Operational Data Foundation Framework for Smart Manufacturing in SMEs: Field Implementation and Evaluation

Yoonseok Kang and Dongchul Park *

Posted Date: 10 April 2026

doi: 10.20944/preprints202604.0762.v1

Keywords: smart manufacturing; SME digitalization; operational data foundation; industrial data infrastructure; data governance

 doi:10.20944/preprints202604.0762.v1

Article

Operational Data Foundation Framework for Smart Manufacturing in SMEs: Field Implementation and Evaluation

Yoonseok Kang [1] and Dongchul Park [2,*]

[1] Department of Convergence Security, Chung-Ang University, Seoul 06974, South Korea
[2] Department of Industrial Security, Chung-Ang University, Seoul 06974, South Korea
* Correspondence: dongchul@cau.ac.kr

Abstract

Smart manufacturing depends on operational data that remain continuous, interpretable, and reusable in practice. In constrained small and medium-sized enterprise (SME) factories, however, the main bottleneck often lies not in later-stage analytics or AI applications, but in securing an operationally viable data foundation under real deployment conditions. A lifecycle-based analysis of smart manufacturing data pipelines, together with recurrent SME deployment constraints identified in prior studies, led this study to derive six recurring operational risks. On that basis, the study proposes an *Operational Data Foundation Framework* structured around core requirements of continuity, governance, diagnosability, operability, reprocessability, and evolvability. These requirements are further articulated through design principles and assessable operational invariants. The framework was instantiated in a real SME factory, where heterogeneous field sources were integrated into a coherent operational data foundation for smart manufacturing through constrained communication paths, durable edge-side capture, cloud-side stream processing, controlled data normalization, and monitoring and alerting functions. Requirement-based evidence from the field implementation showed that the system preserved stable semantics across the pipeline, made failures traceable to specific lifecycle segments, preserved historical records for later reprocessing, and remained manageable under constrained deployment conditions. A representative field case further demonstrated the framework's practical value: severe communication instability was diagnosed through lifecycle-segment discrepancy analysis and improved from approximately 33% to 95% packet reception after targeted intervention. The study contributes a field-grounded and assessable design logic for making smart manufacturing practically achievable in constrained SME factories.

Keywords: smart manufacturing; SME digitalization; operational data foundation; industrial data infrastructure; data governance

1. Introduction

Smart manufacturing is transforming manufacturing from a production environment centered primarily on physical equipment and isolated processes into a connected operational system built on continuous data flows [1,2]. Across the factory, sensing, communication, computation, and decision making are becoming increasingly integrated through technologies such as the Internet of Things (IoT), cyber–physical systems (CPS), artificial intelligence and machine learning (AI/ML), real-time monitoring, predictive maintenance, and digital twins [3–5]. Manufacturing systems should therefore be understood not only as physical production arrangements, but also as data-mediated operational systems [6].

In smart manufacturing, data no longer serve merely as records of what has already happened. They have become a core operational resource that supports visibility, traceability, quality control, maintenance, and data-driven decision making [2,7]. As a result, the practical progress of smart

 doi:10.20944/preprints202604.0762.v1

manufacturing increasingly depends on whether operational data can be captured, interpreted, and reused in a stable and sustained manner [8].

Higher-level capabilities such as predictive analytics, digital twins, and AI-driven optimization do not become operational simply by being introduced. They depend on an underlying data layer that can continuously capture field conditions, preserve meaning across system boundaries, and support downstream reuse [9,10]. The central issue, therefore, is not only what manufacturing systems can do intelligently, but what kind of data foundation must first be maintained to make such intelligence workable in practice.

The value of manufacturing data does not lie in data generation alone. It is realized across an end-to-end flow in which field data are acquired, transmitted, transformed, stored, and reused downstream [11,12]. Manufacturing data problems should therefore be understood not as isolated database or interface issues, but as lifecycle-wide operational systems problems [13]. This challenge becomes more acute in industrial environments, where heterogeneous devices, protocols, timestamps, machine states, manual records, and fragmented platforms coexist [14,15]. Under such conditions, missingness, duplication, delay, semantic inconsistency, and timestamp ambiguity directly weaken traceability, operational visibility, and decision quality [16,17].

What is required, therefore, is not merely an ingestion infrastructure, but an operational layer that can preserve continuity, governed semantics, durable persistence, observability, and downstream reusability across that lifecycle [16,18]. In this study, we refer to this layer as the *operational data foundation*. Rather than functioning as a simple conduit for data movement, it stabilizes heterogeneous field data and makes them dependable for sustained operational and analytical use. As illustrated in Figure 1, this layer occupies the space between heterogeneous operational data sources and higher-level operational and analytical applications.

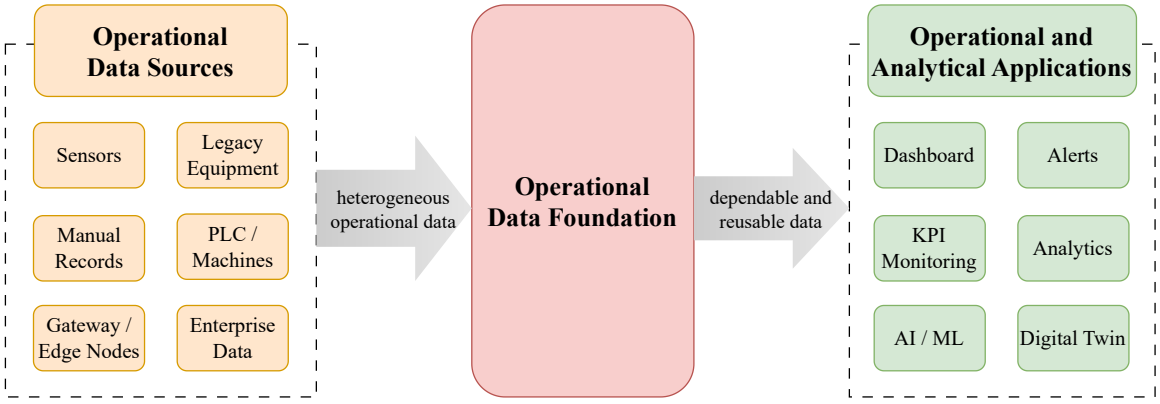

Figure 1. Position of the operational data foundation in a digitalized manufacturing system.

Although dependable manufacturing data systems are difficult to build in general, their practical design requirements become most visible in constrained SME factories [19,20]. These environments rarely provide the conditions often assumed in digitally mature smart-manufacturing settings, such as standardized interfaces, stable internal connectivity, dedicated technical support, and integrated data infrastructure [21,22]. Instead, they are more commonly characterized by legacy equipment, fragmented systems, mixed communication technologies, manual records, limited technical staffing, and strong cost sensitivity [23,24]. SME factories should therefore be understood not merely as smaller plants, but as critical design settings in which the practical assumptions of smart manufacturing data systems are tested most directly [20].

In such environments, the main bottleneck of digitalization does not lie simply in deploying more sensors, collecting more data, or adding downstream applications. Rather, it lies in whether operational data can be captured with continuity, stabilized with interpretable meaning, and sustained as a usable resource despite fragmentation, instability, and limited operational slack. The central difficulty in SME

 doi:10.20944/preprints202604.0762.v1

manufacturing is therefore not merely data scarcity, but the fragility of the operational data lifecycle through which field data must become dependable for ongoing operational use [16,25,26].

This implies that the primary bottleneck of smart manufacturing in constrained SME factories lies not in downstream intelligence itself, but in establishing an operationally viable data foundation. Accordingly, this study examines how an operational data foundation should be designed, and what it must guarantee, in order to make smart manufacturing practically viable in constrained SME factories.

Existing research has approached manufacturing data systems from several complementary directions. Smart manufacturing studies have mainly focused on higher-level applications such as real-time monitoring, predictive maintenance, digital twins, and adaptive production support [1,3,5]. Research on industrial data infrastructures has examined pipelines, integration architectures, edge–cloud coordination, governance, and data quality [11,12,16]. Lifecycle-oriented work has emphasized cross-stage linkage, traceability, contextual continuity, and the preservation of reusable records [8,13,27,28]. Meanwhile, studies on SME digitalization have clarified the realities of constrained deployment, including legacy equipment, fragmented systems, limited staffing, and cost sensitivity [20–22,24].

However, these contributions remain only partially integrated when the problem is viewed from the standpoint of operational viability under constrained deployment conditions. Prior work has often examined applications, architectural components, lifecycle properties, or SME constraints separately, rather than formalizing what a manufacturing data foundation itself must guarantee in order to remain usable in practice. As a result, there is still limited systems-level understanding of how such guarantees should be defined, structured, and assessed in constrained SME manufacturing environments.

To address this gap, this study focuses on the foundational layer presupposed by smart manufacturing applications and formalizes it in terms of operational requirements, design principles, and assessable invariants. More specifically, it develops an *Operational Data Foundation Framework* for constrained SME factories and examines it not only conceptually, but also through field implementation and operational evidence from a real SME manufacturing deployment.

On this basis, the study is guided by the following research questions:

- **RQ1.** What operational risks emerge when smart manufacturing data pipelines are deployed under constrained SME factory conditions?
- **RQ2.** What core requirements must an operational data foundation satisfy under such conditions, and how can these requirements be formalized into design principles and operational invariants?
- **RQ3.** How can the proposed framework be instantiated in a real SME manufacturing setting, and how can its practical value be evaluated through field evidence?

To address these questions, the study proceeds in four steps. First, it establishes a general lifecycle model of smart manufacturing data pipelines as an analytical reference. It then analyzes how recurrent constraints identified in prior SME research act across that lifecycle and derives six operational risks that undermine practical viability. On that basis, it formalizes the Operational Data Foundation Framework by translating those risks into core requirements, design principles, and operational invariants. Finally, it instantiates the framework through field implementation in a real SME manufacturing setting and evaluates its practical value through requirement-based evidence and representative field observations.

The primary contributions of this research are as follows:

1. Reframing the practical bottleneck of smart manufacturing in constrained SME factories as an operational data foundation problem.
2. Deriving the core operational risks that emerge when a general manufacturing data lifecycle is exposed to constrained SME factory conditions.
3. Formalizing the Operational Data Foundation Framework in terms of core requirements, design principles, and operational invariants.
4. Demonstrating the framework's field-grounded operational relevance through real-world implementation in an SME manufacturing environment.

 doi:10.20944/preprints202604.0762.v1

The remainder of this paper is organized as follows. Section 2 reviews the related literature. Section 3 analyzes the general data pipeline lifecycle, SME-specific constraints, and the resulting operational risks. Section 4 presents the proposed Operational Data Foundation Framework. Section 5 describes the field implementation in an SME manufacturing setting. Section 6 evaluates the framework using requirement-based evidence and field observations. Finally, Section 7 concludes the paper.

2. Related Work

2.1. Smart Manufacturing Applications

Smart manufacturing research has developed primarily around higher-level applications and intelligent capabilities, including industrial IoT, cyber–physical systems, digital twins, real-time monitoring, predictive maintenance, and AI/ML-enabled decision making [29–32]. Across these streams, the dominant aim has been to make manufacturing systems more connected, adaptive, and data-driven through enhanced visibility, prediction, and control [1,29,30].

Despite their diversity, these applications share a common prerequisite: continuous or near-real-time operational data from the shop floor [3–5]. Studies on digital twins and real-time manufacturing intelligence repeatedly show that higher-level functions depend on data that are accurate, timely, and interpretable, and that remain aligned with physical system states [5,33,34]. Yet this literature is generally more explicit about what smart manufacturing applications can optimize, predict, or automate than about what kind of data basis must be sustained for those functions to remain operationally viable in practice [5,29,35].

2.2. Industrial Data Infrastructures

A substantial body of research has examined the industrial data basis through manufacturing data pipelines, integration architectures, industrial data management systems, and edge–cloud infrastructures [11,12,36]. This literature makes clear that manufacturing data handling is not merely a storage problem, but a multi-layer systems problem spanning acquisition, communication, processing, storage, and downstream delivery across heterogeneous sources [11,37].

