

**AN INTEROPERABLE FREIGHT EVENT MODEL FOR CROSS-PLATFORM DRIVER, TRIP, TELEMATICS, DOCUMENT, AND PAYROLL SYNCHRONIZATION****Prasanna Sundaravaradan**

Lead Developer, Anveta Inc

**ABSTRACT**

Operational information is exchanged on commercial transportation platforms through mobile application, transportation management systems, electronic logging devices, telematics providers, document platforms, customer portals and payroll systems. Application programming interfaces (APIs) can provide technical connectivity, but differences in semantic representation between platforms remain a challenge for organizations as they represent drivers, trips, locations, duty status, documents and operational exceptions differently. This paper presents a technology neutral semantic model for exchanging operational events between heterogeneous fleet systems, called Interoperable Freight Event Model. The model specifies common envelopes for events, business identifiers, temporal semantics, rules for versioning, source provenance, correction history, sequencing requirements, privacy classifications, and expectations on processing. There are seven core event families defined: Driver Action, Trip State Changed, Duty Status Recorded, Document Captured, Delivery Confirmed, Operational Exception Raised and Compensation Adjustment Proposed. This framework contains mappings for REST application programming interfaces, message queues, publish-subscribe brokers, and event-streaming platforms, and a conformance-testing approach to validate producers and consumers. Consider that ten structural elements (e.g., participants) participate in a single envelope architecture, and that the number of mappings required is ten for that architecture, but is up to forty-five for an alternative, point-to-point mapping of the same ten participants. The seven event families are categorized into three tiers of handling privacy, depending on their relative level of sensitivity, criticality in real time, frequency of correction, and regulatory significance. The proposed model aims to lower the lock-in situation with the vendors and offer a shared platform for interoperable digital freight operations.

**Keywords:** semantic interoperability, event-driven architecture, freight management, electronic logging devices, telematics, publish-subscribe messaging, schema evolution, data provenance, conformance testing, payroll synchronization, transportation management systems, event envelope.

**1. INTRODUCTION**

Commercial trucking and freight-brokerage relies on a diverse array of software platforms, with very few having a common data vocabulary. It runs through a driver-facing mobile app, a Transportation Management System (TMS) for dispatch, an electronic logging device (ELD) or telematics service provider that captures vehicle and duty status information, a document-capture platform to store proof-of-delivery images and bills of lading, a customer-facing visibility portal, and a payroll system that converts trip and duty data into driver paychecks [1]. Each platform tends to be provided by a different vendor, to have a different release cycle, and to have different field names, identifiers, and conventions for drivers, trips, locations and duty status. Technical connectivity between these platforms is provided by Application Programming Interfaces (APIs), but despite their availability, not all the meaning of data exchanged is preserved, a problem that is often referred to as semantic interoperability, and is still a major challenge in digital freight operations.

Semantic incompatibility has practical and cost implications. Dispatchers manually rebalance trip statuses between systems, payroll administrators manually reconcile driver pay to telematics logs and trip sheets, and compliance officers have a challenge in maintaining a single audit trail across the records from multiple vendors. Every additional point-to-point integration adds to the number of custom mappings that need to be created and maintained, and as platforms move forward on their own, any mapping becomes brittle and highly fragile, creating a patchwork of disjointed and wobbly digital ecosystems.

This paper introduces an Interoperable Freight Event Model (IFEM), an operational event semantic model that is technology neutral and designed for representing and exchanging operational events between heterogeneous fleet-management, telematics, documents and payroll systems [2]. The model doesn't impose a single schema that all vendors must implement, but provides a shared schema for the event envelope, a shared vocabulary of business identifiers, explicit temporal semantics, a versioning discipline, source-provenance metadata, and correction-handling rules that build on existing platform-specific schemas, in addition to seven core event families, each representing a recurring class of occurrence that is consumed by multiple downstream systems (Driver Action, Trip State Changed, Duty Status Recorded, Document Captured, Delivery Confirmed, Operational Exception Raised, and Compensation Adjustment Proposed).

