

Streaming Interoperable Construction Data with LDES

Carlos Ramonell-Cazador^{1,*}, Mathias Bonduel²

¹*Neanex Technologies*

Abstract

The construction industry faces persistent interoperability and efficiency bottlenecks when exchanging data across diverse IT systems during both project execution and final delivery to clients. This paper presents an ongoing effort to extend Fundamend, a commercial information management platform, with Linked Data Event Streams (LDES) to enable continuous, standards-based data flows between stakeholder IT solutions. The platform is complemented by a PoC LDES setup, using the open-source LDIO Workbench and the OpenLDES Server, complemented by a custom management interface. The approach is demonstrated through an experimental use case that streams the construction progress of a set of walls to a simulated client system. The experiment indicates that LDES can replace brittle, custom integrations with a uniform publication mechanism that technically decouples systems through standardised data models. It supports efficient, incremental synchronisation instead of full-dataset transfers, provides a predictable operational profile, and creates opportunities for upfront interface agreements among industry stakeholders.

Keywords

Linked Data Event Streams, Construction Information Management, Interoperability

1. Introduction

Construction projects generate information across a highly fragmented landscape of specialised domains, including Building Information Modelling (BIM), Geographical Information Systems (GIS), Systems Engineering (SE), Monitoring and Control (MC), and Electronic Document Management (EDM). These domains typically rely on their own distinct software systems, each with its own semantics, data models, and APIs. Furthermore, the underlying project data within these systems evolves asynchronously over time. Integrating this information throughout project execution and ultimately handing it over to the client's operational asset management environment remains an industry bottleneck [1].

Knowledge graphs and semantic technologies offer a robust paradigm to address this integration challenge [2]. A practical implementation of this approach is found in Fundamend¹, a commercial information management platform that leverages a centralised knowledge graph to ingest data from fragmented project systems. Within this platform, diverse vocabularies and ontologies from various stakeholders are imported and interconnected inside a Labelled Property Graph (LPG). Fundamend enables users to reclassify, validate, and enrich multi-source data into a unified, object-oriented representation, yielding key insights before delivering the data to downstream systems. However, data movement in and out of the platform remains reliant on custom, per-integration scripts. This ad-hoc approach requires bespoke change-set computation and tightly couples data consumers to the platform. Standard public APIs fail to resolve these issues. They merely shift the data-transformation burden to consumers, tightly couple them to specific API versions, and force inefficient polling patterns where clients must repeatedly request complete datasets to identify updates.

To overcome these integration issues, the construction sector is increasingly aligning with open data standards. In Europe, this shift is formalised by EN 17632 for Semantic Modelling and Linking (SML) [3], which addresses syntactic and semantic interoperability across the BIM, GIS, SE, MC, and EDM domains. SML achieves this by using RDF-based language bindings, specifically SKOS, RDFS,

SEMANTICS-2026: 22nd International Conference on Semantic Systems, September 15–17, 2026, Ghent

*Corresponding author.

[†]These authors contributed equally.

✉ carlos.ramonell@neanex.com (C. Ramonell-Cazador); mathias.bonduel@neanex.com (M. Bonduel)

ORCID 0000-0003-0390-3603 (C. Ramonell-Cazador); 0000-0002-3313-924X (M. Bonduel)



© 2026 Copyright for this paper by its authors. Use permitted under Creative Commons License Attribution 4.0 International (CC BY 4.0).

¹<https://fundamend.com/en>

OWL, and SHACL, to establish generic upper-ontology modelling patterns. However, the EN 17632 scope explicitly excludes statements on information transaction and data transportation mechanisms. Consequently, while SML provides the semantic framework to structure and validate data, it lacks a protocol for continuous, live data exchange. The challenge of streaming these standardised data changes dynamically between live systems remains unaddressed.

Linked Data Event Streams (LDES) [4] provide a standardised alternative ideally suited to this challenge. An LDES publishes immutable, versioned RDF objects as an append-only stream structured via the TREE hypermedia specification [5], allowing consumers to replicate the dataset once and maintain synchronisation by fetching only newly appended members. While successfully applied to base registries, sensor observations, and mobility data [6], its use as a change-propagation mechanism between construction systems has not yet been reported. Mapping the structure in the LPG to an OWL ontology and SHACL shapes, Fundamend's data can be dynamically exported into RDF to feed such a stream. LDES thus supplies the missing change-propagation contract, reducing the platform's public interface to static, cacheable HTTP fragments.

This paper presents (i) an architecture that extends Fundamend with LDES publication and consumption capabilities based on open-source components, and (ii) an experimental use case in which the on-site construction progress of a set of walls is streamed to a simulated third-party system.

2. System Architecture

The proposed architecture orchestrates an LDES-based data exchange ecosystem with Fundamend as intermediary (Figure 1). This design divides operations into an ingestion flow driven by *project streams* and a delivery flow driven by *delivery streams*.

