platforms capable of operating across complex retail distribution ecosystems.

The proposed approach combines event-driven integration, streaming data pipelines, cloud-native microservices, location and transportation telemetry, data quality controls, and machine learning-based ETA intelligence. The architecture establishes a unified visibility layer capable of collecting and normalizing events from transportation management systems, warehouse management systems, order management platforms, carrier APIs, GPS or telematics providers, distribution centers, and external data sources. A streaming-oriented processing layer continuously evaluates shipment events, detects deviations, enriches operational context, and updates shipment state without requiring complete transactional-system synchronization.

The paper further examines ETA prediction as an intelligent decision-support capability rather than a static timestamp calculation. Historical transportation patterns, route characteristics, facility processing times, carrier performance, shipment attributes, traffic conditions, weather signals, and real-time operational events can be incorporated into predictive models to dynamically estimate arrival times and confidence intervals. The architecture also introduces mechanisms for model monitoring, event reconciliation, data lineage, observability, security, scalability, and human intervention to support reliable deployment across large retail networks.

By combining real-time integration with predictive intelligence, organizations can transition from retrospective shipment tracking toward proactive supply chain visibility. The resulting platform provides a scalable foundation for improving exception management, transportation coordination, distribution-center planning, customer communication, and operational decision-making while preserving interoperability with existing enterprise systems.

**KEYWORDS:** Supply Chain Visibility, Estimated Time of Arrival (ETA), Real-Time Analytics, Event-Driven Architecture, Retail Distribution, Transportation Intelligence, Predictive Analytics, Machine Learning, Streaming Data, Microservices, Cloud-Native Architecture, Event Correlation, Supply Chain Integration, Exception Management, Logistics Optimization.

## I. INTRODUCTION

Large-scale retail distribution networks have evolved into highly interconnected ecosystems involving suppliers, distribution centers, transportation providers, fulfillment facilities, stores, e-commerce channels, and customers. The movement of a single order may generate hundreds of operational events across warehouse management systems, transportation management systems, order management platforms, carrier applications, GPS and telematics systems, inventory platforms, and external data services. Although these systems provide substantial operational information, the information is frequently distributed across technology boundaries and processed at different time intervals. As a result, organizations can possess large volumes of supply chain data without having a continuously synchronized view of what is happening across the network.

One of the most important capabilities within a real-time visibility platform is estimated time of arrival (ETA) intelligence. Conventional ETA mechanisms may calculate arrival time using planned transportation schedules, static route information, or carrier-provided timestamps. These approaches can become less reliable when actual operating conditions differ from the original transportation plan. Delays at distribution centers, unexpected dwell time, traffic congestion, route changes, weather conditions, carrier performance variations, loading delays, and changes in shipment priority can all influence actual arrival time. Consequently, ETA should be treated as a continuously evolving prediction rather than a fixed value established when a shipment is initially dispatched.

A modern ETA intelligence platform can combine historical and real-time information to continuously refine arrival predictions. Historical shipment records can provide information about route-level travel patterns, facility processing durations, carrier behavior, and recurring delays. Real-time events can provide information about the current shipment location, transportation milestone, vehicle movement, facility status, and deviation from the planned journey. External information sources can further enrich the prediction process with contextual signals such as traffic conditions, weather conditions, road restrictions, and regional operating patterns. Machine learning and statistical prediction techniques can then use these signals to generate updated ETA estimates and associated confidence measures.

A generalized platform can therefore be organized into several architectural layers. The source layer represents transportation, warehouse, order, inventory, carrier, and external systems. An integration and ingestion layer receives APIs, messages, files, telemetry streams, and other operational signals. The streaming layer transports events in near real time and provides durable event processing capabilities. A normalization and enrichment layer transforms heterogeneous source data into a common supply chain event model. A visibility layer maintains the current operational state of shipments and orders, while an intelligence layer applies analytical and machine learning models to estimate arrival times and identify potential exceptions. Finally, user-facing applications and enterprise integrations expose actionable information to planners, transportation teams, distribution-center personnel, customer-service organizations, and other stakeholders.

The platform must also account for the difference between operational visibility and operational intelligence. Visibility answers questions such as *Where is the shipment?*, *What milestone was completed?*, and *What is the current status?* ETA intelligence extends these capabilities by addressing questions such as *When is the shipment expected to arrive?*, *How confident is the prediction?*, and *Has the expected arrival changed significantly enough to require intervention?* This distinction enables organizations to move from passive monitoring toward proactive exception management.

Observability is equally important for maintaining confidence in a real-time platform. Traditional application monitoring alone is insufficient because supply chain visibility depends on the complete movement of information from source systems through ingestion, streaming, processing, prediction, and presentation layers. Platform operators should therefore monitor event latency, processing failures, message backlogs, data freshness, duplicate rates, prediction errors, model drift, API performance, and end-to-end transaction latency. These measurements provide the operational foundation required to identify problems before they significantly affect downstream supply chain processes.

This paper presents a generalized engineering framework for developing real-time supply chain visibility and ETA intelligence platforms for large-scale retail distribution networks. The discussion focuses on architectural principles, event-driven integration, streaming data processing, shipment-state modeling, ETA prediction, exception intelligence, scalability, security, observability, and operational governance. The objective is not to prescribe a single technology stack but to establish reusable architectural patterns that can be adapted to different retail environments, transportation networks, cloud platforms, and enterprise integration landscapes.

The remainder of this paper examines the evolution and characteristics of real-time supply chain visibility, followed by the proposed platform architecture, event ingestion and processing mechanisms, unified shipment-state management, ETA intelligence architecture, predictive modeling considerations, exception management, scalability and resilience, security and governance, observability and operational monitoring, and implementation considerations. The paper concludes by describing how the integration of real-time event processing and predictive intelligence can provide a foundation for more responsive and data-driven retail distribution operations.

## II. EVOLUTION AND CHARACTERISTICS OF REAL-TIME SUPPLY CHAIN VISIBILITY

Supply chain visibility has progressively evolved from periodic status reporting toward continuously updated, event-driven operational intelligence. Earlier distribution environments were primarily dependent on batch interfaces,

scheduled data exchanges, manually entered milestones, and periodic carrier updates. These mechanisms were sufficient when transportation networks were relatively predictable and shipment information was primarily required for historical reporting. However, the increasing complexity of retail fulfillment has created a requirement for information that is both timely and operationally actionable.

#### ***A. From Periodic Tracking to Continuous Visibility***

Traditional shipment tracking commonly relied on predefined transportation milestones such as order creation, shipment confirmation, pickup, departure, arrival, and delivery. These milestones were typically transferred between systems through scheduled interfaces or batch files. Although this approach provided a basic representation of shipment progress, the resulting information could become stale between synchronization cycles.

For example, a shipment may depart a distribution center successfully but subsequently encounter congestion, an unexpected facility delay, or a route deviation. If the next system update occurs several hours later, transportation planners may continue to operate using an outdated expected arrival time. A real-time visibility architecture instead treats each operational change as an event that can immediately update the shipment's current state.

The transition can therefore be represented as:

**Batch Tracking → Event-Based Tracking → Real-Time Visibility → Predictive Visibility → Prescriptive Intelligence**

Each stage increases the timeliness and operational value of the information available to supply chain stakeholders.

#### ***B. Event-Centric Supply Chain Representation***

A real-time visibility platform should represent supply chain activity as a sequence of business events rather than simply as periodic snapshots. Events describe meaningful changes in operational state, such as:

- Shipment created
- Order released
- Load assigned
- Vehicle arrived at facility
- Loading started
- Loading completed
- Shipment departed
- GPS position updated
- Route deviation detected
- Distribution center arrival detected
- Unloading completed
- Delivery confirmed
- Shipment exception generated

Each event can contain identifiers, timestamps, locations, source-system information, event type, sequence information, and contextual attributes. A common event model allows the platform to transform heterogeneous messages from different transportation and enterprise systems into a consistent representation.

