UtilitasMathematica

ISSN 0315-3681 Volume 122 (2), 2025

Unifying Multiple ERP Systems: A Case Study on Data Migration and Integration

Chandra Bonthu
Director MDM, EVERSANA,USA
Email: chandrabonthu78@gmail.com
ORCID: 0009-0009-2745-281X

Abstract
This white paper will look at the merging of three mature or older ERP environments, including SAP ECC 6.0, Oracle E-Business Suite 12.2, and Microsoft Dynamics AX 2012, into an analytics-ready and governed backbone. The program established a canonical data model (Party, Location, Item, GLAccount), and created layered landing, raw, and curated areas in Snowflake using ELT via dbt and orchestration by Airflow. Kafka consumed Change data capture to provide an ongoing harmonization and idempotency upsets. Deterministic, code-set translations and code-set translations and supervised entity resolution (blocking keys, phonetic encodings, n-gram similarities and geospatial proximity)heading Data contracts, executable tests (dbt/Great Expectations), and reconciliation packs tested completeness, conformity, referential integrity, and financial invariance across environments, and RBAC/ABAC, PII masking, immutable run manifests, and segmentation of duties accounted for SOX/GDPR controls. Practiced cutover runbooks had given results of a 7.6-hour switch of production in an eight-hour service level agreement, and no severe cases. Quality after post-maturation was significantly better: mandatory-field completeness was 99.6%, field-level conformity 98.7%, and orphan rates 0.2. DC latency achieved P95 parameter goals. The PR-AUC and F1 exceeded those of a rules-plus-logistic baseline, as gradient-boosted trees and a Siamese architecture allowed matching at millisecond latency. Business outcomes were a 1.8-day time savings on month-end close, 14.6% consolidated indirect spend, 9.2% fewer stockouts, 2.7-point invoice-accuracy improvements, 6.4% faster order creation, 41% faster analytics time-to-insight, and a decrease in the deployment failure rate (7.5% to 2.1%).

Keywords;
ERP integration, Data migration, Canonical Data Model (CDM), Entity resolution, Data quality & governance.

1. Introduction

The organization had three mature Enterprise Resource Planning (ERP) estates: SAP ECC 6.0 in Finance, Sales and Distribution, and Materials Management; Oracle E-Business Suite 12.2 in Accounts Payable and Purchasing; and Microsoft Dynamics AX 2012 in manufacturing execution. Duplicated master records, different charts of accounts, and regional customizations divided reporting. The business units had independent files, customer hierarchy, price lists, and material codes, and as a result, they displayed conflicting KPIs and long closes. Non-compliance risks were identified, such as supplier banking data and tax identifiers that were not maintained consistently, exposing payment fraud and audit risk. Planners were not able to have a centralized view of inventory as well as active purchase orders across plants, which resulted in expedites. The expense of bespoke extracts increased due to the development of shadow systems. Leadership defined the need to unify and provide a single canonical data model, a governed master-data backbone, and an integrated reporting layer that allows more rapid closes, analytics, and secure decommissioning of legacy environments without impacting the order-to-cash or procure-to-pay processes.

The program has quantifiable goals to limit scope and demonstrate value. Master-data match accuracy was required to be 98% plus or minus 0.5% false-merge rate and field-level alignment to a canonical model of 95% or higher when considering customers, vendors, items, and

customers across the general ledger accounts. Viewable pre-validated snapshots and a reversible runbook helped to limit cutover downtime to eight hours. No priority-one defects were processed under hypercare; the mean time to recovery of the incidents was less than four hours. Business KPIs were a decreased month-end close, a 5-per-cent improvement in on-time performance due to system-wide ATP, and a 10-per-cent consolidation of indirect spend through the rationalization of vendors. KPIs used to monitor CDC latency of less than five minutes P95, schema-drift incidents at zero during freeze windows, and data-quality test pass rates of at least 98%. Finance, Procurement, Supply Chain, and IT defined the success with the assistance of an acceptance matrix of metrics versus controls.

In scope were customers, vendors, materials, and SKUs, GL accounts, open accounts receivable and payable, open purchase orders, bills of materials, routings, and transactional history adequate to maintain subledger parity and inventory valuation. The migration horizon was set at a thirty-six-month activity period to maintain auditability and enable forecasting based on seasonal trends. The landing environment consists of on-premises applications, databases replicated to object storage, and a Snowflake warehouse. Orchestrated ELT pipelines made in dbt and CDC change data capture streamed into Kafka topics to be harmonized incrementally. Constraints were a single weekend cutover window at financial period-end, regional data residency to comply with GDPR, segregation-of-duties requirements to comply with SOX, and trading seasons where change-freeze periods would be required. Custom programming at the legacy side (and third-party add-ons) hampered direct access to tables, and thus intermediate stubs and curtailed pulls were demanded. Idempotent loads, deterministic transformations with versioned models, and full rollback capability (verified through dress rehearsals) were required by non-functional constraints.