## BACKGROUND AND RELATED WORK

The data exchange for freight industry has developed in several different ways of integration, each affected by the technological limitations of the time, as shown in Fig. 1. One of the early standardized methods for electronic exchange of structured business documents like purchase orders and shipment notices came with the development of Electronic Data Interchange (EDI), which was the first to be developed, where transaction sets were used in fixed formats and sent in scheduled batches. According to the research by EDI, their success in achieving the standardisation of the document structure across industries was to be expected, but their batch format is not well aligned with the near-real-time visibility that the freight customers are looking for [3]. The temporal coupling that developed with point-to-point integration using SOAP/XML and then with REST APIs resulted from the fact that the former were typically used to poll or request data on the consumer's own schedule, and the latter, in particular, did not lend itself to broadcasting a change of state to many interested parties without a custom notification mechanism per consumer.

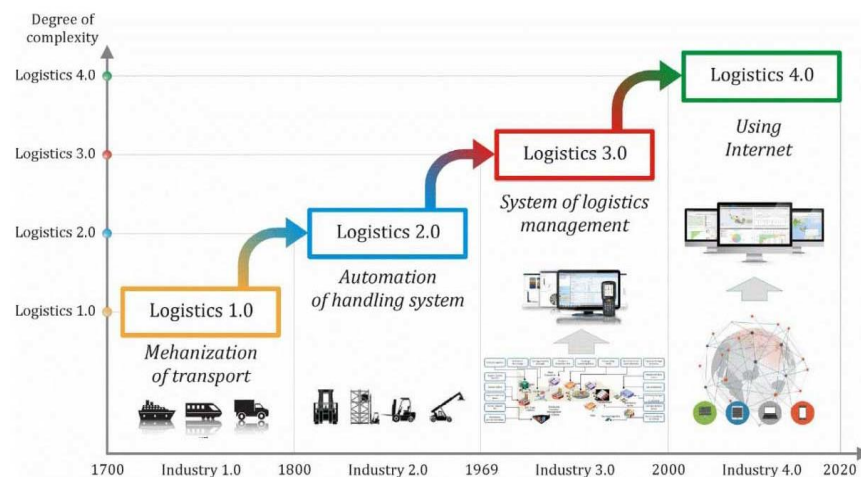


**Fig. 1.** Evolution of freight-industry integration paradigms. (IUJ Journal of Management,2020)

The trend to microservices further broke monolithic applications into smaller, stand-alone services to further enhance agility, but also added to the number of points at which data needed to be translated. Publish-subscribe messaging – in which producers push messages onto topics that they don't know about – and complex event processing (CEP) – which continually works through event streams to look for the higher-level patterns – combine to form the architectural terms of decoupled, event-driven integration. When a 'raw' geofence crossing telemetry reading is correlated with a window of time for an appointment, the resulting 'arrived late' exception has been said by CEP research to be a pipeline event processing pattern that filters, aggregates and correlates primitive events to create derived events.

Semantic-interoperability research in the context of the Internet of Things (IoT) domain has focused on syntactic interoperability (where systems are able to read a shared message format) and semantic interoperability (where systems also agree on the meaning of the fields). It has been argued that the approach to achieving interoperability in the existing installed base is to implement translation and mediation layers instead of universal schema replacement, which is the basis for the layered-envelope approach described in Section III, which provides a common envelope while allowing the platform-specific schema of the payload fields to remain under the envelope [4].

A global-as-view approach, where each source is mapped into the common schema, and a local-as-view approach, where the common schema is defined in terms of the sources, are two extremes for a multi-vendor freight ecosystem where no single entity governs the whole. Global-as-view is a practical middle-ground between rigid schema standardization and schema chaos, when there is no single entity that controls the whole freight ecosystem.