This approach also separates event production from event consumption. A carrier system, for example, can publish a departure event without needing to know which downstream applications will consume it. Visibility services, ETA engines, alerting systems, analytics platforms, and operational dashboards can independently subscribe to the relevant event streams.

#### ***C. Unified Shipment State***

Although individual events are valuable, operational users generally require a consolidated representation of the current shipment condition. A visibility platform therefore maintains a continuously updated shipment state derived from the event stream.

A generalized shipment state may include:

| Attribute           | Description                                          |
|---------------------|------------------------------------------------------|
| Shipment Identifier | Unique business identifier for the shipment          |
| Order Reference     | Associated customer or fulfillment order             |
| Origin              | Shipment origin facility                             |
| Destination         | Planned delivery location                            |
| Current Location    | Latest known geographic or operational location      |
| Current Milestone   | Most recently completed operational milestone        |
| Planned ETA         | ETA derived from the transportation plan             |
| Predicted ETA       | Dynamically calculated arrival estimate              |
| ETA Confidence      | Confidence or uncertainty associated with prediction |
| Carrier             | Transportation provider responsible for movement     |
| Route Status        | Current state of planned versus actual route         |
| Exception Status    | Active operational exception, if any                 |
| Last Event Time     | Time associated with the latest valid event          |
| Data Freshness      | Age of the latest available information              |

The distinction between **planned ETA** and **predicted ETA** is particularly important. Planned ETA represents the expected arrival based on the original transportation plan, whereas predicted ETA reflects the platform's current assessment using available operational and contextual information.

#### ***D. Sources of Real-Time Supply Chain Data***

Large retail networks typically generate visibility information from multiple independent sources. These sources can vary significantly in data format, communication protocol, frequency, and reliability.

Common sources include:

1. **Transportation Management Systems (TMS):** Provide shipment planning, carrier assignment, route, appointment, and transportation milestone information.
2. **Warehouse Management Systems (WMS):** Provide distribution-center activities such as picking, staging, loading, unloading, and shipment release.
3. **Order Management Systems (OMS):** Provide order lifecycle information and fulfillment relationships.
4. **Carrier Systems:** Provide pickup, departure, arrival, delivery, and exception information.
5. **GPS and Telematics Platforms:** Provide vehicle location, movement, speed, route, and other transportation signals.
6. **Enterprise Resource Planning Systems:** Provide inventory, purchasing, financial, and order-related information that can provide additional business context.
7. **IoT and Facility Systems:** Provide operational information from sensors, scanners, dock systems, and other connected equipment.
8. **External Data Providers:** Provide contextual information such as traffic, weather, road conditions, and transportation restrictions.

A generalized visibility platform must therefore support multiple ingestion patterns rather than assuming that every source can communicate through a single API mechanism.

#### ***E. Data Normalization and Canonical Event Models***

One of the fundamental engineering problems in multi-source visibility platforms is semantic inconsistency. Different systems may represent the same business event using different identifiers, terminology, timestamp formats, location models, and status codes.

For example, one carrier may represent a departure as DEPARTED, another as OUT\_FOR\_DELIVERY, while an internal system may use a numeric milestone code. A canonical event model can translate these source-specific representations into a common business vocabulary.

A generalized canonical event can contain fields such as:

**Event ID → Shipment ID → Event Type → Event Time → Ingestion Time → Location → Source → Sequence → Payload → Correlation ID**

Separating **event time** from **ingestion time** is particularly important. An event may occur at 10:05 AM but reach the visibility platform at 10:08 AM because of network delays or source-system processing. Treating both timestamps as identical can distort event ordering and latency measurements.

#### ***F. Real-Time Versus Near-Real-Time Processing***

Not every supply chain event requires identical latency requirements. A visibility platform should therefore classify workloads according to business criticality.

| Processing Category | Typical Characteristics    | Example                        |
|---------------------|----------------------------|--------------------------------|
| Real-Time           | Seconds-level processing   | Route deviation detection      |
| Near Real-Time      | Seconds to several minutes | Shipment ETA refresh           |
| Micro-Batch         | Periodic short intervals   | Carrier status synchronization |
| Batch               | Scheduled processing       | Historical analytics           |
| Offline             | Non-operational processing | Model training                 |

This classification allows organizations to allocate infrastructure resources according to business requirements rather than attempting to process every workload using the same latency model.

#### ***G. From Visibility to Predictive Intelligence***

A mature visibility platform extends beyond displaying the latest shipment status. Once a reliable stream of operational events has been established, analytical services can identify patterns and deviations.

For example, a platform can compare:

##### **Expected Progress vs. Actual Progress**

If a shipment was expected to complete a transportation milestone at a particular time but remains stationary beyond an acceptable threshold, the system can calculate a deviation. When combined with location, historical travel patterns, carrier behavior, and external conditions, the deviation can become an input to an ETA prediction model.

This creates a continuous intelligence loop:

**Event → State Update → Context Enrichment → Prediction → Exception Detection → Notification → Operational Action → New Event**

The resulting architecture changes supply chain visibility from a passive reporting capability into an active operational intelligence platform.

#### ***H. Characteristics of a Large-Scale Visibility Platform***

A production-scale retail visibility platform should exhibit several important characteristics:

##### **1) Scalability:**

The architecture should support increasing numbers of shipments, events, facilities, carriers, and users without requiring fundamental redesign.

##### **2) Low Latency:**

Critical events should propagate through the platform rapidly enough to support operational decision-making.

##### **3) Resilience:**

Temporary failures in individual services or source systems should not result in uncontrolled loss of business events.

##### **4) Interoperability:**

The platform should integrate APIs, message streams, files, databases, telemetry systems, and legacy applications.

##### **5) Data Quality:**

Duplicate, delayed, incomplete, or contradictory events should be detected and reconciled.

##### **6) Explainability:**

ETA predictions and operational alerts should provide sufficient contextual information for users to understand why a shipment's expected arrival changed.

##### **7) Observability:**

The complete event-processing path should be measurable from ingestion through prediction and presentation.

**8) Security:**

Access to shipment, location, carrier, customer, and operational information should be controlled according to organizational policies.

**9) Extensibility:**

New carriers, facilities, data sources, prediction models, and business rules should be incorporable without redesigning the entire platform.

**I. Architectural Transition**

The evolution toward real-time visibility can therefore be understood as a shift in architectural emphasis.

| Traditional Approach            | Modern Visibility Approach             |
|---------------------------------|----------------------------------------|
| Periodic batch synchronization  | Continuous event streaming             |
| Static shipment status          | Continuously calculated shipment state |
| Fixed ETA                       | Dynamically updated ETA                |
| Manual exception identification | Automated exception detection          |
| Isolated application data       | Unified event and context model        |
| Historical reporting            | Real-time and predictive analytics     |
| Point-to-point integration      | Event-driven integration               |
| Reactive intervention           | Proactive operational response         |

This transition establishes the foundation for engineering a platform in which operational data is continuously transformed into actionable information.

**J. Generalized Visibility Processing Model**

A generalized real-time supply chain visibility flow can be represented as:

**Source Systems → Event Ingestion → Event Validation → Normalization → Stream Processing → Shipment State → Context Enrichment → ETA Intelligence → Exception Detection → Operational Applications**

The architecture does not require every organization to implement identical technologies. Instead, the model provides a technology-independent pattern that can be mapped to different cloud providers, streaming platforms, integration frameworks, databases, machine learning services, and enterprise applications.