Within this stream, two themes are especially prominent. First, many studies address heterogeneous integration and interoperability across legacy equipment, devices, protocols, platforms, and enterprise systems [14,15,38]. Second, related work has emphasized semantics, context, and governance, arguing that industrial data must remain interpretable, trustworthy, and reusable rather than merely transferable [16,39,40]. These contributions have significantly advanced understanding of how industrial data infrastructures can be connected, organized, and managed. However, they remain more focused on components, architectures, and enabling techniques than on explicitly formalizing what such infrastructures must guarantee as an operational foundation under deployment constraints [11,12,16].

2.3. Lifecycle Connectivity

Another important line of research treats manufacturing data not as stage-local records, but as lifecycle-wide resources. From this perspective, the value of manufacturing data depends on whether information can remain linked, contextualized, and reusable across design, production, inspection, operation, maintenance, and end-of-life stages [2,7,8,13]. This perspective shifts attention away from isolated data handling and toward continuity of meaning and usability across lifecycle transitions.

The concept of the digital thread has become the clearest expression of this perspective. Recent reviews and implementation studies define the digital thread as a means of linking information across heterogeneous lifecycle artifacts and stages so that traceability, cross-stage visibility, and lifecycle-wide reuse become possible beyond fragmented system silos [27,28,41,42]. Related studies on contextualization, standards-based linkage, and knowledge-graph integration likewise show that lifecycle usefulness depends not merely on storing records, but on preserving associativity, context, and interpretability for later reuse [43–45]. At the same time, this literature has focused mainly on linkage and

 doi:10.20944/preprints202604.0762.v1

traceability, with relatively less explicit discussion of how such continuity and reuse can be sustained operationally under constrained deployment conditions.

2.4. SME Deployment

The SME literature makes those deployment constraints explicit. Studies on smart manufacturing adoption in SMEs consistently show that these environments differ substantially from idealized smart factory assumptions: legacy equipment, fragmented systems, uneven interfaces, limited digital infrastructure, restricted investment capacity, and uneven technical expertise are common [19–22]. In this sense, SMEs should not be viewed simply as smaller versions of large factories, but as constrained deployment settings in which the assumptions of digitally mature architectures are most directly challenged.

These constraints are both technical and operational. Brownfield dependence often forces SMEs to rely on retrofitting, add-on sensing, gateway-based integration, and low-cost connectivity rather than wholesale replacement [23,46–48]. At the same time, recurring barriers include limited staffing, weak digital skills, uncertain return on investment, high implementation cost, and maintenance burden [24–26,49]. Accordingly, SME-oriented studies increasingly advocate phased, modular, and capability-building adoption paths [20,50,51]. Taken together, these studies show that the unresolved problem is not how to deploy smart manufacturing applications, infrastructure components, or lifecycle linkage mechanisms, but how to sustain a data foundation that remains operationally viable under constrained real-world conditions.

3. Problem Analysis

This section analyzes the data basis of smart manufacturing as an operational pipeline and examines how constrained SME factory conditions introduce structural fragility across that pipeline. Rather than merely describing deployment difficulty, it identifies how lifecycle-wide constraints converge into a small set of recurring operational risks. In this way, the section defines the problem structure of operational data foundations under constrained deployment conditions.

3.1. General Data Pipeline Lifecycle

A common analytical reference is needed before constrained SME conditions and their resulting risks can be examined in a structured manner. In smart manufacturing, data do not become operationally useful at the moment of generation alone. Their practical value emerges only through a broader path in which field observations are accepted into a pipeline, transferred across infrastructure layers, transformed into usable forms, preserved, and connected to downstream operational use [11,12]. For this reason, this study adopts a *general data pipeline lifecycle* as an analytical frame for problem interpretation rather than as a normative system architecture.

The lifecycle used in this study is shown in Figure 2. It is adapted from generic data pipeline literature, which treats data systems as multi-stage processing structures characterized by recurring issues related to ingestion, integration, cleaning, transformation, loading, and pipeline-wide quality control [17]. In the smart manufacturing context, this general view is organized as follows:

- **Data sources:** the points at which operational data originate, including sensors, equipment states, manual records, and external systems.
- **Data ingestion:** the intake of source data into the pipeline.
- **Data transmission:** the movement of acquired data across gateway, edge, server, or cloud boundaries toward the next processing point.
- **Data preprocessing:** the cleaning, decoding, normalization, mapping, and transformation required to make raw inputs usable downstream.
- **Data loading:** the controlled insertion of processed records into the target operational layer or storage boundary.

 doi:10.20944/preprints202604.0762.v1

- **Data storage:** the durable preservation of records for later access, recovery, reinterpretation, and reuse.
- **Data sinks:** the downstream endpoints that consume stored data, including dashboards, alerts, reporting, analytics, and decision support.

This lifecycle is analytically useful because it makes visible the full path through which manufacturing data become operationally usable. At the same time, it does not by itself provide a design answer. It describes how data move, but not what must be preserved for the lifecycle to remain operationally viable under constrained conditions. It is therefore used in this study as an analytical reference for the constraint and risk analysis that follows.

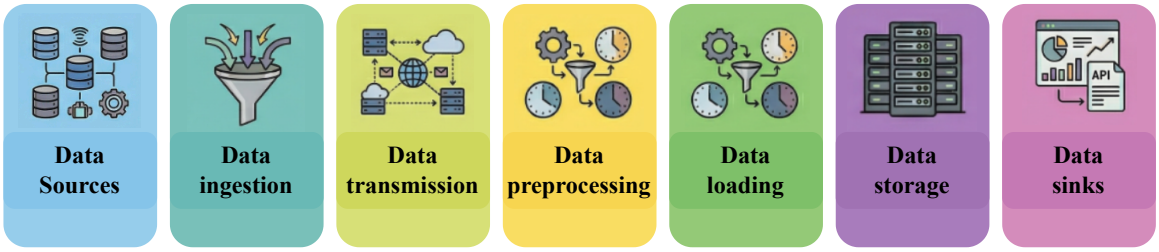

Figure 2. General data pipeline lifecycle in smart manufacturing.

3.2. *SME Constraints*

Constrained SME factories should not be treated simply as smaller versions of digitally mature plants. Prior research shows that their starting conditions are structurally different: legacy equipment, fragmented systems, uneven interfaces, restricted digital infrastructure, limited internal expertise, and strong investment sensitivity are common, whereas standardized connectivity and unified data access are often absent [19–22,52]. SME constraints are therefore better understood not as isolated implementation difficulties, but as recurring design conditions that act across the general data pipeline lifecycle. Figure 3 highlights this point by showing how the major constraint types propagate across source capture, transfer, transformation, storage, and downstream use, rather than remaining confined to a single stage.

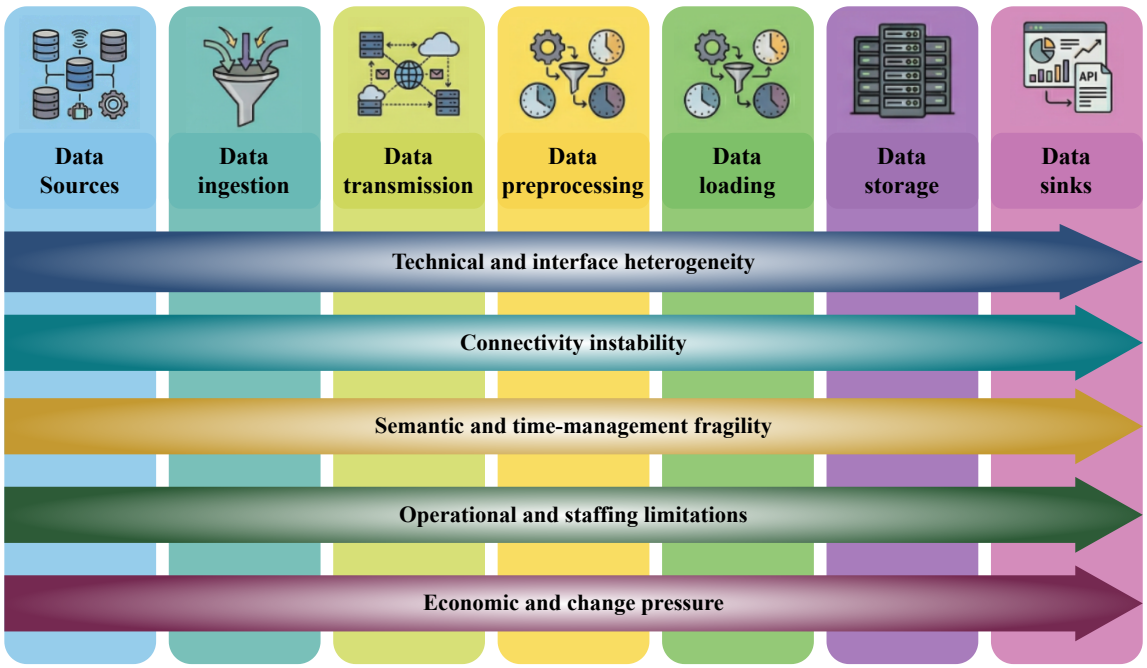

Figure 3. SME constraints across the general data pipeline lifecycle.

 doi:10.20944/preprints202604.0762.v1

Heterogeneity begins at the source side. In SME factories, brownfield machinery, retrofit-based extension, add-on sensing, and gateway-mediated access are common, so inconsistency is already present at the level of data sources and ingestion rather than arising only later during integration [23,46–48]. The challenge is not merely to collect values, but to incorporate diverse formats, units, reporting intervals, and source identities into a coherent operational structure. Source-side variability is therefore not a peripheral inconvenience, but the starting condition from which later coordination and interpretation problems emerge.

Transmission is often fragile in constrained SME deployments. Field data may depend on low-cost wireless communication, battery-powered sensing, and gateway- or edge-dependent relay paths rather than on stable plant-wide infrastructure [53–55]. The ingestion–transmission path therefore becomes a distinct surface of instability. What matters operationally is not simply whether a value was observed once, but whether the record can remain sufficiently continuous and recoverable despite interruption, delay, and relay dependence.

Operational meaning must remain stable across the pipeline. Manufacturing data are operationally usable only when source identity, metric semantics, unit interpretation, and time semantics remain controlled across preprocessing, loading, and storage. Yet constrained environments frequently rely on heterogeneous preprocessing logic, ad hoc mappings, incomplete context, and weak contract discipline, making semantic drift and traceability loss structurally likely [15,16,39,40,56]. Once such instability becomes embedded in stored records, downstream analytics and dashboards may continue to function while resting on an increasingly fragile semantic basis.

These technical difficulties are intensified by operational realities. SME studies repeatedly report limited staffing, uneven digital skills, manual troubleshooting, high maintenance burden, and weak strategic capacity [24,25,49,57]. At the same time, SME digitalization rarely proceeds as a one-shot transformation; instead, it unfolds through phased rollout, retrofit-based extension, KPI revision, and incremental onboarding of new sources and processes [50,51,58,59]. As a result, the practical challenge is not only whether a system can be built, but whether it can remain diagnosable, manageable, and adaptable as requirements evolve.

Taken together, these observations show that the core problem in constrained SME factories is not simply that deployment is difficult. Rather, the assumptions of the general data pipeline lifecycle are systematically destabilized across the lifecycle itself: records may lose continuity, meanings may drift, failures may become difficult to localize, operational burden may increase, historical data may lose replay value, and later extension may require disproportionate rework. The next subsection translates these recurring forms of fragility into the language of operational risk.

3.3. Operational Risks

The SME constraints discussed above do not remain isolated deployment difficulties. When mapped onto the general data pipeline lifecycle, they appear as recurring operational risks that determine whether manufacturing data can remain usable in practice. The issue, therefore, is not only how data move through the pipeline, but also what must be preserved for the resulting record to remain dependable under constrained real-world conditions.

RSK1. **Continuity risk.** Operational records may become interrupted, fragmented, or partially lost as data move from source capture to storage. In constrained manufacturing settings, the essential requirement is not one-time collection success, but the preservation of a gap-minimized and recoverable record despite source instability, transmission disruption, or delayed recovery.