## **PROPOSED INTEROPERABLE FREIGHT EVENT MODEL**

### **A. Design Principles**

The proposed model is based on five principles distilled from the background literature: envelope/payload separation (a small, stable outer structure which is separate from rapidly changing payload content, in line with mediation-based interoperability); explicit temporal semantics (explicit indication of when an event occurred, and when it was recorded, following mediation-based interoperability); batch limitation, addressed by the principle of explicit temporal semantics, in which a correcting event references the record being corrected, rather than overwriting it, to preserve an auditable history of the event; and protocol neutrality, which keeps the envelope and event vocabulary separated from any single transport mechanism, permitting both REST bindings, message-queue bindings, publish-subscribe, and streaming bindings to coexist, as discussed in Section IV.

### **B. Event Envelope Structure**

Under the proposed model, all events that are exchanged are wrapped in a common envelope as shown in Fig. 2. The header layer contains the type of event, model version and schema identifier that is used for payload validation. The business-identifier layer contains persistent identifiers such as driver identifier, trip identifier, vehicle identifier and load identifier, which enable events from various source systems to be correlated. The temporal-semantics layer is responsible for separating delivery time from event time and also comprises a sequence token based on the logical-clock notion of Section II, which consumers use to guarantee reliable ordering even in the face of delivery delays or reordering. The source-provenance layer identifies the system the data came from, how it was captured (manually-entered, telematics, or OCR), and the name of the person or system that triggered the data capture [5]. The correction-history layer stores the event superseding information as well as a correction reason and a revision number, and the payload layer contains the business content of the event that is validated independently of a schema associated with the declared event type.

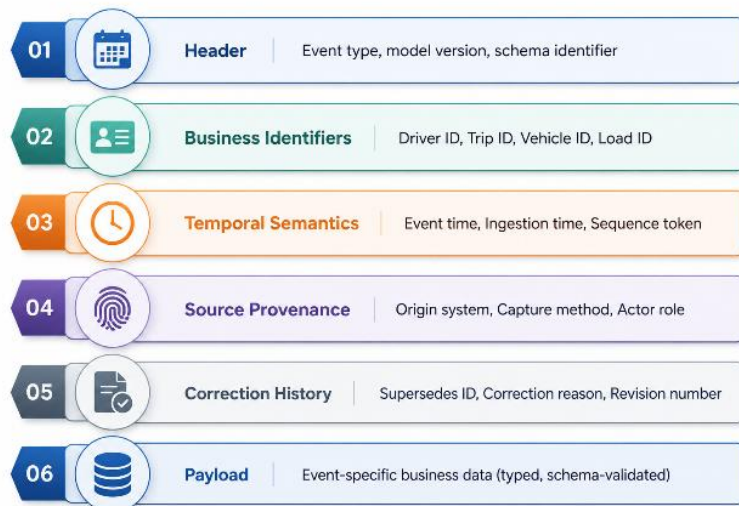


**Fig. 2.** Layered structure of the proposed event envelope.

### C. Core Event Families

Each of the events in the families listed in Table I represents a type of occurrence that typically arises from multiple different sources but is consumed by several. Table I summarizes seven core event families, each a type of occurrence that can originate from multiple sources but be consumed by multiple sources, including Driver Action events, Trip State Changed events, Duty Status Recorded events, Document Captured events, Delivery Confirmed events, Operational Exception Raised events and Compensation Adjustment Proposed events [6].

**Table I.** Core Event Families and Primary Source Systems

| Event Family                   | Primary Source System        | Typical Consumers           |
|--------------------------------|------------------------------|-----------------------------|
| DriverAction                   | Driver mobile application    | Dispatch, exception service |
| TripStateChanged               | Trip / TMS platform          | Customer portal, analytics  |
| DutyStatusRecorded             | Telematics / ELD provider    | Compliance & HOS, payroll   |
| DocumentCaptured               | Document capture system      | Billing, customer portal    |
| DeliveryConfirmed              | Mobile app / document system | Customer portal, billing    |
| OperationalExceptionRaised     | TMS / telematics provider    | Exception service, dispatch |
| CompensationAdjustmentProposed | Trip and duty aggregation    | Payroll calculation engine  |