**III. REFERENCE ARCHITECTURE FOR REAL-TIME SUPPLY CHAIN VISIBILITY**

The reference architecture for a real-time supply chain visibility and ETA intelligence platform should provide a common technical foundation for collecting, processing, correlating, and presenting operational information across a distributed retail network. Because large retailers typically operate heterogeneous application landscapes, the architecture should emphasize loose coupling, standardized event contracts, horizontal scalability, and independent evolution of individual services.

A generalized architecture can be organized into **seven logical layers**:

**Data Sources → Integration & Ingestion → Event Streaming → Processing & Enrichment → Visibility & ETA Intelligence → Experience & Decision Support → Governance & Observability**

**Fig 1: Real-Time Supply Chain Visibility Architecture**

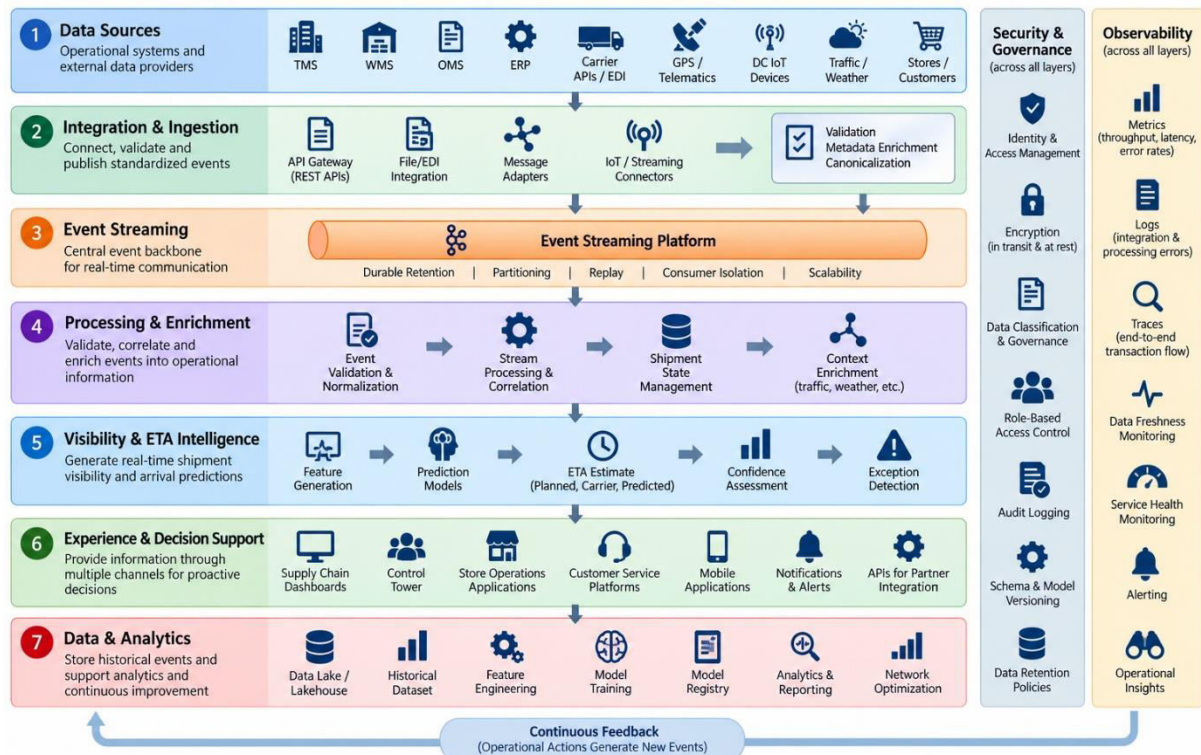


Fig. 1. Reference Architecture for Real-Time Supply Chain Visibility and ETA Intelligence

### A. Data Source Layer

The source layer represents the operational systems that generate information about orders, shipments, inventory, transportation, facilities, and external conditions.

Typical sources include:

- Transportation Management Systems
- Warehouse Management Systems
- Order Management Systems
- Enterprise Resource Planning systems
- Carrier APIs and electronic data interchange interfaces
- GPS and telematics platforms
- Distribution-center scanners and IoT devices
- Fleet-management systems
- Traffic and weather services
- Customer and store fulfillment systems

Each source may use a different integration protocol. Consequently, the architecture should avoid forcing all systems into a single communication pattern.

For example, a modern carrier platform may expose REST APIs, while an established logistics provider may continue to exchange structured files or EDI transactions. A distribution center may produce operational events through messaging infrastructure, while a telematics provider may continuously stream location information.

The integration layer abstracts these differences from downstream visibility services.

### B. Integration and Ingestion Layer

The ingestion layer provides controlled entry points into the platform. Its primary responsibility is to receive external information, perform initial validation, attach technical metadata, and publish standardized events to the streaming infrastructure.

Common ingestion mechanisms include:

**REST APIs → API Gateway → Ingestion Service**

**EDI/File → Integration Adapter → Event Translator**

### IoT/Telematics → Streaming Connector → Event Stream

### Enterprise Messaging → Message Adapter → Event Stream

The ingestion layer should avoid embedding extensive business logic. Instead, it should concentrate on protocol handling, authentication, schema validation, metadata enrichment, and reliable event publication.

A correlation identifier can also be assigned during ingestion to enable tracing of a business transaction across multiple services.

### C. Event Streaming Layer

The event streaming layer acts as the central communication backbone of the platform.

Instead of establishing direct point-to-point connections between every source and consumer, operational events are published to logical streams or topics. Multiple downstream services can consume the same events independently.

For example:

#### Shipment Departure Event

- Visibility Service
- ETA Prediction Service
- Exception Detection Service
- Notification Service
- Analytics Pipeline

This pattern reduces coupling and allows new consumers to be introduced without modifying the original event producer. The streaming layer should provide capabilities such as:

- Durable event retention
- Partitioning
- Consumer isolation
- Ordering within defined boundaries
- Replay capability
- Back-pressure handling
- Dead-letter processing
- Event acknowledgment
- Horizontal scalability

Partitioning strategies should be selected carefully. Shipment ID, order ID, facility ID, or geographic region can be considered depending on the required ordering semantics and workload distribution.

### D. Event Validation and Normalization

Events received from external systems cannot automatically be assumed to be complete or consistent.

A validation service can verify:

- Required identifiers
- Timestamp format
- Event type
- Source authenticity
- Schema version
- Geographic information
- Shipment relationship
- Data completeness

Invalid events should not simply disappear from the processing pipeline. They can be routed to a controlled quarantine or dead-letter mechanism for investigation and possible replay.

After validation, source-specific representations are transformed into a canonical event model.

For example:

#### Carrier-specific status

LOAD\_COMPLETE

may be translated into:

#### Canonical Event

SHIPMENT\_LOADING\_COMPLETED

Similarly, different location representations can be normalized into a common geographic structure.

### ***E. Stream Processing and Event Correlation***

The stream-processing layer converts individual events into meaningful operational information.

A single event may not be sufficient to determine the state of a shipment. The platform may need to correlate several events across time and systems.

For example:

**Order Released → Shipment Created → Load Assigned → Loading Completed → Departure → GPS Movement → Facility Arrival**

Together, these events describe the progression of a shipment.

The processing layer can perform:

- Event correlation
- Event sequencing
- Duplicate detection
- State transitions
- Time-window calculations
- Threshold evaluation
- Context enrichment
- Route deviation analysis
- Facility dwell analysis

This layer is therefore responsible for converting a raw stream of messages into operationally meaningful state.

### ***F. Shipment State Management***

The platform should maintain a current representation of each shipment independently from the historical event stream.

A shipment-state service can maintain attributes such as:

**Shipment ID → Current Location → Current Milestone → Planned ETA → Predicted ETA → Carrier → Route Status → Exception Status → Last Event**

This state can be stored in a low-latency operational data store for dashboard and API access.