RSK2. **Governance risk.** Data may remain available while losing stable meaning. This occurs when source identity, metric semantics, timestamp interpretation, or transformation logic are not kept consistent across collection, preprocessing, loading, and storage. Under such conditions, downstream analysis may continue to operate, but it does so on data whose meaning is no longer reliably governed.

 doi:10.20944/preprints202604.0762.v1

RSK3. Diagnosability risk. When failure occurs, it may be difficult to determine where in the lifecycle the problem emerged and what kind of failure it is. Weak monitoring, poor stage separation, and limited observability turn degradation into a black-box symptom, making it difficult to distinguish among transmission loss, semantic corruption, storage inconsistency, and downstream mismatch in a structurally interpretable way.

RSK4. Operability risk. A system may function technically while becoming too costly or complex to sustain in everyday practice. Tight coupling, heavy maintenance burden, unclear ownership, and non-modular structures increase the effort required to manage, modify, and repair the system, making long-term operation difficult under constrained SME staffing and budget conditions.

RSK5. Reprocessability risk. Historical data may cease to be usable when rules, KPIs, schemas, or interpretation logic change. If raw preservation is weak, preprocessing is effectively irreversible, or lineage is incomplete, past records cannot be reliably replayed, recomputed, or reinterpreted. In that case, the data foundation loses one of its most important operational values: the ability to support correction and renewed interpretation over time.

RSK6. Evolvability risk. New sensors, processes, lines, factories, or downstream requirements may become difficult to incorporate without disproportionate redesign cost or structural disruption. Because SME digitalization usually proceeds through incremental onboarding rather than one-shot integration, a data foundation that cannot absorb change without breaking governance and operational control cannot serve as a durable basis for growth.

Taken together, these risks show that a smart manufacturing data pipeline in constrained SME factories cannot be treated as a simple ingestion flow. The problem is not a collection of isolated device, network, or software issues, but a structural fragility in the lifecycle through which operational data become usable. What is required, therefore, is a foundation that can explicitly preserve continuity, governance, diagnosability, operability, reprocessability, and evolvability under constrained deployment conditions. The next section presents the *Operational Data Foundation Framework* as a structured response to these risks.

4. Operational Data Foundation Framework

To respond to these risks, this study proposes the *Operational Data Foundation Framework*. As shown in Figure 4, the framework is organized as a risk-to-design logic. *Operational risks* are translated into *core requirements*, these requirements are realized through *design principles*, and their fulfillment is assessed through *operational invariants*. In this way, the framework provides an explicit and assessable basis for the design and evaluation of an operational data foundation under constrained deployment conditions.

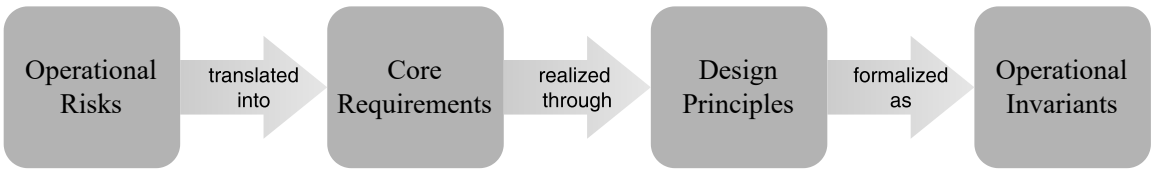

Figure 4. Operational Data Foundation Framework.

4.1. Core Requirements

The framework begins by defining the *core requirements*. These requirements are not merely desirable properties in a general sense, but the minimum lifecycle-wide guarantees needed for operational viability under constrained SME factory conditions. They provide the basis for the design principles and operational invariants developed in the following subsections.

Six core requirements are defined.

R1. Continuity. Data continuity should be preserved as far as practicable despite field disruption and connectivity instability. The essential requirement is not momentary acquisition success, but

 doi:10.20944/preprints202604.0762.v1

the minimization of gaps and the preservation of recoverable continuity so that the operational record remains usable even when intermediate segments are unstable.

R2. **Governance.** Source identity, metric semantics, and time semantics should remain stable across collection, normalization, and storage. Data are not operationally usable merely because they exist; they must retain controlled meaning and interpretable context throughout the pipeline.

R3. **Diagnosability.** Failures should be observable as segment-level discrepancies rather than only as end-to-end symptoms. The requirement is not simply that problems become visible after downstream effects appear, but that the lifecycle location and mode of failure can be identified in a structurally interpretable manner.

R4. **Operability.** The system should remain maintainable and manageable under constrained staffing and budget conditions. An operational data foundation must therefore be sustainable in day-to-day practice without imposing a structurally excessive maintenance burden or operational complexity.

R5. **Reprocessability.** Historical data should remain reprocessable for rule changes, KPI revisions, correction, and recovery. Manufacturing data should not be treated as one-time consumables, but as records that can be revisited, recalculated, and reinterpreted as operational needs evolve.

R6. **Evolvability.** The foundation should accommodate new sources, processes, sites, and applications without breaking the existing structure. Because SME digitalization typically proceeds through incremental onboarding rather than one-time integration, the data foundation must remain structurally open to change and extension.

4.2. Design Principles

The core requirements defined above specify what the operational data foundation must guarantee, but not how those guarantees should be realized in design. This subsection therefore presents the key *design principles* that translate those requirements into a workable design logic for constrained SME factory conditions. Rather than prescribing specific technologies or implementation choices, these principles define the structural directions needed to sustain the required operational guarantees.

P1. **Continuity responsibility should be placed close to the source.** If continuity depends primarily on downstream transmission or cloud reachability, it becomes fragile under field disruption and connectivity instability. For this reason, the primary responsibility for preserving continuity should be placed as close to the source as possible through source-near capture, upstream buffering or persistence, and recovery-oriented ingestion logic.

P2. **Raw events should be preserved before abstraction.** If only transformed outputs are retained, later replay, reinterpretation, and recovery become structurally constrained. Preserving raw events before irreversible abstraction is therefore a fundamental design principle, since rule changes, KPI revision, and correction all depend on access to preserved pre-abstraction records.

P3. **Semantics should be contract-defined.** If source identity, metric naming, timestamp semantics, unit definition, and transformation logic are left to ad hoc implementation, semantic inconsistency accumulates across the pipeline. The design must therefore define semantics explicitly through controlled contracts, schema discipline, version-aware rules, and clear conventions for identity and time.

P4. **Observability should follow pipeline segments.** Diagnosability cannot be secured through black-box end-to-end monitoring alone, because operationally meaningful failures must be localizable to specific lifecycle segments. Observability should therefore be structured along the pipeline itself through segment-level checkpoints, stage-level indicators, and explicit discrepancy visibility.

P5. **Responsibilities should be separated by function.** If acquisition, normalization, storage, and downstream use are tightly coupled, both maintenance burden and change cost rise rapidly. Functional responsibilities should therefore be separated so that components remain structurally independent, interfaces remain replaceable, and modifications in one part do not unnecessarily destabilize the whole.

 doi:10.20944/preprints202604.0762.v1

P6. Operations should remain lightweight, modular, and open. In constrained SME environments, heavy operational structures, costly maintenance models, and vendor dependence undermine long-term viability. The design should therefore remain lightweight in deployment, modular in composition, manageable in maintenance scope, and open to replacement and extension. Where appropriate, open and widely supported components, including open-source tools, can help reduce lock-in and maintenance burden.

P7. Change should be treated as a first-class assumption. Changes in KPIs, schemas, rules, source onboarding, and deployment scope are not exceptional events, but normal operating conditions in SME digitalization. The foundation should therefore be designed from the outset to accommodate and organize change through version-aware structures, replay support, incremental onboarding discipline, and change-tolerant interfaces.

4.3. Operational Invariants

The core requirements and design principles defined above specify what the operational data foundation must guarantee and how those guarantees should be realized. This subsection completes the framework by defining the *operational invariants*, that is, the persistent conditions that should hold if the foundation is functioning as intended.

I1. Durable Capture Invariant. An accepted event should obtain durable persistence before any downstream failure can erase it. Continuity should therefore be judged from the point of first durable capture rather than from downstream availability alone.

I2. Traceability Invariant. Every canonical or stored record should remain traceable to its source identity and operational context. Downstream data should not become detached from the source, metadata, and governing semantics that give them meaning.

I3. Deterministic Normalization Invariant. The same raw event under the same contract and version conditions should always produce the same canonical output. Normalization should function as a repeatable and controlled semantic transformation rather than as ad hoc interpretation.

I4. Segment-Localizable Failure Invariant. A failure should become visible as at least one discrepancy within a recognizable lifecycle segment. It should be possible to attribute the problem to a specific segment, such as the source, transmission, preprocessing, loading, storage, or sink, rather than observing it only as a downstream symptom.

I5. Replayability Invariant. Preserved raw data should remain reprocessable for rule changes, KPI revision, correction, and recovery. Historical data should therefore remain usable for later reinterpretation and recomputation, rather than serving only as passive archival material.

I6. Modular Replacement Invariant. Replacing a specific component should not break the semantic chain of the overall foundation. Changes in storage, parsing, transport, or downstream components should not destroy source identity, metric meaning, or replay context across the system.

I7. Onboarding Consistency Invariant. A new source should be incorporable into the existing structure without breaking the prevailing governance discipline. Incremental onboarding should proceed through the same contract, schema, naming, and validation discipline, rather than through an accumulation of ad hoc exceptions.

These invariants define the persistent conditions by which the framework can be assessed in practice. If requirements declare the guarantees to be achieved and principles define the design logic needed to achieve them, invariants provide the basis for evaluating whether those guarantees remain structurally preserved.

4.4. Framework Synthesis

This subsection synthesizes the framework by reconnecting it to the problem space analyzed in Section 3. Section 3 showed how constrained SME factory conditions destabilize the general data pipeline lifecycle and generate recurring risks related to continuity, governance, diagnosability,

 doi:10.20944/preprints202604.0762.v1

operability, reprocessability, and evolvability. Sections 4.1–4.3 then translated those risks into core requirements, design principles, and operational invariants. The purpose of this subsection is to clarify how these layers are systematically linked to the underlying problem structure.

Table 1 provides the first synthesis. It shows that constrained SME conditions are not merely a list of deployment difficulties, but recurring operational design problems that can be translated systematically. Each major constraint category gives rise to one or more operational risks; each risk is then expressed as a core requirement, translated into one or more design principles, and made assessable through corresponding invariants. In this way, the table shows how the framework is grounded in recurring structural problems in constrained SME manufacturing.

Table 1. From SME constraints to operational design logic.

SME Constraint	Operational Risk	Requirements	Principles	Invariants
Heterogeneous devices and fragmented interfaces	RSK2 (Governance risk)	R2 (Governance)	P3 (Semantics must be contract-defined)	I2 (Traceability), I3 (Deterministic normalization)
Add-on sources and partial visibility	RSK2, RSK6 (Evolvability risk)	R2, R6 (Evolvability)	P3, P7 (Change as a first-class assumption)	I2, I7 (Onboarding consistency)
Unstable connectivity and constrained sensing	RSK1 (Continuity risk)	R1 (Continuity)	P1 (Continuity responsibility must be placed close to the source)	I1 (Durable capture)
Weak cross-segment observability	RSK3 (Diagnosability risk)	R3 (Diagnosability)	P4 (Observability must follow pipeline segments)	I4 (Segment-localizable failure)
Limited staff and high maintenance burden	RSK4 (Operability risk)	R4 (Operability)	P6 (Operations must remain lightweight, modular, and open)	I6 (Modular replacement)
Weak raw-data preservation	RSK5 (Reprocessability risk)	R5 (Reprocessability)	P2 (Raw events must be preserved before abstraction)	I5 (Replayability)
Tight functional coupling across the pipeline	RSK3, RSK4, RSK6	R3, R4, R6	P5 (Responsibilities must be separated by function)	I4, I6
Frequent KPI, rule, schema, and source changes	RSK5, RSK6	R5, R6	P7	I5, I7
Budget sensitivity and lock-in concerns	RSK4, RSK6	R4, R6	P6	I6, I7

One particularly important translation highlighted in Table 1 is the shift from semantic and interface heterogeneity to governance logic. Here, the central issue is not merely integration, but the need to keep source identity, metric meaning, unit interpretation, and timestamp semantics stable across the pipeline. The framework therefore links this problem to *R2 Governance*, *P3 contract-defined semantics*, and the assessable conditions of *I2 Traceability* and *I3 Deterministic Normalization*. This translation makes clear that manufacturing data become operationally reliable only when meaning and origin remain stably governed.