### D. Ordering and Idempotent Processing

Because the same occurrence may be reported by more than one producer, or retransmitted after a network interruption, consumers must recognize duplicate events without discarding legitimate corrections. Equation (1) expresses the happens-before ordering principle adapted from logical-clock theory, in which the sequence token  $S$  assigned to event  $e$  pairs a monotonically increasing logical counter  $C$  with a producer-node identifier  $N$ , used as a tie-breaker when two events share a counter value:

$$S(e) = (C(e), N(e)), \quad a \rightarrow b \Rightarrow C(a) < C(b) \quad (1)$$

Equation (2) expresses the idempotent-consumption check applied by a consumer, in which an event is processed only if its identifier has not already been recorded in a local deduplication set  $D$ :

$$\text{Process}(e) = 1 \text{ if } H(ID(e)) \notin D, \quad \text{else } \text{Process}(e) = 0(2)$$

where  $H$  denotes a deterministic hash of the event identifier. This check allows a consumer to safely reprocess a delivery stream after an outage without producing duplicate payroll adjustments or duplicate compliance records.

## SYSTEM ARCHITECTURE AND PROTOCOL BINDINGS

### E. Reference Architecture

Fig. 3 shows a reference architecture of collecting the five types of source-systems and mapping their native data into events that are envelope compatible [7]. They go through a central event-model layer, which includes envelope-validation and schema-registry functions, before they are sent to the systems that consume them, such as compliance and hours-of-service monitoring, payroll-calculation engine, customer-visibility portal, analytics/reporting and exception-and-alerting service functions. This hub-oriented arrangement decreases the number of mappings that need to be maintained, as each participant only needs to map to and from the shared envelope and not individual mappings with each of the other participants.

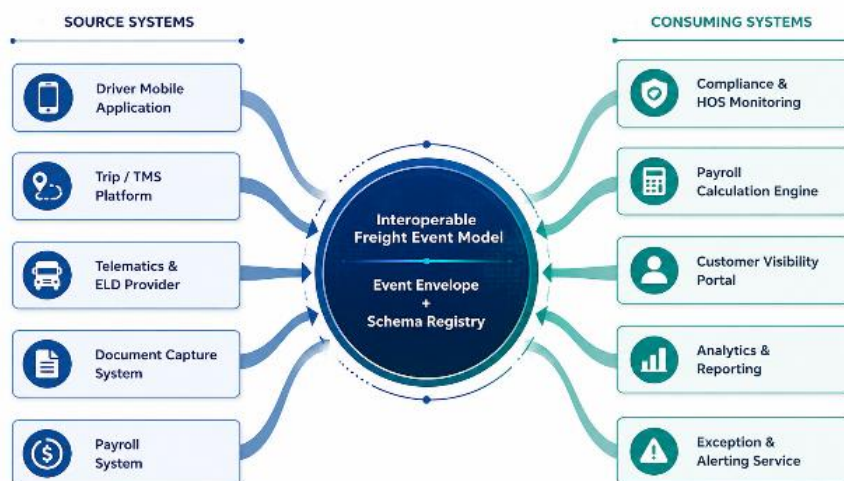


**Fig. 3.** Reference architecture for cross-platform event exchange.

### F. Comparative Analysis of Integration Paradigms

There are four protocol paradigms that are typically used for moving envelope-conformant events: REST APIs, message queues, publish-subscribe, and event-streaming [8]. To compare these paradigms, the five qualitative dimensions presented in this section—namely: loose coupling, ordering guarantee, horizontal scalability, delivery reliability, and real-time suitability—have been rated on a relative one-to-five scale and presented in the form of a comparison in Fig. 4.