Three core containerised services power this ecosystem. The first two are lightweight, modular components open-sourced by the openLDES initiative² for the development of the Flanders Smart Data Space.³ Specifically, the **LDIO Workbench**⁴ enables the construction of configurable pipelines that execute customised data transformations between LDES streams, linked data, and non-linked data sources. The **LDES Server**⁵ is a configurable component used to ingest, store, and publish these event streams. Finally, the **LDES Manager** is a custom graphical interface built by the authors to centrally configure and monitor the pipelines and streams within this openLDES environment.

On *project streams*, external project systems publish their raw asynchronous data updates directly to the append-only project streams. An LDIO Workbench pipeline processes these updates by converting them to RDF and applying SPARQL CONSTRUCT queries to align domain-specific vocabularies with Fundamend's internal conventions. Before merging these updates into the central knowledge graph, the platform validates the data against its internal schema shapes to ensure structural and semantic integrity. Conversely, *delivery streams* handle the dissemination of curated, verified information out to the client's asset management environment. When asset data is enriched or marked as approved within Fundamend, the platform converts these graph updates into RDF format based on the Fundamend schema. A secondary LDIO Workbench pipeline routes these objects to the LDES Server, which fragments and publishes them as a delivery stream conforming to the semantics of the EN 17632 standard and/or other desired vocabularies previously agreed with consumers. Downstream, an LDIO LDES client replicates this delivery stream and applies a consumer-owned mapping to populate the target system. Although Figure 1 suggests that the flow of information is one way, it can be reversed back to projects in case data from asset management systems is needed for maintenance or renovation projects. This architecture fundamentally shifts the integration contract from tightly coupled API integrations or custom endpoints to a stable, semantically defined data model. Consequently, the data exchange layer is decoupled from the underlying software infrastructure, allowing both publishing

²<https://openldes.org/>, <https://github.com/OpenLDES>

³<https://vlaamseoverheid.atlassian.net/wiki/spaces/VSDSSTART/overview>

⁴<https://github.com/OpenLDES/Linked-Data-Interactions>

⁵https://openldes.org/publisher/LDES_server, <https://github.com/OpenLDES/LDESServer>

platforms and consuming systems to evolve their internal schemas independently.

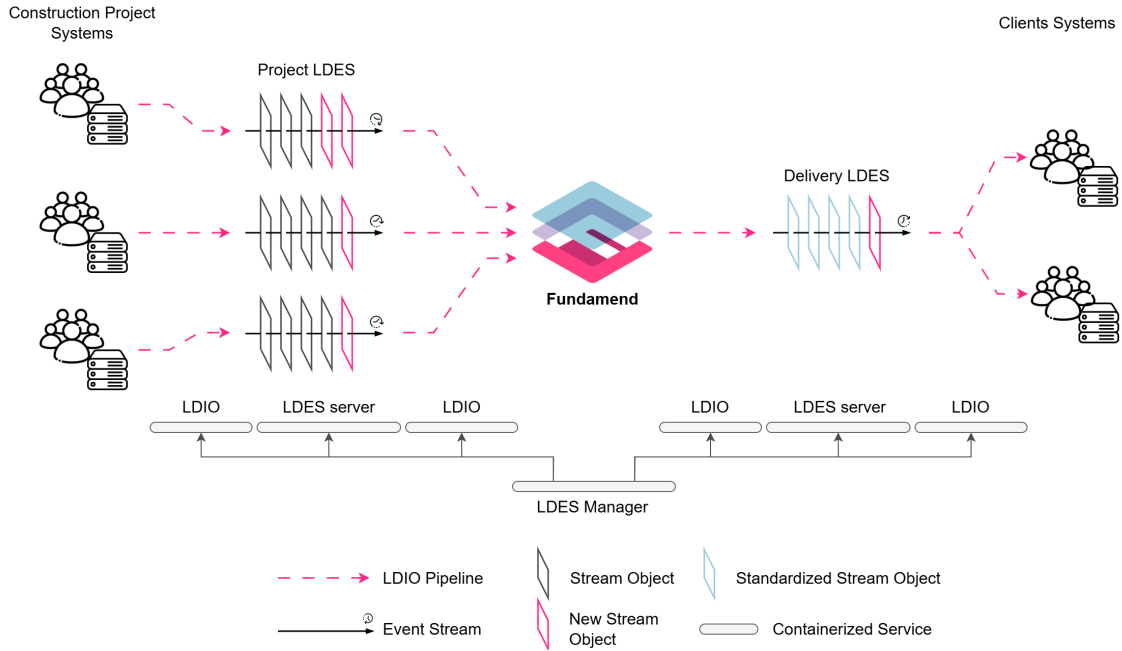


Figure 1: Overview of the LDES architecture spanning from source project systems to downstream client environments.