A second important translation concerns the shift from change pressure to reprocessability and evolvability logic. Here, the central issue is whether the foundation can remain coherent as KPIs, schemas, sources, and operational rules evolve over time. The framework therefore links this problem to *R5 Reprocessability* and *R6 Evolvability*, supported by *P7 change as a first-class assumption* and assessed through *I5 Replayability* and *I7 Onboarding Consistency*. This shows that the framework is designed not only for present-time correctness, but also to preserve operational coherence under future change.

Table 2 provides a second synthesis by showing how the framework adds operational meaning to the general data pipeline lifecycle. Under constrained SME conditions, each lifecycle stage must contribute not only to data movement and processing, but also to the guarantees defined by the operational data foundation. The table makes this added meaning explicit by connecting each stage to

 doi:10.20944/preprints202604.0762.v1

the relevant risks, requirements, design principles, and invariants. In this way, the general lifecycle is retained as an analytical reference, while its role is sharpened for constrained manufacturing conditions.

Table 2. Mapping the general data pipeline lifecycle to the operational foundation logic.

General Lifecycle Stage	Added Operational Meaning	Related Risks	Requirements	Principles	Invariants
Data sources	Stable source identity	RSK2, RSK6	R2, R6	P3, P7	I2, I7
Data ingestion	Source-near continuity and intake control	RSK1, RSK2	R1, R2	P1, P3	I1, I2
Data transmission	Recoverable transfer and visible discrepancies	RSK1, RSK3	R1, R3	P1, P4	I1, I4
Data preprocessing	Contract-defined and deterministic transformation	RSK2, RSK6	R2, R6	P3, P7	I2, I3, I7
Data loading	Controlled persistence with replay context	RSK5, RSK6	R5, R6	P2, P7	I5, I7
Data storage	Replay-ready and replaceable persistence	RSK4, RSK5	R4, R5	P2, P6	I5, I6
Data sinks	Governed downstream reuse	RSK2, RSK5	R2, R5	P3, P7	I2, I5
Monitoring and management	Lifecycle-wide observability and maintainability	RSK3, RSK4	R3, R4	P4, P6	I4, I6

One especially revealing example is the reinterpretation of the transmission stage. In generic pipeline discussions, transmission is often treated as a routine internal link between ingestion and subsequent processing. In constrained SME factory conditions, however, the path from field devices to gateways, edge systems, servers, and cloud systems is often exposed to intermittent connectivity, limited bandwidth, battery constraints, delayed recovery, and relay dependence. Transmission therefore becomes a distinct operational segment in which continuity and diagnosability are directly tested. For this reason, Table 2 assigns the transmission stage the added meanings of recoverable transfer and segment-aware discrepancy visibility, linking it to *R1 Continuity*, *R3 Diagnosability*, *P1 source-near continuity*, *P4 segment-based observability*, *I1 Durable Capture*, and *I4 Segment-Localizable Failure*.

Taken together, Table 1 translates constrained SME conditions into operational design logic, while Table 2 shows how that logic refines the interpretation of the general data pipeline lifecycle under real manufacturing constraints. The result is a single analytical structure that connects the problem space, framework logic, and lifecycle interpretation. This synthesis also provides the basis for reading the field implementation in Section 5 and the evidence-based assessment in Section 6.

5. Field Implementation

This section describes how the proposed framework was instantiated in a real SME manufacturing setting. It first introduces the deployment setting, then explains the field-side acquisition and edge architecture, the cloud-side processing path, and the resulting operational use.

5.1. Deployment Setting

The empirical setting of this study was the Idang Food yakgwa plant, one of several OEM factories managed by ONGOING, an F&B company operating across multiple production sites. The deployment was therefore designed not as an isolated factory pilot, but as the first field implementation of a broader architecture in which operational data from multiple factories could be collected locally and managed through a shared cloud-side structure. In this sense, Idang Food was selected not only as a practical implementation site, but also as a representative case through which this study examines how an operational data foundation can be established under constrained SME manufacturing conditions.

The plant's production flow included raw-material receiving, weighing, mixing, forming, frying, coating, cooling, inner packaging, detection, outer packaging, and storage. Yet the main challenge at

 doi:10.20944/preprints202604.0762.v1

the site did not lie simply in the number of stages. Operationally relevant states were distributed across different rooms, devices, and tasks, while production and quality management depended heavily on manual records and worker judgment. Process conditions were not accumulated as connected records, weighing-related checks remained strongly operator-dependent, and environmental histories across process and storage zones were difficult to preserve and compare over time. As a result, when abnormal conditions or quality issues emerged, the plant did not lack observations in an absolute sense; rather, it lacked continuous and reusable operational records that could support traceability and later interpretation.

This made the site suitable for the present study. Idang Food reflected several conditions that commonly characterize constrained SME factories: heterogeneous and partly non-networked sources, fragmented process visibility, limited internal connectivity, and restricted operational slack for day-to-day technical management. Under such conditions, an operational data foundation could not be assumed to arise simply from downstream software or cloud connectivity. It first had to be built from the field upward by making process states recordable, collectable, and sustainable as data under actual production constraints.

The field deployment took the form of an actual on-site installation and system realization rather than a conceptual design exercise. The study did not begin with an already prepared digital source environment; instead, it proceeded by physically establishing one inside the factory through sensor placement, gateway configuration, and edge-side deployment. Figure 5 presents this implementation context by showing the production environment together with representative sensing and edge elements installed in the field. This point is important because the case demonstrates not only that an operational data foundation was conceptually proposed, but also that its source-side basis was concretely built and operated in a constrained SME factory. Through that installation, the plant acquired the initial operational foothold required for continuity, traceability, and subsequent cloud-side integration.

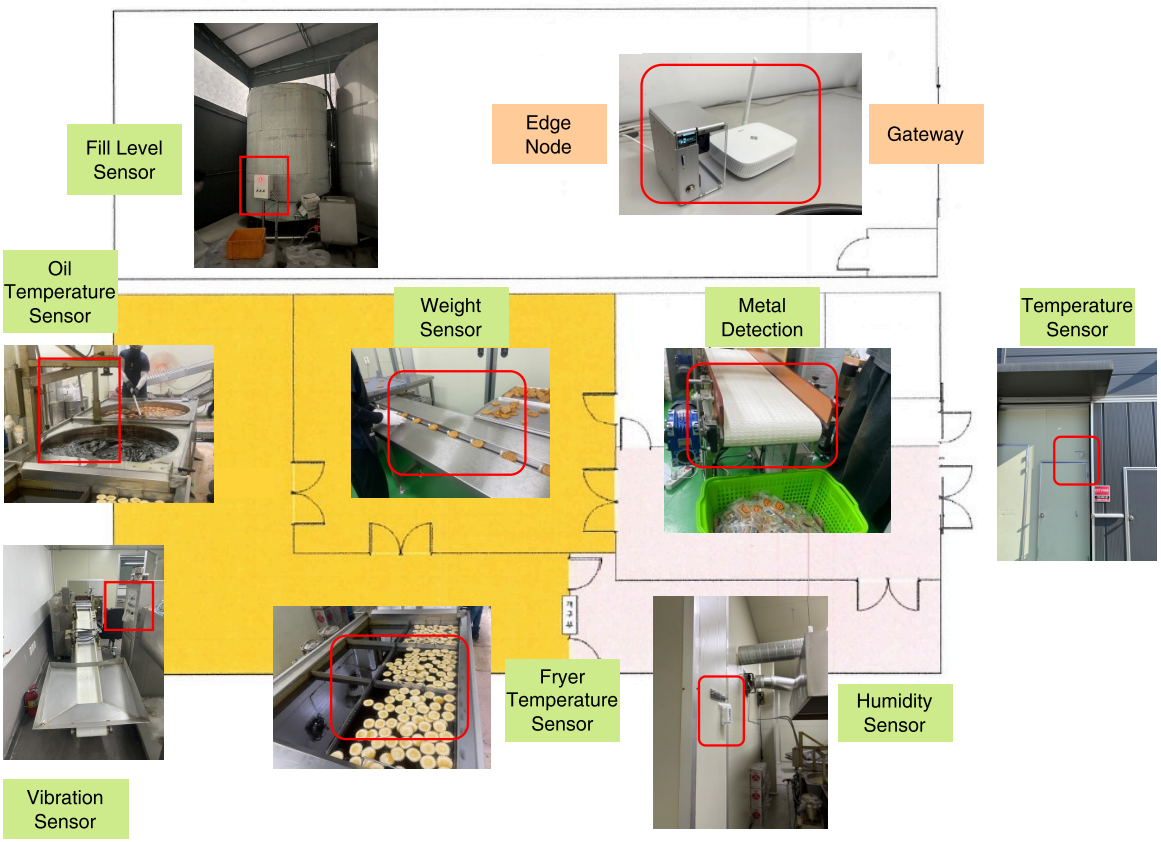

Figure 5. Factory layout with representative sensing and edge deployment.

 doi:10.20944/preprints202604.0762.v1

Accordingly, the role of Idang Food in this study was twofold. At the factory level, it provided a realistic constrained environment in which fragmented sources, manual practices, and brownfield conditions had to be handled in practice. At the system level, it served as the first implementation site of an architecture intended to support factory-to-cloud data flow and centralized multi-factory management. For this reason, the next subsection begins with field data acquisition and the edge architecture, because the first requirement of the operational data foundation was to explain how field data were physically captured and operationally secured at the factory before being forwarded into the broader cloud-side system.

5.2. Field Data Acquisition and Edge Architecture

The factory-side acquisition layer was implemented as an edge-centered intake stack that received field data within the plant, preserved them locally, and then forwarded them to the cloud event pipeline. Because the plant did not provide a uniform network-native source environment, and because stable plant-wide connectivity could not be assumed across distributed process areas, the architecture was organized around a local edge boundary rather than direct source-to-cloud transmission.

LoRaWAN, a low-power wide-area communication approach well suited to distributed sensing, was selected as the field communication network for this deployment. This choice reflected the actual plant environment: the production areas did not provide wired LAN coverage, and stable WiFi-based collection could not be assumed across the factory. Under these conditions, the field network could not be designed on the assumption of always-on internal Internet connectivity. LoRaWAN therefore provided a practical means of collecting distributed field events without extensive wiring while keeping the gateway–edge path locally controlled within the factory.

As shown in Figure 6, the implemented architecture was organized into three zones: the in-factory field layer on the left, the internal edge stack at the center, and the cloud broker boundary on the right. Field events were transmitted over LoRa to the gateway, handled by ChirpStack Gateway Bridge and ChirpStack Server, relayed through an MQTT broker, and then accepted by edge-side services. On the edge node, the MQTT logger and source-specific collectors wrote accepted records into local SQLite stores before any cloud delivery was assumed. This made the acquisition path explicit—field transmission, gateway reception, LoRaWAN handling, MQTT relay, edge logging, and upper-layer forwarding were treated as separate steps—and established the edge node as the first stable intake and persistence boundary of the system.