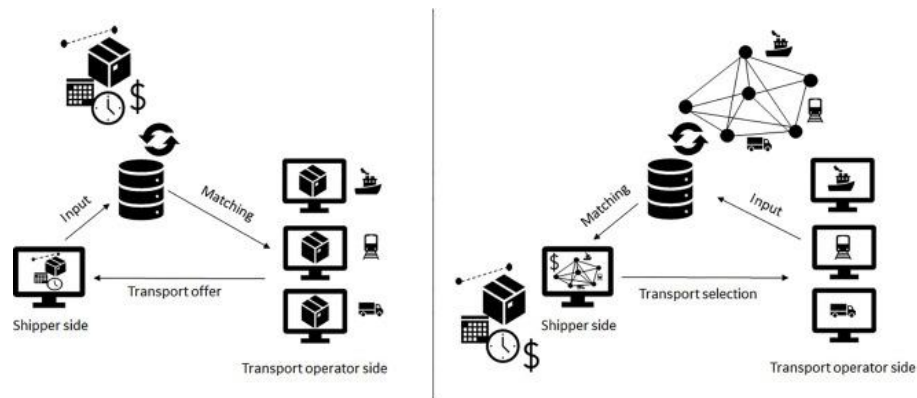


**Fig. 4.** Freight Mobility as a Service: Open platforms for synchromodal transport (ScienceDirect , 2021)

While REST APIs are simple and widely supported, they are not necessarily real-time suitable because they require the consumer to poll the producer, which also has to be up and running. Message queues can help to ensure that the message is delivered reliably, and the producer and consumer are available, which works well in a pipeline that requires each adjustment to occur exactly once, as is the case in a payroll pipeline. Many independent consumers, like compliance and the customer portal, can subscribe to the same event as the producer without having to keep a list of its consumers, making it maximally decoupled [9]. Event-streaming platforms are an extension of publish-subscribe that provides the ability to record events in an ordered, replayable, and persistent manner that is well suited to sequence-token approach of Section III and consumers that need to reprocess events from the past, such as analytics applications.

This comparison is summarized in Table II, and a representative use case is given for each paradigm; Table III gives some binding guidance on which paradigm is most suitable for each of the seven core event families, depending on the paradigm's latency sensitivity and fan-out characteristics.

**Table II.** Comparative Characteristics of Integration Paradigms (Relative Scale, 1 = Low, 5 = High)

| Paradigm          | Coupling | Ordering | Scalability | Reliability | Real-Time | Representative Use Case                               |
|-------------------|----------|----------|-------------|-------------|-----------|-------------------------------------------------------|
| REST APIs         | 2        | 3        | 3           | 3           | 2         | On-demand trip-status lookup by a dispatcher console  |
| Message Queues    | 4        | 4        | 4           | 5           | 3         | Sequential payroll-adjustment processing              |
| Publish-Subscribe | 5        | 3        | 4           | 3           | 4         | Broadcasting trip-state changes to multiple consumers |
| Event Streaming   | 4        | 5        | 5           | 4           | 5         | Real-time telematics ingestion with historical replay |

**Table III.** Protocol Binding Guidance by Event Family

| Event Family     | Latency | Fan-Out | Recommended Binding |
|------------------|---------|---------|---------------------|
| DriverAction     | Medium  | Low     | Message Queue       |
| TripStateChanged | Medium  | High    | Publish-Subscribe   |

|                                |        |        |                   |
|--------------------------------|--------|--------|-------------------|
| DutyStatusRecorded             | High   | Medium | Event Streaming   |
| DocumentCaptured               | Low    | Low    | REST API          |
| DeliveryConfirmed              | Medium | High   | Publish–Subscribe |
| OperationalExceptionRaised     | High   | High   | Event Streaming   |
| CompensationAdjustmentProposed | Low    | Low    | Message Queue     |

## VERSIONING, CORRECTION HANDLING, AND CONFORMANCE TESTING

### G. Schema Evolution and Backward Compatibility

Because source and consuming systems in a multi-vendor ecosystem are upgraded independently, the model must tolerate schema evolution without simultaneous upgrades. An additive-change discipline is adopted: a new schema version is backward compatible with a prior version when the required-field set of the prior version remains present and semantically unchanged, expressed in Equation (3), where  $R$  denotes the required-field set of version  $v$  and  $Fields$  denotes the complete field set of version  $v'$ :

$$\text{Backward-Compatible}(v, v') = \text{TRUE if } R(v) \subseteq \text{Fields}(v')(3)$$

This discipline allows for the optional addition of new fields at any time, but it is a breaking change to remove or rename a required field, or to change its interpretation, as a new major version identifier is needed in the envelope header.