The case adds a reusable canonical data model of Party, Location, Item, and GL Account with controlled vocabulary and governed semantics; a mapping playbook that translates SAP, Oracle, and Dynamics attributes to normalized data structures; and a cutover plan that has been successful in three regions. It reports an active-learning stewarding loop to entity resolution that combines blocking keys with deterministic rules and explainable scoring to cut manual review nearly in half, lifting precision. The governance blueprint also outlines RACI, change approval, and data-contract testing as part of CI/CD. To rehearse faster, practitioners can reuse DBT tests, reconciliation SQL, and Airflow DAG templates. This case illustrates risk controls-maker-checker approvals, SoD, PII masking, lineage, so there will be unification without a reduction in compliance or resilience.

This research has been carried out in several chapters. Chapter 2 describes integration patterns, migration techniques, MDM, governance, and mapping assisted by ML. Chapter 3 explains datasets, cleaning, profiling, the canonical model, and the integration stack: Snowflake, dbt, Airflow, Debezium, and Kafka. In Chapter 4, the data-quality, validation, and reconciliation framework that controlled the acceptance post-go-live has been specified. Chapter 5 contains experimental setup, baselines, ablations, operational outcomes, and cost effects—Chapter 6 talks of generalization, trade-offs, risks, and validity threats. Chapter 7 provides an overview of work to be undertaken in the area of streaming, programmatic labeling, online learning, knowledge graphs, and observability. The eighth chapter concludes by offering some recommendations that can be incorporated into ERP unification efforts.

2. Literature Review

2.1 ERP Integration Patterns & Canonical Models

To make cross-system orchestration or unified analytics plausible, large organizations with multiple enterprise resource planning (ERP) platforms need to rationalize diverse and heterogeneous data and process semantics. The question of connectivity topology is usually divided into the question of semantic harmonization by the architectural literature.

Point-to-point Integration links systems with customized interfaces that are interconnected bilaterally; it is convenient in the case of a small scope, as its size scales quadratically with increasing endpoints, change isolation can be poor, and testing recurs across connections. The centralized structure, or hub-and-spoke designs, consolidate mediation in a central broker. When the broker is an enterprise service bus (ESB), the message enrichment, transformation, and routing policies can be centralized and controlled as a common infrastructure; however, this approach can lead to a monolithic integration brain bottlenecking throughput and institutionalizing legacy pipes.

In modern portfolios, integration-platform-as-a-service (iPaaS) or elasticity, managed connectors, and integrated tooling are often needed in place of ESBs [18]. However, more complex patterns of orchestration are still supported by the ESB paradigm. Concurrently, API first strategies provide a consistent, versioned interface to domain capabilities like Maintain Vendor or Post Goods Movement that can both decrease heterogeneity at an interface level, enhance observability through a standardized telemetry view, and allow zero-downtime evolution via contract-aware versioning. As shown in the figure below, an ERP core exists in the form of a hub-and-spoke, representing various areas of financial, supply-chain, manufacturing, CRM, and analytics, with integration patterns that can be illustrated as based on a central broker (ESB) or iPaaS platforms managing transformation, routing, and orchestration. The contract-versioned API-first, cross-system interface is needed to reduce heterogeneity of interfaces, enable unified telemetry, and enable zero-downtime system evolution, critical to cross-system semantics and consolidated reporting and enterprise operations worldwide today.

Figure 1: Central ERP hub connecting domains for unified analytics and orchestration

Issues of semantic harmonization have been addressed using a canonical data model (CDM), which specifies enterprise entities, attributes, and relationships without reference to a particular ERP schema. The pragmatic CDM separates the idea of core concepts that remain stable: Party, Location, Item, GLAccount, and Document, with the extension areas that may vary by region, product lifetime, or regulations. Canonical identifiers separate downstream consumers of volatile source keys, and the rules governing the survivorship of attributes across multiple systems provide a golden representation of the attributes. Mapping patterns would be those of direct Mapping, where the field is isomorphic, computed Mapping, when derived attributes are included (such as incoterm normalization), and Mapping, where translation tables to translate codes are designed to maintain traceability of the values. In data platforms, this model is operationalized into laid zones (landing, raw, curated): landing provides source veridical landing, raw provides immutable storage of reproduction, and curated offers canonical tables keyed on surrogate IDs. The canonical schemas are tested to ensure that they are backward-compatible as sources advance. Canonical messages are often published in the form of event-driven integration to protect consumers uninterested in the background details (for example, Party Updated, Item Changed) [22]. Reduced coupling of the consuming party and increased velocity of change is a common outcome.