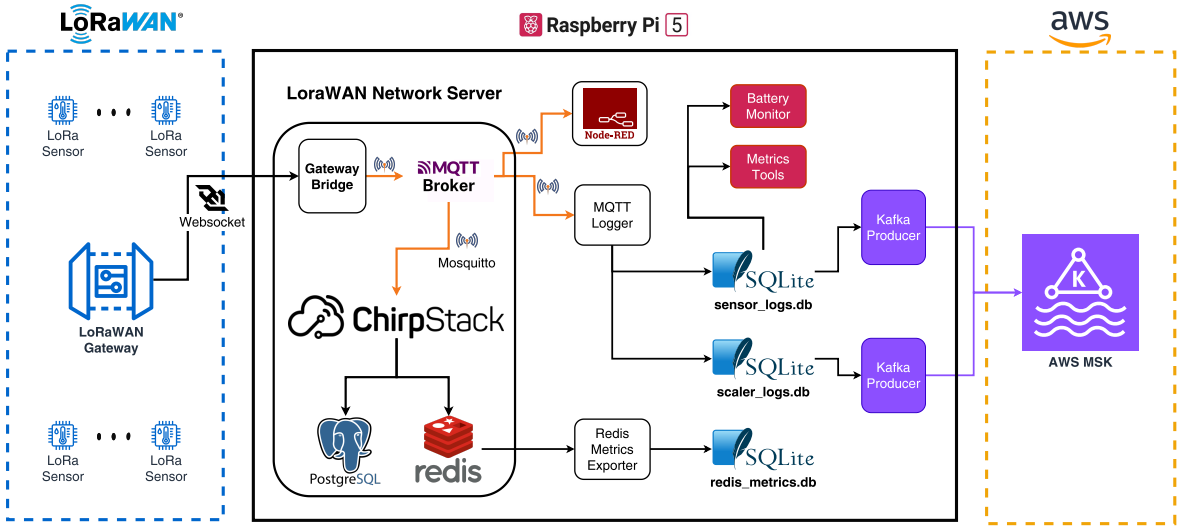

Figure 6. Edge architecture for field data acquisition.

The edge stack also accommodated heterogeneous source conditions, rather than only native wireless sensor uplinks. A representative example was the production electronic scale, which did

 doi:10.20944/preprints202604.0762.v1

not provide native network communication and exposed only a printer port as its available output interface. To incorporate the device without replacement, that port was connected to the Raspberry Pi through an RS232 serial link. A dedicated daemon collector parsed the incoming weighing values, stored them locally, and then transmitted them through a LoRa device attached to the Raspberry Pi so that they could enter the same acquisition path as other field events. In practical terms, this flow can be summarized as $Scale \rightarrow RS232 \rightarrow RPi \rightarrow LoRa$. This shows that the edge architecture could absorb non-networked legacy equipment as well as standard wireless sensor uplinks within the same factory-side event flow.

Once accepted and retained at the edge, the records were forwarded by producer services to the cloud broker layer. In the broader architecture, this upstream path was organized around Kafka-based streaming, including AWS MSK on the cloud side, so that raw events from each factory could be published into a shared event pipeline for subsequent parsing, normalization, and persistence. The next subsection explains this cloud-side processing path.

5.3. Cloud Processing and Event Flow

On top of the field acquisition layer, the system implemented a cloud-side processing path that received raw events from factory edges, routed them through a brokered stream architecture, and transformed them into persistent and service-ready records. Multiple factory-originated event streams were forwarded to the AWS cloud and published into Kafka topics in the MSK cluster, where subsequent stages handled normalization, derivation, persistence, and operational consumption. Figure 7 presents this deployment-level structure by showing how data from multiple factories converged within a shared AWS-based cloud environment and entered a common brokered processing path. In this way, the cloud layer served not as a single application endpoint, but as a staged processing structure connecting multi-factory intake to downstream use.

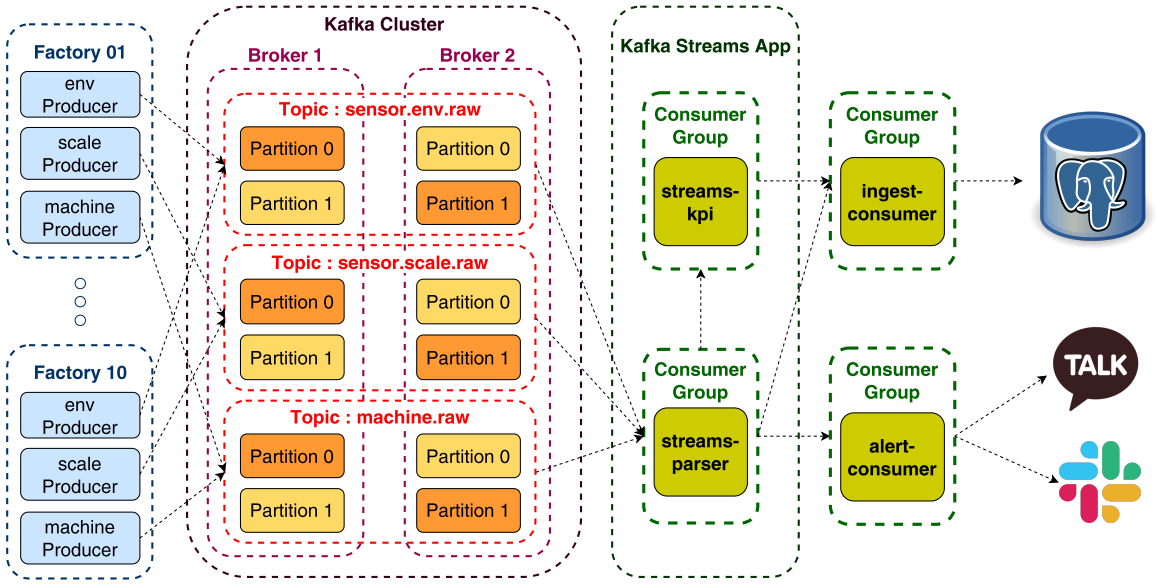

Figure 7. Cloud-side processing architecture.

Within this cloud path, raw factory events first entered broker topics such as `sensor.env.raw`, `sensor.scale.raw`, and `machine.raw`. These raw topics were consumed by the parser stream, which decoded heterogeneous payloads, mapped sources, unified metrics, normalized timestamps, applied deduplication, and produced canonical events for downstream use. On top of this normalized stream, KPI processing and alert-side consumption proceeded as separate downstream paths: KPI streams generated derived indicators such as production quantity and yield, while alert consumers used the same governed event basis to detect conditions requiring notification or intervention. Finally, ingest consumers persisted the normalized and derived outputs to the database layer for later dashboard and

 doi:10.20944/preprints202604.0762.v1

analytical use. In this way, the cloud path functioned as a staged transformation flow through which heterogeneous raw messages were converted into governed and reusable operational records.

A central design objective of this layer was functional decoupling. Factory-side producers were able to publish raw events to the broker without depending directly on parser execution, KPI computation, alert handling, or database ingestion. This separation reduced failure propagation across stages and made the system easier to modify as rules, sources, and downstream requirements evolved. In practical terms, the broker layer provided a shared event boundary through which intake, interpretation, derivation, persistence, and service consumption could remain structurally separated.

The parser stream concentrated the semantic normalization responsibility of the cloud path. Rather than allowing later components to interpret device-specific payload structures independently, the system converted heterogeneous source messages into a governed common form at this stage. In this implementation, `source_id` functioned as the canonical identifier of each operational source by organizing identity around controlled attributes such as tenant, line, process, device type, and metric, while timestamp semantics were stabilized by distinguishing `event_time`, `edge_ingest_time`, and `db_ingest_time` as separate time axes with different operational roles. This prevented ambiguity in ordering and interpretation and kept downstream processing aligned with a consistent semantic discipline.

After canonicalization, KPI computation was intentionally performed in the stream layer rather than deferred to dashboard query time. If production quantity, yield, and related indicators had been computed repeatedly from raw or lightly processed records at query time, each view request would have required additional aggregation and business-rule evaluation over a larger event set. By computing these indicators continuously from the normalized stream and persisting the resulting outputs as ready-to-use records, the system shifted repeated query-time work into incremental event-time processing. This reduced dashboard-side computational burden, simplified query logic, and supported more stable low-latency operational views. At the same time, alert consumers could react to the same governed event basis without overloading either the parser or the persistence layer.

Persistent storage was handled through ingest consumers. These consumers received already normalized or derived events and wrote them into the database without reintroducing business logic at the persistence boundary. As a result, semantic interpretation remained in the parser layer, derived computation remained in the KPI stream, alert-side reaction remained in dedicated consumers, and the storage path remained focused on durable append-oriented persistence. This separation of responsibilities made the cloud-side stack easier to reason about and more robust to later changes in processing rules or downstream interfaces.

Taken together, the cloud-side processing path completed the second half of the operational data foundation. If the field edge established source-near acceptance and local durability, the cloud layer provided brokered multi-factory intake, governed normalization, separated derivation and alert handling, and service-ready persistence. The result was a processing architecture that did not expose raw heterogeneous payloads directly to downstream services, but instead converted them into controlled operational records that could support dashboards, alerts, traceability, and future analytical extension.

5.4. Operational Use

The implementation extended beyond data capture and cloud-side processing into an operational layer through which plant personnel could observe conditions and respond when necessary. This layer mattered because the system was not intended to remain merely a background pipeline. Instead, it was designed to make governed operational records directly usable in day-to-day manufacturing practice and thereby connect the data foundation to actual plant awareness and intervention.

Dashboards translated processed records into operational views suitable for both operators and managers. Rather than exposing raw signals alone, the system presented current conditions, recent trends, source health, packet reception quality, and derived indicators in forms that supported routine monitoring. This changed the practical mode of work at the plant. Before digitalization, condition

 doi:10.20944/preprints202604.0762.v1

checking depended heavily on manual records, fragmented local observation, and worker judgment. With the implemented layer, plant personnel could instead rely on a shared and continuously updated reference for process and environmental conditions, reducing exclusive dependence on tacit judgment and making operational awareness more consistent and reviewable.

Alerting formed the intervention path of this layer. When operationally meaningful conditions were detected—for example, fryer oil temperature departing from its expected range, storage-area temperature rising sharply during extreme summer heat, prolonged silence from a source, or battery-related degradation affecting sensor reliability—the system generated asynchronous alerts so that personnel could respond in time. In this design, alerts were not merely an accessory to dashboards, but the mechanism through which monitored states became actionable events. The resulting loop can be understood as observe–detect–notify–act.

The implemented layer also reached the point of machine-side intervention capability. The system was built to support actual device power control in the field, showing that the operational data foundation did not remain limited to passive observation and notification. Although this study does not center on closed-loop control automation itself, this capability is important because it shows that the same governed data basis could already support downstream action as well as monitoring. Taken together, the operational layer demonstrates that the implemented system functioned not merely as a data pipeline, but as a practical interface connecting visibility, alerting, human response, and intervention on a common operational data foundation.

6. Evaluation of the Framework

This section evaluates whether the proposed framework functioned as an operational data foundation in a real manufacturing setting. Rather than emphasizing benchmark performance or algorithmic accuracy, the evaluation examines how the six framework requirements became visible in the implemented system and its operational history. Accordingly, Section 6.1 reviews requirement-wise evidence, and Section 6.2 presents a representative field case that illustrates more concretely how the framework became visible in actual operation.

6.1. Framework Compliance

This subsection examines framework compliance across the six core requirements defined in Section 4. Rather than reducing each requirement to a single metric, it considers multiple forms of representative evidence, summarizes them in Table 3, and interprets their operational meaning.

Table 3. Requirement-wise compliance snapshot.

Requirement	Representative evidence
Continuity	local persistence, recovery path, gap reduction
Governance	source identity, canonical semantics, time separation
Diagnosability	segment-wise visibility, stage discrepancy checks
Operability	modular stack, monitoring and intervention points
Reprocessability	raw-like retention, layered persistence
Evolvability	common event contract, extensible onboarding

Continuity was assessed not in terms of perfect packet arrival, but in terms of whether the operational record remained continuous and recoverable under unstable field conditions. In the implemented system, gateway reception, edge-side logging, downstream ingest, and database persistence were observable as separate stages, allowing continuity to be examined through persisted-to-expected ratios, prolonged no-data intervals, and before–after recovery patterns. Under an expected daily volume of about 48,000 events, early persistence remained near 15,870 events, and some source groups showed repeated no-data gaps of 2–5 hours. After communication-related adjustments, persisted events recovered to about 45,600 per day, while extended-gap cases decreased from roughly 18 per

 doi:10.20944/preprints202604.0762.v1

day to fewer than two. These observations show that continuity was realized as recoverable record preservation despite intermediate instability, rather than as uninterrupted delivery.