The historical events remain available for audit, analytics, replay, and machine learning, while the current-state store provides rapid access to the latest operational information.

This separation is useful because real-time applications generally require the current state rather than repeatedly reconstructing the entire shipment history.

### ***G. Context Enrichment Layer***

ETA intelligence becomes significantly more useful when shipment events are combined with contextual information.

Potential enrichment signals include:

- Historical route duration
- Carrier performance
- Distribution-center processing time
- Traffic conditions
- Weather conditions
- Route characteristics
- Shipment priority
- Vehicle characteristics
- Appointment windows
- Historical facility dwell time
- Regional transportation patterns

The enrichment service can associate these signals with the relevant shipment before the prediction service evaluates the ETA.

For example:

**Current Shipment Location + Destination + Historical Route Time + Traffic + Facility Dwell Pattern**

can provide a richer prediction context than shipment location alone.

### ***H. ETA Intelligence Layer***

The ETA intelligence layer generates and continuously updates arrival predictions.

A generalized prediction pipeline can be expressed as:

**Operational Events → Feature Generation → Prediction Model → ETA Estimate → Confidence Assessment → ETA Publication**

The prediction service can use a combination of statistical methods, machine learning models, business rules, and deterministic transportation calculations.

A prediction may include:

- Estimated arrival timestamp
- Prediction confidence
- Expected delay
- Baseline ETA
- Prediction timestamp
- Model version
- Major contextual factors
- Prediction quality indicators

The system should retain the distinction between **planned ETA**, **carrier ETA**, and **platform-generated ETA**. These values may serve different operational purposes and should not be silently overwritten.

***I. Exception Detection and Decision Support***

An ETA prediction becomes operationally valuable when significant deviations generate actionable information.

An exception engine can evaluate conditions such as:

**Predicted ETA – Planned ETA > Threshold**

or:

**Actual Progress < Expected Progress**

or:

**Current Location outside Expected Route**

or:

**Facility Dwell Time > Normal Range**

Based on configurable business policies, the platform can classify exceptions by severity and route them to appropriate operational teams.

Examples include:

- Potential late delivery
- Route deviation
- Excessive facility dwell
- Missing shipment event
- Stale location data
- Carrier performance anomaly
- Delivery-window risk

The objective is not to generate alerts for every deviation but to identify events that require human or automated intervention.

***J. Experience and Consumption Layer***

The experience layer exposes information through multiple channels.

Possible consumers include:

- Supply chain dashboards
- Transportation control towers
- Distribution-center applications
- Store operations applications
- Customer-service platforms
- Mobile applications
- Notification systems
- Enterprise reporting platforms
- External customer portals

APIs can provide shipment-specific information to downstream applications, while dashboards can present network-level operational conditions.

A control-tower interface, for example, may provide:

**Network Overview → Facility Status → Shipment Exceptions → ETA Risk → Shipment Details**

This allows users to move from high-level network visibility to individual shipment analysis.

### ***K. Data and Analytics Layer***

Historical events should be retained independently from operational state so that organizations can perform deeper analysis.

A typical analytical architecture can include:

**Event Stream → Data Lake/Lakehouse → Historical Dataset → Feature Engineering → Model Training → Model Registry**

Historical information can support:

- Carrier performance analysis
- Route analysis
- Facility dwell analysis
- ETA model training
- Model validation
- Operational trend analysis
- Capacity planning
- Network optimization

The same historical dataset can also support retrospective investigation when operational teams need to determine why a shipment experienced a significant delay.

### ***L. Security and Governance Layer***

Security should operate across every architectural layer rather than being implemented only at the API boundary.

Important controls include:

- Identity and access management
- API authentication and authorization
- Encryption in transit
- Encryption at rest
- Role-based access control
- Service-to-service authentication
- Secrets management
- Audit logging
- Data classification
- Schema governance
- Data retention policies

Governance should additionally cover event ownership, schema versioning, model versions, prediction lineage, and access to sensitive operational information.

### ***M. Observability Layer***

A real-time platform requires visibility into its own behavior.

Observability should cover three dimensions:

#### **Metrics:**

Event throughput, processing latency, API latency, consumer lag, prediction latency, error rates, and data freshness.

#### **Logs:**

Validation failures, integration errors, processing exceptions, authentication failures, and model-service errors.

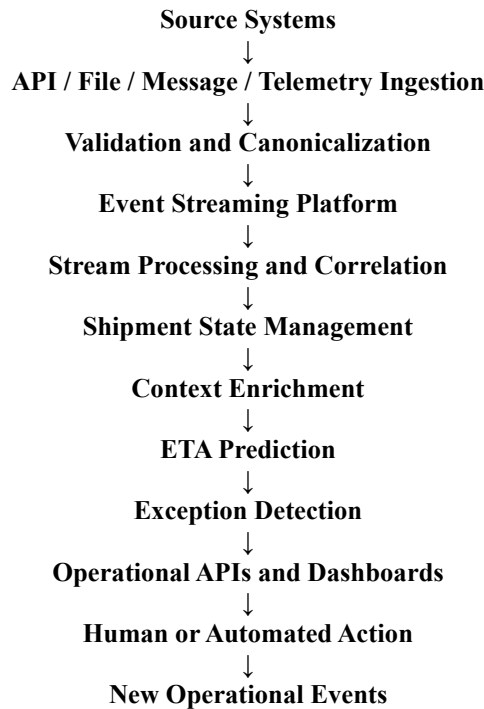
#### **Traces:**

End-to-end movement of a business transaction from source ingestion through event processing, state update, prediction, and downstream consumption.

A useful operational metric is **event-to-visibility latency**, representing the time between the occurrence of a business event and its availability in the operational visibility layer.

#### *N. End-to-End Reference Flow*

The complete architecture can be summarized through the following processing sequence:



The final stage creates a continuous feedback loop. Operational actions can generate new events, which are subsequently processed by the same platform.

#### *O. Architectural Design Principles*

The proposed architecture is based on several principles:

1. **Event-first integration** to reduce dependency on synchronous point-to-point communication.
2. **Loose coupling** between event producers and consumers.
3. **Canonical data representation** to manage heterogeneous source systems.
4. **Separation of current state and historical events** for efficient operational and analytical workloads.
5. **Continuous prediction** rather than one-time ETA calculation.
6. **Configurable exception policies** rather than hard-coded operational thresholds.
7. **Independent scalability** of ingestion, processing, prediction, and presentation services.
8. **Built-in observability** across the complete event lifecycle.
9. **Security by design** across APIs, services, data stores, and analytical components.
10. **Progressive modernization** that allows existing enterprise systems to participate without immediate replacement.

### IV. EVENT INGESTION, STREAM PROCESSING, AND REAL-TIME DATA ENGINEERING

The effectiveness of a real-time supply chain visibility platform depends heavily on its ability to transform heterogeneous operational signals into reliable, low-latency events. Large retail distribution networks generate data continuously from transportation systems, warehouses, carriers, vehicles, stores, fulfillment centers, and external services. The event-processing architecture must therefore accommodate different data formats, event frequencies, delivery guarantees, and source-system behaviors while maintaining a consistent representation of supply chain activity.