3. Example: Streaming Construction Progress of Walls

The approach is demonstrated through an experimental project that simulates the monitoring of the construction of a set of walls.⁶ For the sake of simplicity, the experiment exercises only the delivery side of the architecture described in Section 2; the ingestion side relies on the same components and is the subject of ongoing research. Each wall is represented in Fundament as a classified wall object instance, linked to its 3D geometric representation from a BIM model, and carrying a status property with one of three values (*planned*, *under construction*, or *completed*), which is updated through the platform as completion information arrives from the construction site (Figure 2, left).

A publisher pipeline in LDIO retrieves the modified walls from the staging graph, converts them to linked data through a JSON-LD context, and applies a SPARQL CONSTRUCT transformation that maps the Fundament representation onto a small client vocabulary agreed for the exchange and defined as an SML application vocabulary [3]: walls specialise `sm1:PhysicalObject`, and the status values are individuals of an `sm1:EnumerationType`. Each change is emitted as an immutable version object, linked to the stable IRI of the wall via `dct:isVersionOf` and timestamped with `prov:generatedAtTime`, following the versioning mechanism of the LDES specification. The OpenLDES Server appends these members to a stream constrained by a SHACL shape that enforces both the client data model and its SML alignment. It exposes it through three views with different fragmentation strategies: flat pagination, time-based fragmentation, and reference fragmentation by status value. A retention policy keeps the arbitrary number of versions of each wall (only 3 in the case of this test), balancing the need for bounding the storage growth of the stream and keeping a meaningful historical record.

On the consuming end, a second pipeline standing in for the client system replicates the stream through a generic LDES client. It materialises the members into a relational table, monitored through

⁶A video showing status changes applied in the platform and propagating to the dashboard of the simulated client system accompanies this submission, together with the client-side configuration and data of the experiment (the stream and view definitions, the SML application vocabulary with its SHACL shapes, example and captured stream members, the materialised client table, and the consumer pipeline): <https://drive.google.com/drive/folders/1yPv9DhUkStyE47iHy0g0zYdKIJVogfxK?usp=sharing>.

a dashboard as shown on the right of Figure 2. In the experimental setup, the publication pipeline was triggered on demand rather than by automatic change detection on the graph, which allowed the propagation of updates to be observed under controlled conditions. Deploying a continuous polling mechanism instead is straightforward, requiring only a change to the LDIO pipeline’s input configuration. We configured the platform and client system synchronisation to run in one-minute cycles, which is more than sufficient for the intended use. The value of this approach is the automated and incremental propagation of changes rather than low latency. Longer cadences would still remain adequate for construction project monitoring scenarios. Moreover, a newly connected consumer is able to replicate the retained history of the walls before switching to incremental synchronisation. Although the experiment involves a deliberately small dataset, the replication of fragmented, cacheable pages and the configurable retention policy are the mechanisms expected to carry larger data volumes; their limitation at a realistic project scale, however, is out of the scope of this experiment.

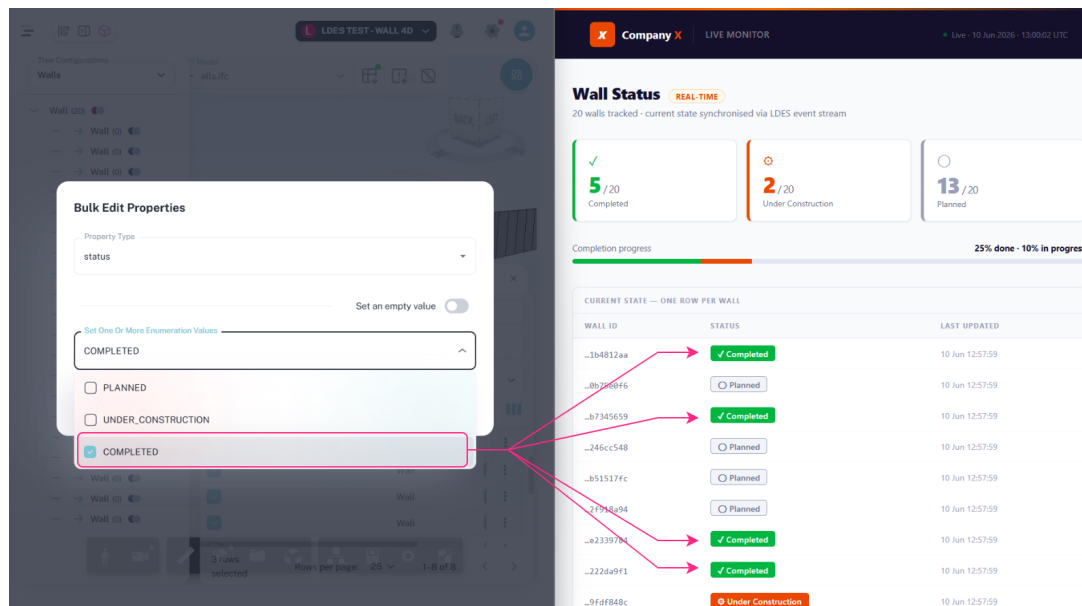


Figure 2: Status changes applied to a set of walls in Fundamend (left) propagate through the delivery stream and become visible in the monitoring dashboard of the simulated client system (right).