Governance was evaluated in terms of whether heterogeneous field events were brought under a stable semantic discipline across collection, normalization, and storage. In the implemented system, source identity was governed through `source_id`, while the parser stage fixed metric definitions, timestamp interpretation, and normalization rules before downstream use. The distinction among `event_time`, `edge_ingest_time`, and `db_ingest_time` prevented ambiguity between occurrence time, pipeline reference time, and persistence time. In addition, the separation of raw-like event storage, source metadata, and derived KPI layers preserved a clear boundary between operational records and business-level abstraction. Taken together, these structures show that governance was achieved through controlled meaning, traceable identity, and stable interpretation across the pipeline.

Diagnosability was assessed in terms of whether degradation remained a black-box symptom or became visible as a discrepancy localizable to specific lifecycle segments. Because gateway-visible counts, local edge logs, parser and consumer stages, and database persistence counts could be compared, the same symptom of "too little data" could be interpreted as different failure modes depending on where the discrepancy first appeared. In some periods, gateway-visible counts were already far below expectation, directing attention to upstream collection and communication. In other cases, the divergence widened only after gateway reception, indicating the need to inspect downstream processing or storage. This segment-wise visibility was especially important in the LoRa case discussed in Section 6.2, where the main source of degradation could be narrowed to the communication and configuration layer rather than misread as a parser or database issue.

Operability was reflected not simply in whether the system could be made to run, but in whether it could be deployed, inspected, adjusted, and maintained under limited staffing and maintenance capacity. The implemented structure separated field-side acquisition, gateway and network services, edge persistence, cloud-side stream processing, database storage, and dashboard and alert interfaces by function, thereby reducing unnecessary coupling among acquisition, normalization, persistence, derivation, and presentation. In practice, interventions such as rejoin handling, communication-parameter adjustment, logging verification, interval tuning, packet comparison, dashboard inspection, and alert-based response could be carried out at the relevant layer without redesigning the system as a whole. This indicates that the foundation remained operationally manageable as a field system rather than merely as a one-time prototype.

Reprocessability was assessed in terms of whether collected records remained available for later reinterpretation, recalculation, correction, and recovery, rather than being consumed only by the current dashboard or rule logic. The implemented system avoided transformed-only persistence by separating raw-like event storage, source metadata, and derived KPI layers, so that later parser revision or KPI redefinition did not depend on reconstructing already abstracted outputs. Historical data therefore remained available for rule revision, retrospective audit, recomputation, and reconstruction of prior operational states. More importantly, the retained records were preserved in a structure that supported replay-oriented reinterpretation, which is the practical core of reprocessability in a changing manufacturing environment.

Evolvability was evaluated in terms of whether the structure could absorb new source types, process-specific measurements, monitoring functions, KPI logic, and downstream services without breaking its governance discipline. This property was reflected not only in the architecture itself, but also in the system's actual expansion history. The implemented foundation accommodated heterogeneous environmental sensors, a retrofitted legacy scale, cloud-side stream processing, derived KPI generation, and dashboard and alert functions within a common event flow governed by shared identity and normalization rules. Because the structure was organized around an explicit event contract, parser-stage normalization, layered persistence, and modular separation of responsibilities, extensions could be handled mainly through onboarding rules, parser updates, and additional downstream

 doi:10.20944/preprints202604.0762.v1

consumers rather than through full pipeline redesign. This indicates that the framework supported staged expansion while preserving semantic and operational coherence.

Taken together, the evidence above indicates that the proposed framework was realized not only as a conceptual structure, but also as an operational one in the field. To make this more concrete, the next subsection examines a representative field case through which the framework's practical relevance can be seen in actual operation.

6.2. Operational Case: LoRa Reliability Improvement

A field data foundation does not remain operationally reliable simply because it has once been installed. In practice, reception quality, device condition, battery state, wireless configuration, and usage patterns continue to change over time. For this reason, the value of the proposed framework lies not in whether it delivered perfect behavior from the outset, but in whether emerging failures could be detected, localized, and corrected in an operationally meaningful way. This subsection presents a representative field case in which severe degradation in periodic uplink reception was diagnosed and improved within the implemented foundation.

During the early operational period, plant-wide periodic collection failed to sustain its expected level. Reception across multiple monitored sensor sets dropped to roughly 33%, daily persisted events were 15,870 out of an expected 48,000, and repeated no-data intervals of 2–5 hours appeared across several monitored periods. This was not merely a matter of reduced communication quality. It directly weakened line-level visibility, reduced confidence in dashboard and alert outputs, and undermined the continuity of the operational record on which later interpretation depended.

Because the implemented system did not rely solely on final database status, the degradation could be interpreted as a segment-wise discrepancy rather than as a vague end-to-end symptom. Expected transmissions, gateway-visible events, edge or local logs, ingest-stage counts, and database persistence counts could be examined together. Figure 8 summarizes this diagnostic logic as a sequence of adjacent-stage discrepancy checks, in which the first substantial gap indicates the most likely failure boundary. As shown by the actual segment-wise counts in Table 4, the main loss had already occurred before parsing and database persistence: in a representative 24-hour window, only about 16,400 events were visible at the gateway, whereas downstream retention from gateway-visible events to database persistence remained consistently high at roughly 98–99%. This observed pattern indicated that the dominant loss was occurring before the parser and database layer, making the collection and communication segment the primary locus of investigation.

The subsequent diagnosis localized the main bottleneck not to parser failure or database loss, but to the LoRa communication and configuration layer. The problem was not simply a general reduction in signal strength, but a network condition in which frequent periodic confirmed uplinks depended on downlink acknowledgements under limited active-channel availability. When acknowledgements were delayed or missed, retransmissions accumulated, channel contention increased, and reception became uneven across devices. In addition, stale device participation state in part of the fleet required rejoin recovery before stable collection could resume. The diagnosed issue was therefore a concrete reception bottleneck in the LoRa network path itself: ACK-dependent uplink behavior, limited effective channel use, and device-state instability were jointly degrading collection before downstream processing began.

Once the likely bottleneck had been localized, the response proceeded through targeted intervention rather than unguided trial and error. The main adjustments included reducing confirmed-mode dependence, expanding active channels, performing device rejoin, tuning transmission-related parameters, and refining monitoring intervals. After these adjustments, collection stability improved markedly. Packet reception increased from approximately 33% to 95%, daily persisted events rose from 15,870 to 45,600, extended no-data intervals decreased from about 18 cases per day to fewer than two, and average gap duration dropped from roughly 145 minutes to 18 minutes. The before–after time-series plots also show that the dense missing segments visible before adjustment were substantially reduced afterward, while observation regularity and line-to-line consistency improved (Table 5, Figure 9).

 doi:10.20944/preprints202604.0762.v1

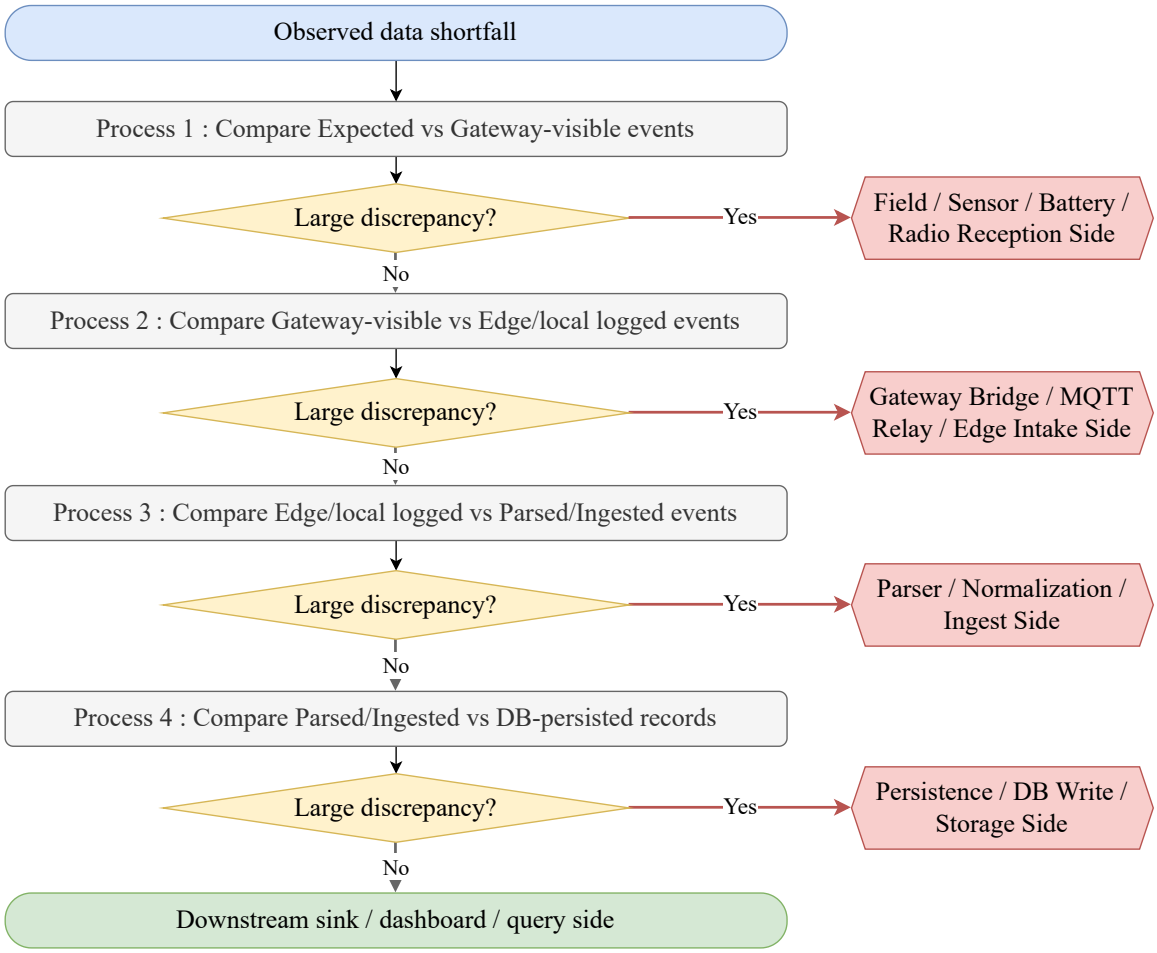

Figure 8. Segment-wise failure localization through adjacent-stage discrepancy checks.

Table 4. Event counts across pipeline segments during reception degradation.

Stage / Metric	Expected	Observed	Ratio (%)
Expected transmissions	48,000	–	–
Gateway-visible events	48,000	16,400	34.2
Edge/local logged events	16,400	16,120	98.3
Parsed/Ingested events	16,120	15,980	99.1
DB-persisted events	15,980	15,870	99.3

Table 5. Outcome of reception reliability improvement.

Metric	Before	After	Change
Reception rate (%)	33	95	+62 pp
Persisted events/day	15,870	45,600	+187%
Average gap duration (min)	145	18	–87.6%
Long-gap events/day	18	2	–88.9%
Gateway-to-DB consistency (%)	96.8	99.4	+2.6 pp

The significance of this case extends beyond successful LoRa tuning. First, it shows that *R1 Continuity* was assessable in operational terms and that *I1 Durable Capture* could be examined through the degradation and recovery of the persisted record under severe field-side instability. Second, it shows that *R3 Diagnosability* and *I4 Segment-Localizable Failure* were realized through the comparison of expected transmissions, gateway-visible events, edge logs, ingest-stage records, and database persistence, which made the dominant discrepancy visible in the communication segment rather

 doi:10.20944/preprints202604.0762.v1

than only at the final database boundary. What this case demonstrates, therefore, is not simply that reception improved after adjustment, but that the implemented foundation made it possible to observe the failure as a structured operational discrepancy, to narrow it to a specific lifecycle segment, and to evaluate recovery within the same evidence framework. In that sense, the case provides a concrete field illustration of the framework's role in diagnosis and corrective improvement under constrained SME conditions.