Fig 2 : Real-Time Event Streaming Architecture

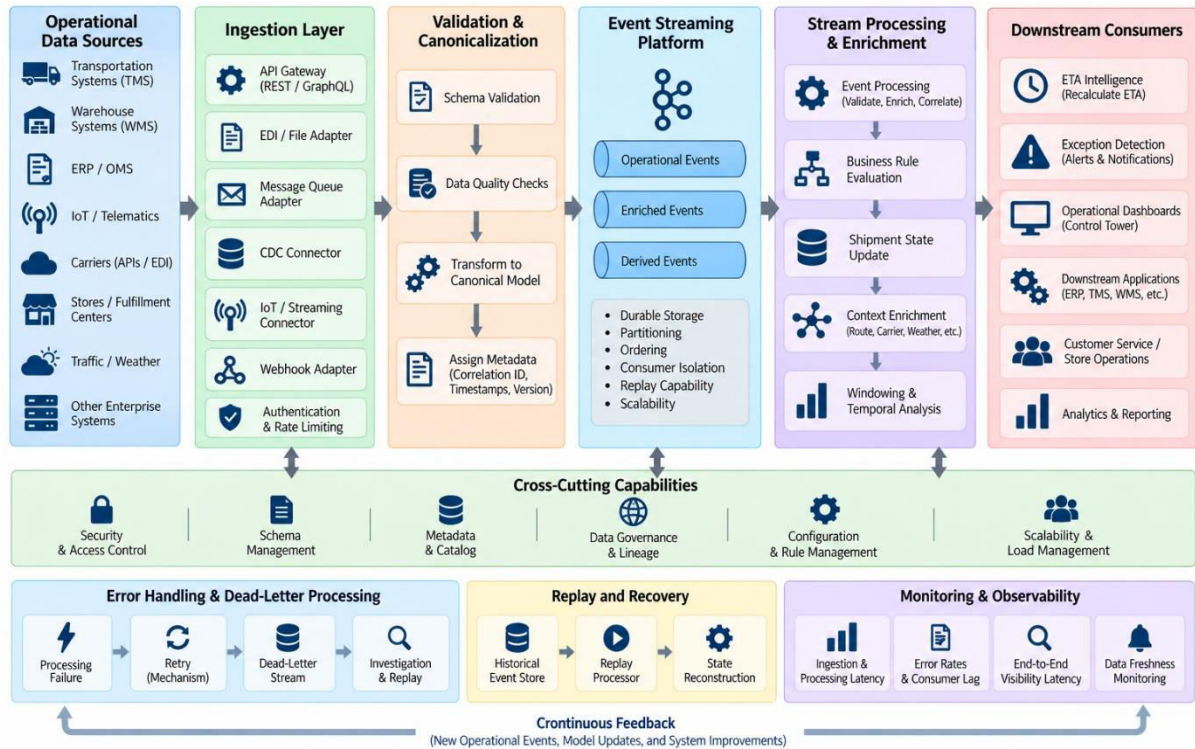


Fig. 2. Event Ingestion, Stream Processing, and Real-Time Data Engineering Architecture

### A. Multi-Source Event Ingestion

A production-scale platform should provide multiple ingestion mechanisms rather than relying on a single interface technology. Different participants in the supply chain may expose different integration capabilities.

Typical ingestion patterns include:

- REST and GraphQL APIs
- Message queues
- Event streams
- EDI transactions
- Secure file transfers
- Database change data capture
- IoT and telematics streams
- Webhooks
- Scheduled enterprise interfaces

An ingestion gateway can provide a controlled boundary between external systems and internal processing services. Authentication, schema validation, rate limiting, request tracing, and basic payload validation can be performed at this boundary.

The ingestion architecture can be generalized as:

**Operational Sources → Integration Adapters → Validation → Event Transformation → Streaming Platform**

This design allows new carriers or systems to be integrated through adapters without modifying the downstream ETA and visibility services.

### B. Event Contract and Canonical Data Model

A common event contract is essential when information originates from multiple systems. Without normalization, downstream services would need to understand the unique data structures and terminology of every source.

A generalized event structure may contain:

| Field          | Purpose                                   |
|----------------|-------------------------------------------|
| Event ID       | Uniquely identifies the event             |
| Event Type     | Identifies the business event             |
| Shipment ID    | Associates the event with a shipment      |
| Order ID       | Associates the event with an order        |
| Event Time     | Time when the business event occurred     |
| Ingestion Time | Time when the platform received the event |
| Source System  | Identifies the originating application    |
| Location       | Geographic or facility location           |
| Sequence       | Supports event ordering                   |
| Correlation ID | Enables end-to-end tracing                |
| Schema Version | Supports controlled contract evolution    |
| Payload        | Business-specific event attributes        |

The use of explicit schema versions allows producers and consumers to evolve independently. Backward-compatible changes can be introduced without immediately requiring every downstream application to be upgraded.

### C. Event-Time and Processing-Time Semantics

Real-time supply chain processing must distinguish between **event time** and **processing time**.

Event time represents when an operational activity actually occurred, whereas processing time represents when the platform processed the corresponding message.

For example:

**Shipment Departure:** 10:05:00

**Event Received:** 10:07:15

**Event Processed:** 10:07:18

The three timestamps describe different stages of the information lifecycle.

This distinction becomes particularly important when messages arrive out of order. A shipment may generate an arrival event after a departure event, but network delays could cause the arrival message to reach the platform before the departure message. Stream-processing logic must therefore support controlled event-time processing rather than blindly assuming that message arrival order represents business order.

### D. Event Ordering and Late-Arriving Data

Distributed supply chain systems frequently produce late-arriving or out-of-order events.

Consider the following sequence:

**Departure → Location Update → Arrival**

If the location update is delayed, the platform may receive:

**Departure → Arrival → Location Update**

A simplistic state engine could incorrectly move the shipment backward after receiving the delayed location event.

To prevent this, the platform can use event timestamps, sequence numbers, milestone precedence rules, or state-transition validation. Late events can either update historical information without changing the current state or trigger controlled state reconciliation.

This is particularly important for ETA calculations because an incorrect shipment state can propagate directly into an incorrect arrival prediction.

### E. Duplicate Detection and Idempotent Processing

Duplicate events are another common characteristic of distributed integrations. Retries, network failures, producer acknowledgments, and message redelivery can result in the same business event being delivered multiple times.

The platform should therefore support idempotent processing.

A simplified mechanism is:

#### Event ID → Duplicate Check → Process Once → Persist Processing Result

If the same event is received again, the system can recognize the previously processed identifier and avoid performing the business operation a second time.

Idempotency is especially important for events associated with inventory movement, delivery confirmation, payment-related operations, or other transactions where repeated processing could create inconsistent downstream state.

#### F. Stream Processing Architecture

After ingestion, events are published into durable streams. Stream-processing services can then perform real-time transformations and state calculations.

A generalized processing sequence is:

**Consume Event → Validate → Enrich → Correlate → Apply Business Rules → Update State → Publish Derived Event**

Derived events are useful because they allow complex operational conditions to be represented explicitly.

For example:

##### Raw Events

Shipment departure + GPS movement + route information

↓

##### Derived Event

SHIPMENT\_PROGRESS\_CONFIRMED

Similarly:

Facility arrival + extended dwell duration

↓

##### Derived Event

FACILITY\_DWELL\_EXCEPTION

These derived events can be consumed by ETA services, alerting services, and operational dashboards.

#### G. Event Correlation

Supply chain events frequently need to be correlated across multiple business identifiers.

A single shipment may be associated with:

**Customer Order → Fulfillment Order → Shipment → Load → Vehicle → Route → Delivery Appointment**

Different systems may use different identifiers for these objects.

A correlation service can maintain relationships between these identifiers, allowing events from multiple systems to be associated with a common business transaction.

This enables the platform to answer questions such as:

- Which customer orders are affected by a delayed vehicle?
- Which shipments are associated with a delayed distribution-center operation?
- Which deliveries are affected by a route deviation?
- Which stores may be impacted by a transportation exception?

#### H. Real-Time Context Enrichment

Raw events often contain insufficient information to make an operational decision. Context enrichment adds information from other services or data stores.

For example:

##### GPS Event

Vehicle = V123

Latitude = ...

Longitude = ...

can be enriched with:

**Vehicle → Shipment → Route → Destination → Delivery Window → Historical Travel Time**

The resulting context becomes much more useful for ETA calculation and exception detection.