### H. Correction History and the Event Lifecycle

The lifecycle of an event instance described in Fig. 5 is the capture at a source system, construct an envelope, publish to an appropriate event channel for the selected paradigm, route and deliver to the event consumer, consume and acknowledge, and, if needed, correct or supersede [10].

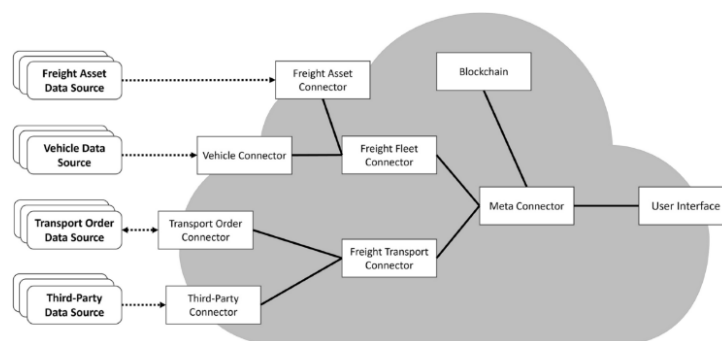


Fig. 5. Event lifecycle from capture through correction or supersession. (SpringerNature,2020)

Payroll and compliance data have legal and financial implications and each event is published as a snapshot which cannot be changed; a wrong duty status entry is never edited to reflect the error, but instead a new event is published to contain the superseding identifier and a structured event correction reason which may be used to reconstruct the current authoritative value and the entire history of event corrections [11][12].

### I. Conformance Testing Approach

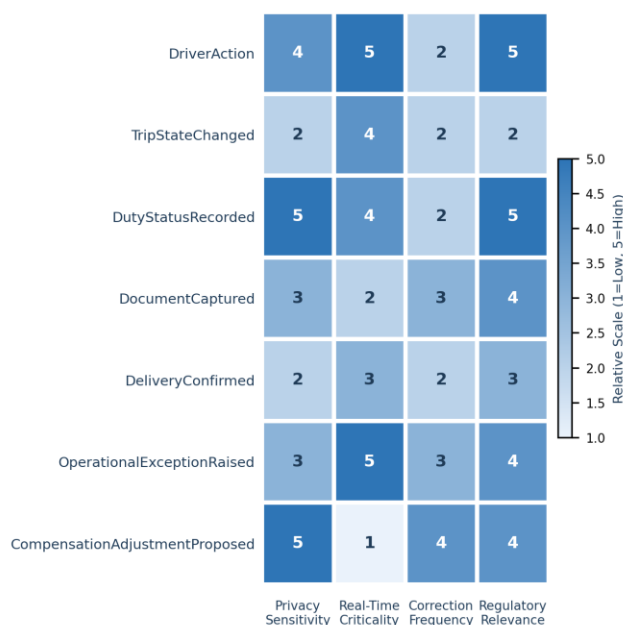
A conformance-testing approach ensures that a candidate producer/consumer implementation conforms to the model prior to deployment. Four kinds of tests are summarized in Table IV: structural tests confirm that produced events are valid with respect to the declared envelope and payload schema; semantic tests confirm that business-identifier fields reference entities in a consistent manner with respect to a reference test fixture; ordering tests confirm that events that are sent out of order are correctly ordered as they arrive; and idempotency tests confirm that events sent by a producer that are already received are correctly deduplicated as per Equation (2). If an implementation passes all four categories with respect to a shared reference suite, it can be said to be conformant and thus pairwise integration testing between each source and consuming system is not necessary [13].

**Table IV.** Conformance Test Categories

| Test Category | Objective                                                                                  | Related Equation |
|---------------|--------------------------------------------------------------------------------------------|------------------|
| Structural    | Validate produced events against the declared envelope and payload schema                  | —                |
| Semantic      | Verify business-identifier fields reference entities consistently with a reference fixture | —                |
| Ordering      | Verify correct sequence-token comparison under out-of-order delivery                       | Eq. (1)          |
| Idempotency   | Verify correct deduplication when a producer retransmits an event                          | Eq. (2)          |

### GOVERNANCE, PRIVACY, AND ADOPTION CONSIDERATIONS

Operational freight events come in all shapes and sensitivities that will contain information, how frequently the information needs to be corrected, and how important the information is to the regulatory audit. The seven key event families are described in Fig. 6 with respect to four dimensions which are relevant to governance: Privacy sensitivity, real-time criticality, correction frequency, and regulatory relevance, each relative on a one-to-five scale.