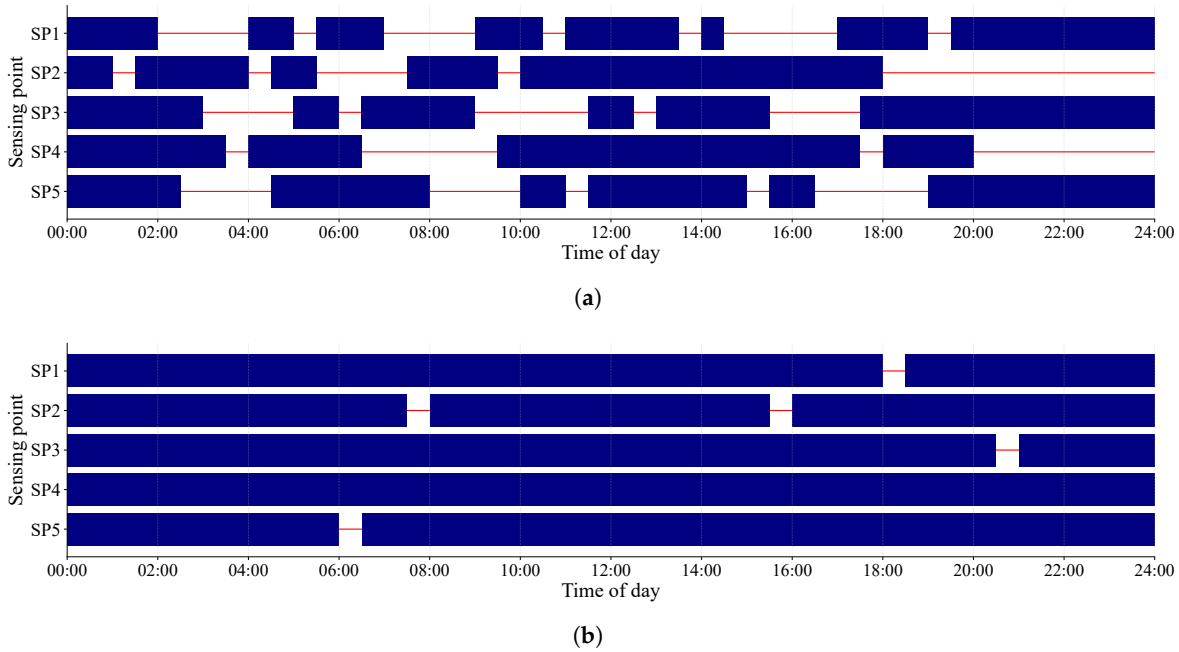

Figure 9. Reception stability across sensing points over time. (**a**) Before adjustment. (**b**) After adjustment.

7. Conclusions

This study addressed the practical bottleneck of smart manufacturing in constrained SME factories as an operational data foundation problem. To address this issue, it proposed the *Operational Data Foundation Framework*, organized around six core requirements—continuity, governance, diagnosability, operability, reprocessability, and evolvability—together with their corresponding design principles and operational invariants. The framework was examined not only conceptually, but also through implementation and operation in a real SME food-manufacturing factory. The evaluation showed that the implemented system preserved controlled semantics, supported segment-wise observability, maintained replayable records, and remained manageable under constrained field conditions. The LoRa reception-instability case further demonstrated the framework's practical value by showing that degradation could be detected as a structured lifecycle discrepancy, localized to a specific segment, and improved through targeted intervention.

The proposed framework therefore offers a reference design logic for making smart manufacturing more continuous and practically operable in constrained SME settings, while also providing a basis for future extension toward more human-centric and sustainability-oriented manufacturing. This study nevertheless has several limitations. The framework was examined through a single SME food-manufacturing deployment, and its broader generalizability across multiple factories and more diverse industrial settings remains to be tested. In addition, although the implementation supported operational visibility, alerting, and machine-side intervention, the study did not directly evaluate large-scale multi-site deployment, fully automated replay workflows, or closed-loop control integration. Future work may therefore extend the framework through multi-factory replication, richer interoperability settings, automated replay-oriented pipelines, and tighter integration with analytics, digital twins, and CPPS-oriented control.

 doi:10.20944/preprints202604.0762.v1

Author Contributions: Conceptualization, Y.K. and D.P.; methodology, Y.K.; software, Y.K.; validation, Y.K.,; formal analysis, Y.K.; investigation, Y.K.; resources, Y.K.; data curation, Y.K.; writing—original draft preparation, Y.K.; writing—review and editing, D.P.; visualization, Y.K.; supervision, D.P.; project administration, D.P.; funding acquisition, D.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research was partly supported by the MSIT(Ministry of Science and ICT), Korea, under the Convergence security core talent training business support program(IITP-2026-RS-2023-00266605, 50%), IITP(Institute of Information & Communications Technology Planning & Evaluation)-ITRC(Information Technology Research Center) grant funded by the Korea government(MSIT) (IITP-2026-RS-2024-00438056, 50%).

Data Availability Statement: Restrictions apply to the availability of these data. The data used in this study are available from the corresponding author upon reasonable request, subject to the permission of the participating enterprise.

Acknowledgments: The authors thank ONGOING and Idang Food for their cooperation and support in the field implementation. The authors especially thank Kihoon Lee, ONGOING production team manager, for his on-site support and cooperation, and also thank Jisu Kwon, Seungjae Yoo, and Jonghoon Hwang for their assistance with the field implementation.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Song, Y. A Comprehensive Review of Key Technologies and Applications in Smart Manufacturing Systems: From Digital Foundations to Intelligent Applications. In Proceedings of the 2025 2nd International Conference on Industrial Automation and Robotics, New York, NY, USA, 2025; IAR '25, pp. 328–333. https://doi.org/10.1145/3778886.3778938.
2. Tao, F.; Qi, Q.; Liu, A.; Kusiak, A. Data-driven smart manufacturing. *J. Manuf. Syst.* **2018**, *48*, 157–169. https://doi.org/10.1016/j.jmsy.2018.01.006.
3. Tao, F.; Zhang, M. Digital Twin Shop-Floor: A New Shop-Floor Paradigm Towards Smart Manufacturing. *IEEE Access* **2017**, *5*, 20418–20427. https://doi.org/10.1109/ACCESS.2017.2756069.
4. Ding, K.; Chan, F.T.S.; Zhang, X.; Zhou, G.; Zhang, F. Defining a Digital Twin-based Cyber-Physical Production System for autonomous manufacturing in smart shop floors. *Int. J. Prod. Res.* **2019**, *57*, 6315–6334. https://doi.org/10.1080/00207543.2019.1566661.
5. Nguyen, P.; Kim, M.; Nichols, E.; Yoon, H.S. AI-Driven Digital Twins for Manufacturing: A Review Across Hierarchical Manufacturing System Levels. *Sensors* **2025**, *26*, 124. https://doi.org/10.3390/s26010124.
6. Cerquitelli, T.; Pagliari, D.J.; Calimera, A.; Bottaccioli, L.; Patti, E.; Acquaviva, A.; Poncino, M. Manufacturing as a Data-Driven Practice: Methodologies, Technologies, and Tools. *Proc. IEEE* **2021**, *109*, 399–422. https://doi.org/10.1109/JPROC.2021.3056006.
7. Ren, S.; Zhang, Y.; Liu, Y.; Sakao, T.; Huisingh, D.; Almeida, C.M.V.B. A comprehensive review of big data analytics throughout product lifecycle to support sustainable smart manufacturing: A framework, challenges and future research directions. *J. Clean. Prod.* **2019**, *210*, 1343–1365. https://doi.org/10.1016/j.jclepro.2018.11.025.
8. Bernstein, W.Z.; Hedberg Jr., T.D.; Helu, M.; Feeney, A.B. Contextualising manufacturing data for lifecycle decision-making. *Int. J. Prod. Lifecycle Manag.* **2018**, *10*, 326–347. https://doi.org/10.1504/IJPLM.2017.090328.
9. Walton, R.B.; Ciarallo, F.W.; Champagne, L.E. A Unified Digital Twin Approach Incorporating Virtual, Physical, and Prescriptive Analytical Components to Support Adaptive Real-Time Decision-Making. *Computers & Industrial Engineering* **2024**, *193*, 110241. https://doi.org/10.1016/j.cie.2024.110241.
10. Zaborowski, P.; Bye, B.L.; Berre, A.J.; Atkinson, R.; Villar, A.; Voidrot, M.F.; Palma, R. The Role of Standards in The Environmental Digital Twins Architectures. In Proceedings of the IGARSS 2024 – 2024 IEEE International Geoscience and Remote Sensing Symposium, 2024, pp. 271–273. https://doi.org/10.1109/IGARSS53475.2024.10640598.
11. Ismail, A.; Truong, H.L.; Kastner, W. Manufacturing process data analysis pipelines: a requirements analysis and survey. *J. Big Data* **2019**, *6*, 1. https://doi.org/10.1186/s40537-018-0162-3.
12. Raptis, T.P.; Passarella, A.; Conti, M. Data Management in Industry 4.0: State of the Art and Open Challenges. *IEEE Access* **2019**, *7*, 97052–97093. https://doi.org/10.1109/ACCESS.2019.2929296.

 doi:10.20944/preprints202604.0762.v1

13. Hedberg Jr., T.D.; Feeney, A.B.; Helu, M.; Camelio, J.A. Toward a Lifecycle Information Framework and Technology in Manufacturing. *J. Comput. Inf. Sci. Eng.* **2017**, *17*, 021010. https://doi.org/10.1115/1.4034132.
14. Zeid, A.; Sundaram, S.; Moghaddam, M.; Kamarthi, S.; Marion, T. Interoperability in Smart Manufacturing: Research Challenges. *Machines* **2019**, *7*, 21. https://doi.org/10.3390/machines7020021.
15. Pereira, R.M.; Szejka, A.L.; Canciglieri Junior, O. Towards an information semantic interoperability in smart manufacturing systems: contributions, limitations and applications. *Int. J. Comput. Integr. Manuf.* **2021**, *34*, 422–439. https://doi.org/10.1080/0951192X.2021.1891571.
16. Morris, K.C.; Lu, Y.; Frechette, S. Foundations of Information Governance for Smart Manufacturing. *Smart Sustain. Manuf. Syst.* **2020**, *4*, 43–61. https://doi.org/10.1520/SSMS20190041.
17. Foidl, H.; Golendukhina, V.; Ramler, R.; Felderer, M. Data pipeline quality: Influencing factors, root causes of data-related issues, and processing problem areas for developers. *J. Syst. Softw.* **2024**, *207*, 111855. https://doi.org/10.1016/j.jss.2023.111855.
18. Davis, J.; Malkani, H.; Dyck, J.; Korambath, P.; Wise, J. Cyberinfrastructure for the Democratization of Smart Manufacturing. In *Smart Manufacturing*; Soroush, M.; Baldea, M.; Edgar, T.F., Eds.; Elsevier, 2020; pp. 83–116. https://doi.org/10.1016/B978-0-12-820027-8.00004-6.
19. Mittal, S.; Khan, M.A.; Romero, D.; Wuest, T. A critical review of smart manufacturing & Industry 4.0 maturity models: Implications for small and medium-sized enterprises (SMEs). *J. Manuf. Syst.* **2018**, *49*, 194–214. https://doi.org/10.1016/j.jmsy.2018.10.005.
20. Krishnan, R. Challenges and benefits for small and medium enterprises in the transformation to smart manufacturing: a systematic literature review and framework. *J. Manuf. Technol. Manag.* **2024**, *35*, 918–938. https://doi.org/10.1108/JMTM-07-2022-0255.
21. Mittal, S.; Khan, M.A.; Purohit, J.K.; Menon, K.; Romero, D.; Wuest, T. A smart manufacturing adoption framework for SMEs. *Int. J. Prod. Res.* **2020**, *58*, 1555–1573. https://doi.org/10.1080/00207543.2019.1661540.
22. Ghobakhloo, M.; Ching, N.T. Adoption of digital technologies of smart manufacturing in SMEs. *J. Ind. Inf. Integr.* **2019**, *16*, 100107. https://doi.org/10.1016/j.jii.2019.100107.
23. Alqoud, A.; Schaefer, D.; Milisavljevic-Syed, J. Industry 4.0: a systematic review of legacy manufacturing system digital retrofitting. *Manufacturing Rev.* **2022**, *9*, 32. https://doi.org/10.1007/978-3-030-57997-5_13.
24. Huang, D.; Chin, C.P.Y. The Barriers and Challenges in Smart Manufacturing Adoption for SMEs: A Review. *J. Adv. Manuf. Syst.* **2025**, *24*, 533–557. https://doi.org/10.1142/S0219686725500246.
25. Kumar, R.; Dutta, G.; Phanden, R.K. Digitalization Adoption Barriers in the Context of Sustainability and Operational Excellence: Implications for SMEs. *Eng. Manag. J.* **2025**, *37*, 355–371. https://doi.org/10.1080/10429247.2024.2372519.
26. Narwane, V.S.; Raut, R.D.; Gardas, B.B.; Narkhede, B.E.; Awasthi, A. Examining smart manufacturing challenges in the context of micro, small and medium enterprises. *Int. J. Comput. Integr. Manuf.* **2022**, *35*, 1395–1412. https://doi.org/10.1080/0951192X.2022.2078508.
27. Abdel-Aty, T.A.; Negri, E. Conceptualizing the digital thread for smart manufacturing: a systematic literature review. *J. Intell. Manuf.* **2024**, *35*, 3629–3653. https://doi.org/10.1007/s10845-024-02407-1.
28. Zhang, Q.; Liu, J.; Chen, X. A Literature Review of the Digital Thread: Definition, Key Technologies, and Applications. *Systems* **2024**, *12*, 70. https://doi.org/10.3390/systems12030070.
29. Rozhok, A.; Abate, R.; Manoli, E.; Nele, L. A Review of Recent Advanced Applications in Smart Manufacturing Systems. *J. Manuf. Mater. Process.* **2026**, *10*, 1. https://doi.org/10.3390/jmmp10010001.
30. Bueno, A.; Godinho Filho, M.; Frank, A.G. Smart production planning and control in the Industry 4.0 context: A systematic literature review. *Comput. Ind. Eng.* **2020**, *149*, 106774. https://doi.org/10.1016/j.cie.2020.106774.
31. Cinar, Z.M.; Nuhu, A.A.; Zeeshan, Q.; Korhan, O. Digital Twins for Industry 4.0: A Review. In *Industrial Engineering in the Digital Disruption Era*; Calisir, F.; Korhan, O., Eds.; Lecture Notes in Management and Industrial Engineering, Springer: Cham, 2020. https://doi.org/10.1007/978-3-030-42416-9_18.
32. Atalay, M.; Murat, U.; Oksuz, B.; Parlaktuna, A.M.; Pisirir, E.; Testik, M.C. Digital twins in manufacturing: systematic literature review for physical–digital layer categorization and future research directions. *Int. J. Comput. Integr. Manuf.* **2022**, *35*, 679–705. https://doi.org/10.1080/0951192X.2021.2022762.
33. Ojstersek, R.; Javernik, A.; Buchmeister, B. Optimizing smart manufacturing systems using digital twin. *Adv. Prod. Eng. Manag.* **2023**, *18*, 475–485. https://doi.org/10.14743/apem2023.4.486.
34. Qamsane, Y.; Chen, C.Y.; Balta, E.C.; Kao, B.C.; Mohan, S.; Moyne, J.; Tilbury, D.; Barton, K. A Unified Digital Twin Framework for Real-time Monitoring and Evaluation of Smart Manufacturing Systems. In Proceedings