Enrichment should be implemented carefully because excessive synchronous calls can increase processing latency. Frequently used reference information can therefore be cached or maintained in low-latency stores.

#### I. Streaming Windows and Temporal Analysis

Many supply chain conditions are inherently time-based.

For example, an organization may want to identify:

- Vehicles stationary for more than a defined duration.

- Shipments without an expected milestone within a time window.
- Facilities experiencing unusually long dwell times.
- Multiple route deviations within a short period.

Streaming windows allow events to be evaluated over defined periods.

A simplified model is:

**Event Stream → Time Window → Aggregation → Threshold Evaluation → Exception Event**

Both fixed and sliding windows can be used depending on the operational requirement.

#### ***J. Backpressure and Load Management***

Large retail networks may experience significant variations in event volume. Promotional periods, holidays, weather disruptions, or large-scale operational incidents can produce event bursts.

A streaming architecture should therefore support backpressure and workload isolation.

When event volume increases, the platform can scale processing consumers horizontally while allowing durable queues or streams to absorb temporary differences between producers and consumers.

This prevents short-term event spikes from immediately overwhelming downstream services.

#### ***K. Error Handling and Dead-Letter Processing***

Not every event can be successfully processed on the first attempt.

Errors can arise from:

- Invalid schemas
- Missing identifiers
- Unknown shipment references
- Temporary downstream failures
- Data-quality issues
- Unsupported schema versions
- External service timeouts

Rather than discarding failed messages, the platform can route them through controlled error-handling paths.

A generalized model is:

**Primary Stream → Processing Service → Success Stream**

and

**Primary Stream → Processing Failure → Retry → Dead-Letter Stream**

The dead-letter stream provides a controlled mechanism for operational investigation, correction, and replay.

#### ***L. Replay and Recovery***

Event retention provides an additional resilience mechanism. When processing logic changes or a downstream service fails, historical events can potentially be replayed to reconstruct state.

For example:

**Historical Event Stream → Replay Processor → Shipment State Reconstruction**

This capability is particularly valuable during disaster recovery, application migration, debugging, and model validation. However, replay operations should be isolated from live processing where necessary to prevent historical events from unintentionally triggering current operational notifications.

#### ***M. Data Freshness and Latency Monitoring***

Real-time visibility requires explicit measurement of data freshness.

Important metrics include:

##### **Ingestion Latency**

Ingestion Time – Event Time

##### **Processing Latency**

Processing Completion Time – Ingestion Time

##### **End-to-End Visibility Latency**

Visibility Update Time – Event Time

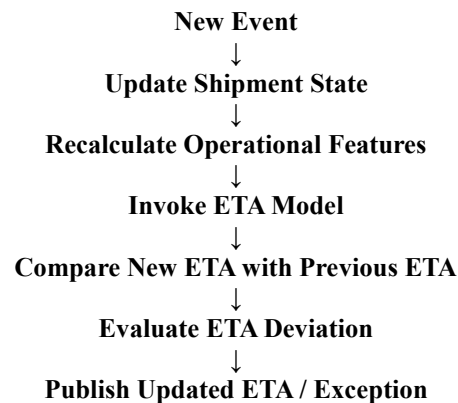
These measurements help identify whether delays originate from source systems, network transmission, streaming infrastructure, processing services, or downstream data stores.

Data freshness can also be exposed directly to users. A shipment dashboard displaying a location update from several hours earlier should clearly distinguish stale information from current information.

### *N. Event Processing and ETA Intelligence*

The event-processing layer provides the foundation for continuous ETA prediction.

A typical cycle is:



For example, a shipment may initially have an ETA of **14:30**. A significant facility delay event may change the prediction to **16:10**. The ETA service can publish the updated prediction and allow the exception engine to determine whether the change crosses a business-defined threshold.

### *O. Reference Processing Workflow*

The complete real-time data-engineering workflow can therefore be summarized as:

#### **Source Event Generation**

- Secure Ingestion
- Schema Validation
- Canonicalization
- Event Publication
- Stream Processing
- Event Correlation
- Context Enrichment
- Shipment State Update
- ETA Recalculation
- Exception Evaluation
- Operational Publication
- Dashboard / API / Notification

This continuous pipeline forms the technical backbone of the visibility platform. Reliable event ingestion and stream processing ensure that the ETA intelligence layer receives current, contextualized, and trustworthy information rather than isolated or stale shipment updates.

### *P. Engineering Considerations*

Several engineering principles should guide implementation:

1. **Design for duplicate delivery rather than assuming exactly-once behavior everywhere.**
2. **Preserve event timestamps and source metadata for traceability.**
3. **Use schema versioning to support independent system evolution.**
4. **Separate raw events from derived business events.**
5. **Support replay for recovery and analytical purposes.**
6. **Measure end-to-end latency rather than only application response time.**
7. **Design for event bursts and seasonal traffic patterns.**
8. **Treat invalid and delayed events as manageable operational conditions.**
9. **Maintain clear lineage between source events and derived shipment state.**
10. **Keep real-time processing sufficiently decoupled from downstream user interfaces.**

With this event-processing foundation established, the next architectural challenge is to transform continuously arriving events into **intelligent and dynamically updated ETA predictions**.

## V. ETA INTELLIGENCE AND PREDICTIVE ARRIVAL-TIME ARCHITECTURE

Estimated time of arrival is one of the most operationally significant outputs of a supply chain visibility platform. A conventional ETA is often derived from planned departure time, expected travel duration, and scheduled delivery information. Although this approach provides a useful baseline, it does not adequately represent the constantly changing conditions encountered during transportation and fulfillment operations. Modern ETA intelligence therefore treats arrival time as a dynamic prediction that can be continuously recalculated as new information becomes available.

Research on ETA prediction has demonstrated the applicability of machine learning to transportation environments. For example, machine-learning approaches have been investigated for intermodal transportation networks using historical transportation and external data, with the objective of supporting proactive responses to potential delays. Deep learning has also been applied to package delivery ETA prediction using spatial and temporal characteristics of delivery routes. These approaches provide useful foundations for designing enterprise-scale ETA services, although production implementations must additionally address data quality, integration latency, explainability, model governance, and operational reliability.

### A. ETA as a Continuous Prediction

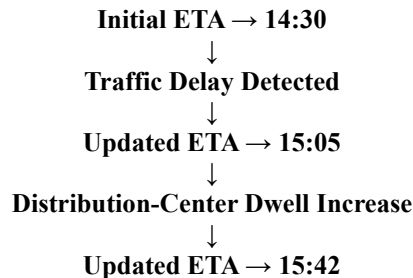
A generalized ETA calculation can be expressed as:

$$\text{ETA}(t) = \text{Current Time} + \text{Predicted Remaining Travel Time} + \text{Predicted Operational Processing Time}$$

The remaining travel time can depend on current location, destination, route, traffic, vehicle characteristics, historical route behavior, and environmental conditions. Operational processing time may include expected loading, unloading, appointment, gate, dock, or distribution-center processing.

Unlike a static calculation, the prediction is recalculated whenever material information changes.

For example:



This continuous recalculation enables the platform to provide a more current representation of expected shipment arrival.

### B. ETA Prediction Inputs

The quality of an ETA model depends substantially on the quality and relevance of its input features.