**Fig. 6.** Governance characterization of the seven core event families.

Duty Status Recorded and Compensation Adjustment Proposed events are highly rated for inclusion on the personally identifiable information list, since they are related to a driver's location history and compensation awarded, whereas Trip State Changed and Delivery Confirmed events are mostly related to shipment state, and rate comparatively lower.

The analysis is translated into three levels of privacy classification for storage, transmission and retention – a restricted level that includes encryption at rest and in transit and role-based access controls, for duty-status and compensation events; a controlled level that would include access logging and limits on retention, for document and driver-action events; and a general level that would apply to trip-state and delivery-confirmation events,

## *International Journal of Applied Engineering & Technology*

describing the status of the shipment, but not the actual content. A single governance policy can be applied consistently to consuming systems through access-classification rules included in the envelope's provenance layer.

**Table V.** Privacy Classification Levels and Handling Requirements

| Classification | Applicable Event Families                             | Handling Requirements                                        |
|----------------|-------------------------------------------------------|--------------------------------------------------------------|
| Restricted     | DutyStatusRecorded,<br>CompensationAdjustmentProposed | Encryption at rest and in transit; role-based access control |
| Controlled     | DocumentCaptured, DriverAction                        | Access logging; retention limits                             |
| General        | TripStateChanged, DeliveryConfirmed                   | Standard access controls                                     |

The main challenge to a common interoperability standard from an adoption perspective is a collective-action problem, where the cost of integration falls on the individual vendors, but the value comes to the whole ecosystem. The proposed model has been designed as an envelope whose content is added to the data models already in place within an individual vendor, rather than a replacement of the entire data model, thereby reducing the marginal cost of adoption for an individual vendor, as it is necessary to map only once the data model into the shared envelope.

### **EVALUATION APPROACH**

Due to the fact that this project is an "on paper" design and not a deployed system, there are four dimensions for evaluation. Schema-evolution evaluation focuses on finding out if representative changes (e.g. a new document type or exception category) can be made as additive changes (those satisfying Equation (3)) with no impact on existing consumers. As participants increase, the envelope approach becomes increasingly structurally efficient: As more and more participants are added to a five-source and five-consuming system as shown in Fig. 3, the number of mappings required becomes 10, whereas the number of mappings required for a fully point-to-point mesh is up to 45 per participant. Duplicate event handling evaluation is conducted to determine whether the idempotency check of Equation (2) works as intended under a variety of simulated retries and whether a complete record of corrections is available across simulated retries in a representative compensation adjustment scenario that includes an initial duty status record, a correction, and a final payroll calculation. Traceability evaluation is conducted to determine whether a complete record of corrections is available across simulated retries in a representative compensation adjustment scenario that includes an initial duty status record, a correction, and a final payroll calculation.

### **CONCLUSION AND FUTURE WORK**

This paper combined the concepts from EDI, event-driven architecture, CEP, semantic interoperability, distributed-systems ordering theory and data-provenance research to present an Interoperable Freight Event Model to exchange operational events between driver, trip, telematics, document and payroll systems. The model adds a layered event envelope that separates different types of metadata from the platform-specific payload content: header information, business-identifier, temporal, provenance, and correction-history metadata; seven event families representing common categories of freight occurrences; ordering and idempotency mechanisms based on the principles of logical-clock and deduplication; conformance testing approach that minimizes integration effort in pairwise scenarios; and an additive schema-evolution discipline that allows for independent vendor upgrade cycles. A comparison of the paradigms of REST, message-queue, publish-subscribe, and event-streaming suggests that none of these transports are ideal for all event families, which drives a protocol-neutral binding approach in the model.