 doi:10.20944/preprints202604.0762.v1

of the 2019 IEEE 15th International Conference on Automation Science and Engineering (CASE), 2019, pp. 1394–1401. https://doi.org/10.1109/COASE.2019.8843269.

35. Ciano, M.P.; Pozzi, R.; Rossi, T.; Strozzi, F. Digital twin-enabled smart industrial systems: a bibliometric review. *Int. J. Comput. Integr. Manuf.* **2021**, *34*, 690–708. https://doi.org/10.1080/0951192X.2020.1852600.
36. Saqlain, M.; Piao, M.; Shim, Y.; Lee, J.Y. Framework of an IoT-based Industrial Data Management for Smart Manufacturing. *J. Sens. Actuator Netw.* **2019**, *8*, 25. https://doi.org/10.3390/jsan8020025.
37. O'Donovan, P.; Leahy, K.; Bruton, K.; O'Sullivan, D.T.J. An industrial big data pipeline for data-driven analytics maintenance applications in large-scale smart manufacturing facilities. *J. Big Data* **2015**, *2*, 25. https://doi.org/10.1186/s40537-015-0034-z.
38. Modoni, G.E.; Doukas, M.; Terkaj, W.; Sacco, M.; Mourtzis, D. Enhancing factory data integration through the development of an ontology: from the reference models reuse to the semantic conversion of the legacy models. *Int. J. Comput. Integr. Manuf.* **2017**, *30*, 1043–1059. https://doi.org/10.1080/0951192X.2016.1268720.
39. Hildebrandt, C.; Köcher, A.; Küstner, C.; López-Enríquez, C.M.; Müller, A.W.; Caesar, B.; Gundlach, C.S.; Fay, A. Ontology Building for Cyber–Physical Systems: Application in the Manufacturing Domain. *IEEE Trans. Autom. Sci. Eng.* **2020**, *17*, 1266–1282. https://doi.org/10.1109/TASE.2020.2991777.
40. Westermann, T.; Hranisavljevic, N.; Fay, A. Accessing and Interpreting OPC UA Event Traces based on Semantic Process Descriptions. In Proceedings of the 2022 IEEE 27th International Conference on Emerging Technologies and Factory Automation (ETFA), Stuttgart, Germany, 2022; pp. 1–7. https://doi.org/10.1109/ETFA52439.2022.9921565.
41. Bianchini, D.; Fapanni, T.; Garda, M.; Leotta, F.; Mecella, M.; Rula, A.; Sardini, E. Digital Thread for Smart Products: A Survey on Technologies, Challenges, and Opportunities in Service-Oriented Supply Chains. *IEEE Access* **2024**, *12*, 125284–125305. https://doi.org/10.1109/ACCESS.2024.3454375.
42. Hedberg Jr., T.D.; Bajaj, M.; Camelio, J.A. Using Graphs to Link Data Across the Product Lifecycle for Enabling Smart Manufacturing Digital Threads. *J. Comput. Inf. Sci. Eng.* **2020**, *20*, 011011. https://doi.org/10.1115/1.4044921.
43. Kwon, S.; Monnier, L.V.; Barbau, R.; Bernstein, W.Z. Enriching standards-based digital thread by fusing as-designed and as-inspected data using knowledge graphs. *Adv. Eng. Inform.* **2020**, *46*, 101102. https://doi.org/10.1016/j.aei.2020.101102.
44. Schmidt, N.; Lueder, A. The Flow and Reuse of Data: Capabilities of AutomationML in the Production System Life Cycle. *IEEE Ind. Electron. Mag.* **2018**, *12*, 59–63. https://doi.org/10.1109/MIE.2018.2818748.
45. Monnier, L.V.; Shao, G.; Foufou, S. A Methodology for Digital Twins of Product Lifecycle Supported by Digital Thread. In Proceedings of the ASME 2022 International Mechanical Engineering Congress and Exposition, Columbus, Ohio, USA, 2022; p. V02BT02A023. https://doi.org/10.1115/IMECE2022-95182.
46. Etz, D.; Brantner, H.; Kastner, W. Smart Manufacturing Retrofit for Brownfield Systems. *Procedia Manuf.* **2020**, *42*, 327–332. https://doi.org/10.1016/j.promfg.2020.02.085.
47. Tran, T.A.; Ruppert, T.; Eigner, G.; Abonyi, J. Retrofitting-Based Development of Brownfield Industry 4.0 and Industry 5.0 Solutions. *IEEE Access* **2022**, *10*, 64348–64374. https://doi.org/10.1109/ACCESS.2022.3182491.
48. Park, H.M.; Jeon, J.W. OPC UA based Universal Edge Gateway for Legacy Equipment. In Proceedings of the 2019 IEEE 17th International Conference on Industrial Informatics (INDIN), Helsinki, Finland, 2019; pp. 1002–1007. https://doi.org/10.1109/INDIN41052.2019.8972187.
49. Hulla, M.; Herstätter, P.; Wolf, M.; Ramsauer, C. Towards digitalization in production in SMEs – A qualitative study of challenges, competencies and requirements for trainings. *Procedia CIRP* **2021**, *104*, 887–892. https://doi.org/10.1016/j.procir.2021.11.149.
50. Chavez, Z.; Baalsrud Hauge, J.; Bellgran, M. A Conceptual Model for Deploying Digitalization in SMEs Through Capability Building. In *Advances in Production Management Systems. Towards Smart and Digital Manufacturing*; Lalic, B.; Majstorovic, V.; Marjanovic, U.; von Cieminski, G.; Romero, D., Eds.; Springer: Cham, 2020; Vol. 592, *IFIP Advances in Information and Communication Technology*. https://doi.org/10.1007/978-3-030-57997-5_13.
51. Banerjee, A.; Jayaraman, P.P.; Fizza, K.; Wang, S.; Jin, J.; Ghaderi, H. Low-cost digital manufacturing solution for process manufacturing SMEs - Lesson and experiences from real-world pilot. *IET Conf. Proc.* **2024**, *2024*, 109–115. https://doi.org/10.1049/icp.2024.3494.
52. Rauch, E.; Dallasega, P.; Unterhofer, M. Requirements and Barriers for Introducing Smart Manufacturing in Small and Medium-Sized Enterprises. *IEEE Eng. Manag. Rev.* **2019**, *47*, 87–94. https://doi.org/10.1109/EMR.2019.2931564.

 doi:10.20944/preprints202604.0762.v1

53. Gao, C.; Wang, Z.; Chen, Y. On the Connectivity of Highly Dynamic Wireless Sensor Networks in Smart Factory. In Proceedings of the 2019 International Conference on Networking and Network Applications (NaNA), Daegu, Korea (South), 2019; pp. 208–212. https://doi.org/10.1109/NaNA.2019.00045.
54. Noor-A-Rahim, M.; John, J.; Firyaguna, F.; Sherazi, H.H.R.; Kushch, S.; Vijayan, A.; O'Connell, E.; Pesch, D.; O'Flynn, B.; O'Brien, W.; et al. Wireless Communications for Smart Manufacturing and Industrial IoT: Existing Technologies, 5G and Beyond. *Sensors* **2023**, *23*, 73. https://doi.org/10.3390/s23010073.
55. Hawkridge, G.; Hernandez, M.P.; de Silva, L.; Terrazas, G.; Tlegenov, Y.; McFarlane, D.; Thorne, A. Tying Together Solutions for Digital Manufacturing: Assessment of Connectivity Technologies & Approaches. In Proceedings of the 2019 24th IEEE International Conference on Emerging Technologies and Factory Automation (ETFA), Zaragoza, Spain, 2019; pp. 1383–1387. https://doi.org/10.1109/ETFA.2019.8869411.
56. Nagorny, K.; Scholze, S.; Colombo, A.W.; Oliveira, J.B. A DIN Spec 91345 RAMI 4.0 Compliant Data Pipelining Model: An Approach to Support Data Understanding and Data Acquisition in Smart Manufacturing Environments. *IEEE Access* **2020**, *8*, 223114–223129. https://doi.org/10.1109/ACCESS.2020.3045111.
57. Nasirinejad, M.; Afshari, H.; Sampalli, S. Challenges and Solutions to Adopt Smart Maintenance in SMEs: A Literature Review and Research Agenda. *IFAC-PapersOnLine* **2024**, *58*, 917–922. https://doi.org/10.1016/j.ifacol.2024.09.164.
58. Doyle, F.; Cosgrove, J. Steps towards digitization of manufacturing in an SME environment. *Procedia Manuf.* **2019**, *38*, 540–547. https://doi.org/10.1016/j.promfg.2020.01.068.
59. McFarlane, D.; Ratchev, S.; de Silva, L.; Hawkridge, G.; Schönfuß, B.; Terrazas Angulo, G. Digitalisation for SME Manufacturers: A Framework and a Low-Cost Approach. *IFAC-PapersOnLine* **2022**, *55*, 414–419. https://doi.org/10.1016/j.ifacol.2022.04.229.