Potential features include:

| Feature Category | Example Inputs                                |
|------------------|-----------------------------------------------|
| Shipment         | Weight, priority, shipment type               |
| Route            | Distance, road characteristics, planned route |
| Location         | Current latitude/longitude, facility location |
| Transportation   | Vehicle type, carrier, transportation mode    |
| Time             | Departure time, day of week, season           |
| Facility         | Historical loading and unloading duration     |
| Carrier          | Historical transit and service performance    |
| Traffic          | Congestion and road conditions                |
| Weather          | Temperature, precipitation, severe weather    |

| Feature Category | Example Inputs                                 |
|------------------|------------------------------------------------|
| Operational      | Dwell time, appointment delay, route deviation |
| Historical       | Similar routes, previous shipment durations    |
| Network          | Regional congestion and facility workload      |

The platform should distinguish between features that are available at prediction time and information that becomes available only after delivery. Using future information during model training can introduce data leakage and produce misleading model-performance measurements.

### C. Feature Engineering

Raw operational events are often not directly suitable for machine learning.

For example, a GPS stream may provide:

#### Latitude, Longitude, Timestamp

The feature-engineering layer can transform these values into:

- Distance remaining
- Average recent speed
- Time since last movement
- Route deviation distance
- Estimated distance to next facility
- Historical travel duration for the route
- Current dwell duration
- Time spent in the current operational state

Similarly, event sequences can be transformed into features such as:

#### Time Since Departure

#### Time Since Last Milestone

#### Actual Progress / Expected Progress

#### Facility Dwell / Historical Facility Dwell

These derived features can provide greater predictive value than individual raw events.

### D. Hybrid ETA Architecture

A production-grade ETA platform does not necessarily need to rely exclusively on machine learning. A hybrid architecture can combine deterministic transportation calculations, business rules, statistical methods, and machine learning.

A generalized approach is:

#### Baseline ETA

•

#### Operational Adjustment

•

#### Machine-Learning Prediction

•

#### Business Constraints

=

#### Final ETA

The baseline provides a stable reference, while machine learning captures nonlinear patterns and contextual effects.

Business constraints can prevent predictions from violating known operational conditions. For example, if a destination facility has a defined operating window, the final ETA may need to account for the next available receiving period.

### E. Model Selection

Different machine-learning approaches may be appropriate for different ETA problems.

Potential approaches include:

- Linear and generalized regression
- Gradient-boosted decision trees
- Random forests
- Neural networks

- Recurrent neural networks
- Spatial-temporal models
- Survival or hazard models
- Ensemble models

The choice should be driven by data characteristics, prediction latency, interpretability requirements, model-maintenance capability, and required accuracy.

Research has demonstrated that different model families can be useful for different transportation prediction tasks. For example, studies of delivery-stop duration have evaluated regression, classification, and hazard-duration approaches using GPS-derived information.

#### ***F. Prediction Confidence***

A single ETA timestamp does not fully communicate prediction uncertainty.

A more informative representation can include:

**Predicted ETA: 15:42**

**Expected Range: 15:30–15:58**

**Confidence: High**

The confidence mechanism can be derived from model uncertainty, historical prediction errors, data completeness, and the similarity between the current shipment and historical observations.

When input data is incomplete—for example, when the latest location update is significantly stale—the platform should reduce confidence rather than presenting an apparently precise prediction.

#### ***G. Continuous ETA Recalculation***

ETA should be recalculated when a material event changes the shipment context.

Potential recalculation triggers include:

- Shipment departure
- Significant GPS movement
- Route change
- Traffic condition change
- Weather event
- Facility arrival
- Excessive facility dwell
- Missed appointment
- Carrier status update
- Delivery-window change
- Significant deviation from expected progress

A configurable recalculation policy can prevent excessive model invocation when high-frequency telemetry is available. For example, GPS events may arrive every few seconds, while an ETA update may only be required when the predicted arrival changes beyond a defined threshold.

#### ***H. ETA Deviation Analysis***

The platform should compare multiple temporal representations:

**Planned ETA**

**Previous Predicted ETA**

**Current Predicted ETA**

**Actual Arrival**

The difference between these values can be used to identify operational deviation.

For example:

**ETA Change = Current Predicted ETA – Previous Predicted ETA**

**Plan Deviation = Current Predicted ETA – Planned ETA**

These metrics provide useful signals for exception management and model-quality monitoring.

#### ***I. Route and Facility Intelligence***

Transportation time is not exclusively determined by road distance.

For retail distribution networks, facility processing can represent a significant component of end-to-end delivery time. A shipment may arrive at a distribution center on schedule but still experience a delay because of:

- Dock congestion

- Yard congestion
- Labor constraints
- Appointment delays
- Loading or unloading variability
- Inventory availability
- Operational interruptions

Consequently, ETA models should consider both **transportation time** and **node-processing time**.

Research on intermodal ETA prediction has similarly highlighted the importance of logistics-node processing times in addition to transportation times.

#### ***J. Network-Level ETA Intelligence***

Individual shipment predictions can also be aggregated to provide network-level intelligence.

For example, a control tower can calculate:

- Shipments at risk of late arrival
- Average ETA deviation by carrier
- Average facility dwell time
- Route-level delay patterns
- Regional transportation disruption
- Delivery-window risk
- Concentration of exceptions

This transforms ETA from a shipment-level field into a network intelligence capability.

#### ***K. Prediction Feedback Loop***

Actual delivery outcomes should be captured and compared against previous predictions.

The feedback process can be represented as:

##### **Prediction**

→ **Shipment Execution**

→ **Actual Arrival**

→ **Prediction Error Calculation**

→ **Model Evaluation**

→ **Feature / Model Improvement**

The prediction error can be calculated using metrics such as:

$$\text{MAE} = (1/n) \sum |\text{Actual ETA} - \text{Predicted ETA}|$$

Additional metrics can include median absolute error, percentile error, bias, and error distribution by route, carrier, facility, and shipment category.

This enables organizations to determine where the ETA model performs reliably and where additional data or specialized modeling may be required.

#### ***L. Model Drift and Operational Monitoring***

Transportation patterns can change over time. New routes, carrier changes, seasonal traffic patterns, facility changes, road construction, and operational policy changes can alter the relationship between model features and actual arrival time.

The platform should therefore monitor:

- Prediction error
- Feature distribution changes
- Data freshness
- Missing-feature rates
- Model invocation latency
- Prediction distribution
- Error by carrier
- Error by route
- Error by facility

A model that performs well during one operating period may require retraining or recalibration when the underlying transportation environment changes.

### **M. Digital-Twin Integration**

A mature visibility platform can eventually evolve toward a digital representation of the physical supply chain.

A digital twin can combine operational events, asset information, logistics processes, and analytical models to represent the current and potential future state of the network. Research has identified digital twins as a potential mechanism for improving logistics visibility and enabling predictive metrics and projections.

Within an ETA platform, a digital-twin layer could represent:

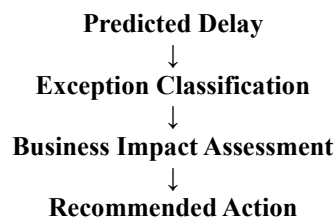
**Shipment → Vehicle → Route → Facility → Delivery Network**

The platform could then simulate the potential effect of a disruption before recommending an operational response.

### **N. Operational Decision Support**

ETA intelligence becomes valuable when predictions lead to appropriate operational decisions.

For example:



Possible actions may include:

- Reassigning transportation capacity
- Adjusting delivery appointments
- Prioritizing affected shipments
- Notifying downstream facilities
- Updating customer communication
- Initiating alternate transportation
- Escalating critical exceptions

The system should distinguish prediction from decision. The ETA model estimates a future condition, while business rules, operational users, or optimization services determine the appropriate response.

### **O. Reference ETA Intelligence Workflow**

The complete ETA intelligence workflow can therefore be represented as:

#### **Operational Event**

- Shipment-State Update
- Feature Generation
- Context Enrichment
- ETA Model
- Confidence Estimation
- ETA Comparison
- Exception Evaluation
- Operational Decision
- Actual Outcome
- Model Feedback

This architecture creates a closed-loop intelligence capability in which predictions continuously improve as the platform accumulates additional operational outcomes.