The proposed model could be extended in a number of directions, such as: Formal machine-readable schema definitions for each core event family would enable the automated validation of tooling to be built directly on the model, while the four protocol bindings would be exercised in a multi-vendor test harness, possibly simulated, to allow for an evaluation of the mapping-effort and duplicate-handling in a real-world setting; The correction-

history mechanism could be extended to interoperate with other distributed-ledger approaches investigated in the context of blockchain-based supply chains to further strengthen cross-organizational trust in shared compensation and compliance records without a single centrally trusted intermediary; The proposed event vocabulary could be aligned with the broader physical-internet and logistics-interoperability research to position the model in a larger interoperable freight and supply-chain ecosystem beyond the scope addressed here.

## REFERENCES

- [1] F. Casino, T. K. Dasaklis, and C. Patsakis, "A systematic literature review of blockchain-based applications: Current status, classification and open issues," *Telematics and Informatics*, vol. 36, pp. 55–81, 2019.
- [2] G. Cugola and A. Margara, "Processing flows of information: From data stream to complex event processing," *ACM Computing Surveys*, vol. 44, no. 3, Art. no. 15, pp. 1–62, 2012.
- [3] N. Dragoni, S. Giallorenzo, A. L. Lafuente, M. Mazzara, F. Montesi, R. Mustafin, and L. Safina, "Microservices: Yesterday, today, and tomorrow," in *Present and Ulterior Software Engineering*, M. Mazzara and B. Meyer, Eds. Cham, Switzerland: Springer, 2017, pp. 195–216.
- [4] P. T. Eugster, P. A. Felber, R. Guerraoui, and A.-M. Kermarrec, "The many faces of publish/subscribe," *ACM Computing Surveys*, vol. 35, no. 2, pp. 114–131, 2003.
- [5] M. Ganzha, M. Paprzycki, W. Pawłowski, P. Szmeja, and K. Wasielewska, "Semantic interoperability in the Internet of Things: An overview from the INTER-IoT perspective," *Journal of Network and Computer Applications*, vol. 81, pp. 111–124, 2017.
- [6] L. Lamport, "Time, clocks, and the ordering of events in a distributed system," *Communications of the ACM*, vol. 21, no. 7, pp. 558–565, 1978.
- [7] M. Lenzerini, "Data integration: A theoretical perspective," in *Proc. 21st ACM SIGMOD-SIGACT-SIGART Symp. Principles of Database Systems*, 2002, pp. 233–246.
- [8] P. Mell and T. Grance, "The NIST definition of cloud computing," National Institute of Standards and Technology, Gaithersburg, MD, USA, NIST Special Publication 800-145, 2011.
- [9] S. Narayanan, A. S. Maruchek, and R. B. Handfield, "Electronic data interchange: Research review and future directions," *Decision Sciences*, vol. 40, no. 1, pp. 121–163, 2009.
- [10] National Academies of Sciences, Engineering, and Medicine, *Commercial Motor Vehicle Driver Fatigue, Long-Term Health, and Highway Safety: Research Needs*. Washington, DC, USA: National Academies Press, 2016.
- [11] S. Saberi, M. Kouhizadeh, J. Sarkis, and L. Shen, "Blockchain technology and its relationships to sustainable supply chain management," *International Journal of Production Research*, vol. 57, no. 7, pp. 2117–2135, 2019.
- [12] Y. L. Simmhan, B. Plale, and D. Gannon, "A survey of data provenance in e-science," *ACM SIGMOD Record*, vol. 34, no. 3, pp. 31–36, 2005.
- [13] C. S. Tang and L. P. Veelenturf, "The strategic role of logistics in the industry 4.0 era," *Transportation Research Part E: Logistics and Transportation Review*, vol. 129, pp. 1–11, 2019.
- [14] H. Treiblmaier, K. Mirkovski, P. B. Lowry, and Z. G. Zacharia, "The physical internet as a new supply chain paradigm: A systematic literature review and a comprehensive framework," *The International Journal of Logistics Management*, vol. 31, no. 2, pp. 239–287, 2020.