2.2 Migration Methodologies & Cutover Strategies

The definition of the migration methodology impacts when, across what period, risk, validation workload, and user disruption are dispersed. Big-bang cutover squashes the scopes into one transition window: master data, as well as open transactional balances, are loaded, interfaces are redirected, and users switch during a standard transition window. The strengths have been real-time semantic convergence, less dual maintenance, and less complex governance of the source of truth. The demerits include a focused operational risk, a narrow rollback window, and exhaustive rehearsal that simulates production volumes and schedules. Phased waves separate the scope by entity, geography, business unit, or process (for example, migrate vendors and purchasing before customers and order-to-cash) to minimize scope and allow it to stabilize, graft by graft. Waves, however, add interim reconciliation obligations, cross-system bridges, and fastidious dependency management to ensure continuity in processes between overlap periods. A parallel-run strategy concurrently runs the legacy and target systems over a specific time frame and reconciles the outputs to provide greater confidence before decommissioning; this approach is expensive because of extended dual data entry or the use of complex synchronization. Blue/green strategies keep two production systems behind a traffic cutover switch. There is rehearsal traffic at blue, and non-destructive validation at green, with cutover to the green system, which provides immediate rollback to blue by switching traffic back.

Regardless of strategy, literature has highlighted the importance of rehearsal discipline: extract-transform-load (ETL/ELT) dry-runs against production-scale snapshots; dress rehearsals to ensure that timing and performance, and control totals are correct; and hypercare plans covering the first days after go-live. Business continuity and incident response principles emphasize such parameters as effective determination of the decision threshold, communication trees, and contingency playbooks that specify the rollback criteria and stabilization actions during the cutover [20]. In practice, an amalgamation of methods is used, including waves to complex areas, blue-green across interfaces, and a time-boxed big-bang to final ledger opening. Automation is used to recreate target environments repeatedly, seeding common reference data, rebuilding indexes, and performing complete validation under a realistic workload. Entry gates have been formalized for each stage of rehearsal and require evidence of a reconciled balance, defect trend stability, and operator readiness before advancing.

2.3 MDM & Entity Resolution Techniques

MDM also aims to establish governance and technical means to seamlessly maintain relatively stable entities across a multi-ERP environment: customers, vendors, materials, and financial structures. The golden record is the central artifact, and it is a consolidated representation with survivorship rules in place that prioritizes the source systems, recency, and field-level completeness, as well as regulatory restrictions. Golden records need to be unravelable: lineage can be used to recapture every contributing attribute, including its origin, the type of transformation that was applied, and why it survived. Stewardship workflows on modern MDM platforms include steps to review, approve, and raise disputes on items in the database, and expose APIs and events to update processes back in the operational systems.

Entity resolution (ER) is an approach used to identify the circumstances in which two or more of the represented records are about the same real-world entity. Fellegi-Sunter-like probabilistic models take similarity vectors of names, addresses, identifiers, emails, and geospatial proximity and compute matching probability. In practice, ER pipelines use blocking to cull candidate pairs before scoring: sorted-neighborhood algorithms create windows on standardized keys (such as phonetic networks of names and postal codes) and locality-sensitive hashing (LSH) can partition similar strings in the same buckets to avoid $O(n^2)$ queries. The process of feature engineering is decisive: it is essential to carry out the tokenization with punctuation stripping and case folding, transliterations to

eliminate diacritics, dictionaries of nicknames and business names, noise removal (such as Ltd, LLC), and geocoding to compare the coordinates with the accuracy of a city. Thresholding methods are tradeoffs between precision and recall: a higher threshold results in auto-merges, a lower threshold leads to rejection, and a gray zone is placed on hold awaiting steward review. Clustering generalises pairwise matches into entity graphs and limits the possibility of errors of transitive closure, which may lead to forming chains of merges; graph-based rules ensure that records not fulfilling graph-consistent identifiers cannot be joined. The performance characteristics are incremental matching of change-data-capture (CDC) streams, idempotent updates, and replay safety [26]. Evaluation should report both pairwise precision/recall and cluster-level error (over-merge and over-split), as well as business impact such as duplicate reduction and blocking of invoices. Lastly, the golden layer is policed by the data quality rules, which are operational, including orphan detection, referential integrity checks, and code-set validation.