4. Benefits, Challenges and Opportunities

The experiment suggests several potential improvements over traditional file exchanges or point-to-point integrations in the construction sector. First, it automates data flows, eliminating manual data transfers and the need for custom calculations to identify updates. Second, by basing the data exchange on a shared, stable data model rather than specific software connections, data providers and users can modify or upgrade their internal software and data structures independently. Third, users only download new updates rather than full datasets, minimising network traffic and processing workloads, while a clear history of project changes is naturally preserved. Fourth, the server distributes unchanging data pages that can use standard web caching, keeping operational costs low and predictable. Finally, because features like data semantics, fragmentation, update timelines, and retention policies are fully configurable, stakeholders can agree on how the data will be managed before the exchange begins.

These benefits, however, remain to be validated beyond the experimental setting, and several challenges can already be anticipated. The approach presumes a degree of knowledge of linked data standards for the configurations and mappings, which is currently rare in the construction industry. It may require an investment in specialised external partners to accompany those interested stakeholders through the adoption process. Furthermore, data availability at the source remains a bottleneck, as streams are only as live as site processes feed them, and registration practices on construction sites

are uneven. Additionally, the construction industry relies heavily on geometrical data (e.g., BIM and GIS), resulting in complex and heavy datasets whose compatibility with LDES implementations remains to be fully explored. The most significant challenge, which is also the greatest opportunity, involves getting multiple stakeholders to adopt shared standards. Mapping different software systems to use the same data definitions requires substantial initial effort. However, unlike traditional integrations that must be custom-built for every new connection, a shared data stream serves as a reusable asset. As more project participants adopt the same standard, the effort required to connect any new system drops significantly. This turns the initial challenge of standard compliance into a scalable foundation that enables genuine, long-term interoperability across the industry.

5. Conclusion

This paper presented an ongoing extension of an existing construction information management platform with LDES, through which its consolidated knowledge graph can be exposed as standards-based, append-only event streams. The experimental use case indicates that information registered on-site can reach third-party systems incrementally and without per-update manual intervention, replacing bespoke point-to-point integrations with a single, reusable publication mechanism aligned with Semantic Web standards. While the results are encouraging, they were obtained with synthetic data in a controlled environment, and the approach still needs to be confronted with the data quality, scale, and organisational constraints of a live construction project. Future work is therefore directed towards improving the LDES setup, measuring performance gains internally related to the time dedicated to integrations and exploring the alignment of 3D geometries, GIS and other documents with linked data streams.

Declaration on Generative AI

During the preparation of this work, the authors used Gemini for grammar and spelling checks. Further, the authors used Claude Code for editing the code for the experiments presented. After using these services, the authors reviewed and edited the content as needed and take full responsibility for the publication's content.

References

- [1] S. T. Matarneh, M. Danso-Amoako, S. Al-Bizri, M. Gaterell, R. Matarneh, Building information modeling for facilities management: A literature review and future research directions, *Journal of Building Engineering* 24 (2019) 100755. doi:10.1016/j.jobe.2019.100755.
- [2] C. Ramonell, R. Chacón, H. Posada, Knowledge graph-based data integration system for digital twins of built assets, *Automation in Construction* 156 (2023) 105109. doi:10.1016/j.autcon.2023.105109.
- [3] European Committee for Standardization, EN 17632-1:2022 — Building information modelling (BIM) — Semantic modelling and linking (SML) — Part 1: Generic modelling patterns, Technical Report, CEN, Brussels, Belgium, 2022.
- [4] SEMIC, Linked data event streams (LDES), version 1.0.0-alpha, <https://semiceu.github.io/LinkedDataEventStreams/>, 2024.
- [5] P. Colpaert, The TREE hypermedia specification, W3C Draft Community Group Report, <https://w3id.org/tree/specification>, 2025.
- [6] D. Van Lancker, P. Colpaert, H. Delva, B. Van de Vyvere, J. R. Meléndez, R. Dedecker, P. . Michiels, R. Buyle, A. De Craene, R. Verborgh, Publishing base registries as linked data event streams, in: *Web Engineering (ICWE 2021)*, volume 12706 of *Lecture Notes in Computer Science*, Springer, 2021, pp. 28–36. doi:10.1007/978-3-030-74296-6_3.