### **P. Engineering Considerations**

A production ETA intelligence platform should follow several principles:

1. Treat ETA as a continuously evolving prediction rather than a static field.
2. Separate planned, carrier-provided, and platform-generated ETA values.
3. Use only information available at prediction time when generating model features.
4. Preserve prediction history for audit and model evaluation.
5. Combine deterministic constraints with predictive models where appropriate.
6. Represent uncertainty instead of presenting false precision.
7. Monitor prediction quality across routes, carriers, facilities, and operating periods.

8. Design the prediction service independently from the user interface.
9. Capture actual outcomes for continuous model evaluation.
10. Provide operational explanations and supporting context for significant ETA changes.

The combination of real-time event processing and predictive ETA intelligence establishes the foundation for proactive supply chain management. However, prediction alone does not resolve operational disruption. The next stage is to connect ETA intelligence with **exception management, alert prioritization, and automated operational workflows**, allowing the platform to identify significant risks and route them to the appropriate response mechanisms.

## VI. CONCLUSION

Real-time supply chain visibility has become an important architectural capability for large-scale retail distribution networks because transportation and fulfillment operations increasingly depend on rapidly changing information. Traditional batch-oriented tracking approaches can provide historical and milestone-based information, but they are less suited to environments where shipment conditions, facility operations, transportation routes, and external factors change continuously.

This paper presented a generalized architecture for engineering real-time supply chain visibility and ETA intelligence platforms. The proposed architecture combines heterogeneous data ingestion, canonical event models, event streaming, stream processing, shipment-state management, contextual enrichment, predictive ETA services, exception processing, operational APIs, analytics, security, and observability. These components collectively establish a foundation through which operational events can be transformed into continuously updated supply chain intelligence.

A central principle of the proposed architecture is the separation between raw operational events and current shipment state. Historical events provide the foundation for auditing, analytics, replay, and machine learning, while the current-state layer enables low-latency access to operational information. This separation also supports progressive modernization because existing transportation, warehouse, order, and carrier systems can continue operating while modern event-driven services are introduced around them.

ETA intelligence extends visibility by converting operational information into forward-looking predictions. Instead of treating arrival time as a fixed transportation-plan attribute, the platform can continuously recalculate ETA using current shipment state, historical transportation behavior, facility processing characteristics, route information, and contextual signals. Machine learning can enhance this capability, while deterministic rules and operational constraints provide additional controls for production environments.

The reliability of the overall platform depends on more than predictive-model performance. Data quality, event ordering, duplicate handling, late-arriving events, schema evolution, replay, observability, security, and model governance are equally important. A highly accurate model cannot provide reliable operational value if its input data is stale, incomplete, incorrectly correlated, or disconnected from the actual shipment lifecycle.

The architecture also provides a path toward increasingly intelligent supply chain operations. Once reliable event processing and ETA prediction capabilities are established, organizations can introduce more advanced capabilities such as network-level risk analysis, digital twins, predictive exception management, capacity optimization, and automated decision-support workflows. Research has similarly identified digitalization, end-to-end visibility, analytics, and digital-twin technologies as important mechanisms for improving supply chain transparency and resilience.

Ultimately, real-time supply chain visibility should not be viewed as merely a tracking application. It represents an architectural capability that connects physical transportation and fulfillment activities with digital event processing and predictive intelligence. By continuously converting operational events into trusted state, predictions, and actionable exceptions, organizations can establish a more responsive foundation for managing complex retail distribution networks while retaining interoperability with existing enterprise systems.

## REFERENCES

- [1] R. Srinivasan and M. Swink, "An Investigation of Visibility and Flexibility as Complements to Supply Chain Analytics: An Organizational Information Processing Theory Perspective," *Production and Operations Management*, vol. 27, no. 10, 2018, doi: 10.1111/poms.12746.
- [2] S. Zhu, J. Song, B. T. Hazen, K. Lee, and C. Cegielski, "How Supply Chain Analytics Enables Operational Supply Chain Transparency," *International Journal of Physical Distribution & Logistics Management*, vol. 48, no. 1, pp. 47–68, 2018, doi: 10.1108/IJPDLM-11-2017-0341.
- [3] A. Roy, S. M. Gilbert, and G. Lai, "The Implications of Visibility on the Use of Strategic Inventory in a Supply Chain," *Management Science*, vol. 65, no. 4, pp. 1752–1767, 2019, doi: 10.1287/mnsc.2018.3033.
- [4] F. Wu and L. Wu, "DeepETA: A Spatial-Temporal Sequential Neural Network Model for Estimating Time of Arrival in Package Delivery System," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, no. 1, pp. 774–781, 2019, doi: 10.1609/aaai.v33i01.3301774.
- [5] S. Hughes, S. Moreno, W. F. Yushimito, and G. Huerta-Cánepa, "Evaluation of Machine Learning Methodologies to Predict Stop Delivery Times from GPS Data," *Transportation Research Part C: Emerging Technologies*, vol. 109, pp. 289–304, 2019, doi: 10.1016/j.trc.2019.10.018.
- [6] A. Balster, O. Hansen, H. Friedrich, and A. Ludwig, "An ETA Prediction Model for Intermodal Transport Networks Based on Machine Learning," *Business & Information Systems Engineering*, vol. 62, pp. 403–416, 2020, doi: 10.1007/s12599-020-00653-0.
- [7] C. Kritzing et al., "Digital Twin-driven Supply Chain Planning," *Procedia CIRP*, vol. 93, pp. 198–203, 2020, doi: 10.1016/j.procir.2020.04.154.
- [8] A. R. W. M. A. Moshood et al., "The Role of Digital Twins in the Fulfilment Logistics Chain," *IFAC-PapersOnLine*, vol. 53, no. 2, pp. 10574–10578, 2020, doi: 10.1016/j.ifacol.2020.12.2807.
- [9] D. Zouari, S. Ruel, and L. Viale, "Does Digitalising the Supply Chain Contribute to Its Resilience?" *International Journal of Physical Distribution & Logistics Management*, vol. 51, no. 2, pp. 149–180, 2021, doi: 10.1108/IJPDLM-01-2020-0038.
- [10] D. Ivanov, "Digital Supply Chain Management and Technology to Enhance Resilience by Building and Using End-to-End Visibility During the COVID-19 Pandemic," *IEEE Transactions on Engineering Management*, vol. 71, pp. 10485–10495, 2021, doi: 10.1109/TEM.2021.3095193.
- [11] T. D. Moshood, G. Nawanir, S. Sorooshian, and O. Okfalisa, "Digital Twins Driven Supply Chain Visibility within Logistics: A New Paradigm for Future Logistics," *Applied System Innovation*, vol. 4, no. 2, Art. no. 29, 2021, doi: 10.3390/asi4020029.
- [12] S. H. M. H. Ben-Daya et al., "Contrasting Supply Chain Traceability and Supply Chain Visibility: Are They Interchangeable?" *The International Journal of Logistics Management*, vol. 32, no. 3, pp. 942–972, 2021, doi: 10.1108/IJLM-05-2020-0214.
- [13] S. Li and B. Huo, "Information Sharing, Coordination and Supply Chain Performance: The Moderating Effect of Demand Uncertainty," *Industrial Management & Data Systems*, vol. 119, no. 5, pp. 1046–1071, 2019, doi: 10.1108/IMDS-10-2018-0453.



INTERNATIONAL  
STANDARD  
SERIAL  
NUMBER  
INDIA



# INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN SCIENCE | ENGINEERING | TECHNOLOGY

 9940 572 462  6381 907 438  [ijirset@gmail.com](mailto:ijirset@gmail.com)



[www.ijirset.com](http://www.ijirset.com)

Scan to save the contact details