2.4 Data Quality & Governance in ERP Contexts

Data quality transcends structural, semantic, and temporal aspects and requires ownership. Complete coverage is evaluated at both field and record granularity with entity-specific rule weights; conformance ensures compliance to standards like ISO 4217 Currencies and ISO 3166 Countries; consistency ensures consistent referential integrity across hierarchies (such as material to valuation class) and cross-process joins; accuracy triangulates against authoritative registries and financial control total; timeliness provides service-level targets on CDC latency and batch freshness. A practical governance model establishes the ownership of domains and entities by data stewards and a platform team that manages controls. Policies are statements formalizing retention, masking of personally identifiable information, and segregation of duties between build and approve functions.

Corporate and IT governance cascades to data governance, which in turn drives data management and a formal data quality management capability, as shown in Figure 2 below. This framework delegates stewardship and platform ownership into the relationship between metadata, platform architecture, and platform quality strategy, policies, and standards, processes and controls. It operationalizes completeness, conformance to ISO 4217/3166, consistency of data, accuracy compared to registries and financial totals and timeliness of data via CDC SLAs, with retention, PII masking, and segregation-of-duties policies to business units.

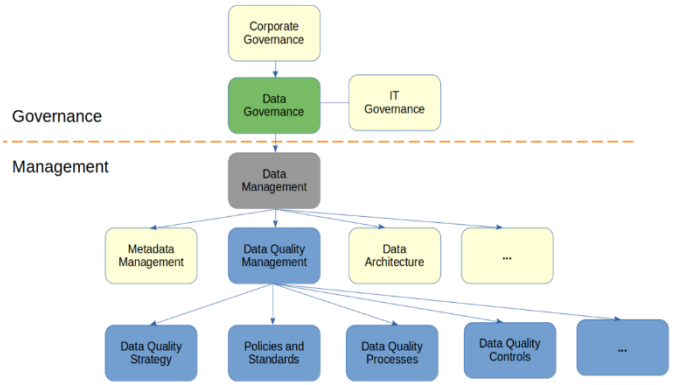

Figure 2: Governance and management hierarchy for ERP data quality ownership

Quality controls must be formulated as testable procedures fastened to pipelines and designs, not as non-operative documents. Deterministic rules enforce formats (such as VAT/ TIN regular expressions), code lists, and unit conversions; statistical monitors monitor skews in distributions, uniqueness, and duplication; reconciliation rules compare quantities and totals between source and target branches; and end-to-end assertions check that transforms derived across trial balances, sub-ledgers and open item lists still add to zilch overall. Dashboards roll up metrics, and alerts scale breaches to an on-call rotation. Incident management prescribes the severity, response goals, and evidence levels. Governance committees adjudicate rule changes and tolerate risk in the application of exceptions where such is justified by regional regulation or contractual reality [8]. Integrating these control mechanisms

into pre-production environments is conducive to shift-left validation; integration gates in deployment pipelines ensure that regressions never reach production. In cutover, the focus of governance is on traceability and sign-off: it is critical to produce immutable manifests of row counts, hashes, and parameter values so that the audit can reconstruct the chain of custody. After go-live, continuous quality incorporates defect breadboards, root-cause classification, and quarterly controls testing, to ensure ongoing quality control as structures and master data continue to change.

2.5 ML for Schema Mapping & Field Prediction

Machine learning is used to supplement conventional Mapping and validation by learning mappings between heterogeneous schemas, as well as to predict plausible values where sources conflict or are incomplete. The auto-mapping models use header tokens, value distribution, and sampled records to create a representation of similarity to canonical attributes, the candidate mappings being ranked with calibrated confidence. Embedding-based models can be trained using vector representations of the token and values such that semantically related columns cluster; the models can reduce manual mapping effort and maintain traceability by emitting human-readable justifications and examples of otherwise agreeing values supporting each recommendation. At the field level, supervised models will be used to predict values like tax codes, payment terms, or unit conversions based on available details of a vendor, material code, and transaction history. Weak supervision can help in such cases by using rule templates and high-precision heuristics to generate training labels, and by directing ambiguous cases to stewards to refine the model in successive iterations. Explainability is critical: feature importance, contribution plots, and counterfactual examples will allow the stewards to accept or overrule a prediction responsibly.

Operationally, such capabilities must be a part of continuous integration and delivery processes so that they can safely evolve with schema and volumes. Pipeline steps enforce training data lineage, checks of target leakage, model version recording, and parameter recording; release gates enforced reproducible gas metrics, bias test on new slices of data. DevSecOps advice is to make code analysis (static, dynamic, and dependency scanning) part of the build pipeline to ensure integration code, transformation logic, and model packaging achieve security baselines [16]. In production, models publish confidence scores and decision logs in addition to outputs, and thresholds drive auto-accept, auto-reject, and steward-review paths. Feedback loops record steward decisions and downstream exceptions, so the learning process is complete, and label data can be used to retrain. Predication-related service-level objectives are set based on latency and freshness to ensure automation does not hinder any data migration and data integration timelines.

3. Methods and Techniques

3.1 Data Set Description & Labeling

The program moved the operational and financial data in three incumbent ERP suites, SAP ECC 6.0, Oracle E-Business Suite 12.2, and Microsoft Dynamics AX 2012, to one typical warehouse called the canonical variant. Ingested entities spanned the entire order-to-cash, procure-to-pay, record-to-report, and plan-to-produce: customers and vendors; items, materials, and bills of material; routings and work centers; general ledger accounts and cost centers; open receivables and payables; sales orders, purchase orders, deliveries, and goods movements. A thirty-six-month period was used to factor in the seasonality, regulatory change, and master-data churn to support rehearsal and post-migration comparisons. The size of typical flows was in millions of parties, tens of millions of items, and open-item positions and billions of document lines, with sensitive attributes (tax IDs, bank accounts, email addresses, phone numbers, and ship-to addresses) in the PII category subject to PII restrictions. Extracts of native keys, timestamps, and change indicators were maintained in the extracted material to permit lineage, point-in-time reconstruction, and incremental refresh.

As highlighted in the figure below, centralized unification is a way of uniting heterogeneous ERP suites, such as SAP ECC/S4HANA, Oracle E-Business/NetSuite, and Microsoft Dynamics AX/365, in a canonical warehouse that supports supply chain, CRM, HR, finance, and accounting areas. The data covers plan-to-produce, order-to-cash, procure-to-pay, and record-to-report transactions in the thirty-six-month timeframe, retaining timestamps, change flags, and native keys. High-volume masters and transactions contain sensitive PII fields that allow lineage, rehearsal comparisons, incremental refresh, and stringent labelling to ensure the accurate consolidation of masters and transactions across the enterprise.

Figure 3: Multiple ERP suites unified across SCM, CRM, HR, Finance, Accounts.

Labeling was used both to do supervised entity resolution on the parties and the items, and to check field-level mappings (such as payment terms, tax categories, incoterms, units of measure). There were three channels through which gold labels were generated. Original legacy sources, such as cross-reference tables and interface journals, have an authoritative but outdated reference value between records traveling among systems. Second, the domain stewards adjudicated stratified samples of candidate pairs in a review workbench that surfaced record context, similarity features, and transformation pedigree. High-precision keys (tax ID + country, manufacturer part + brand, IBAN) were also used as positive seeds, where deterministic matching was applied after an automated validity check [31]. Balanced sampling cost/coverage: the candidate pairs were blocked by phonetic keys of the name, normalized addresses, and part identifiers, and further sub-sampled according to geography, business unit, and record age. Imbalance was handled by using fixed favorable ratios by stratum and with hard, informative nearest neighbor negatives. The inter-rater agreement (Cohen's k) was checked; disputed cases were referred to a steward council. The vendor consolidation strategy has several suppliers with qualifications in the regulated categories, so attributes of suppliers (approval status, quality rating, sourcing flags) have been labeled with special attention as part of the analytics about the dual-sourcing resilience [9].

3.2 Pre-processing & Standardization

Pre-processing harmonized heterogeneous data models, representations, and conventions before transforming to the canonical data model. Country, currency, and unit identifiers were harmonized with ISO codes; anchors on language-specific text were normalized through Unicode normalization, removal of punctuation, folding of accents, and locale-aware title and sentence casing. Emails were normalised to lower case with spaces removed; phone numbers were standardised into E.164 format with inferences on country based on the address line. Other fields were checked against VAT/TIN country patterns. The postal addresses were dimensioned (street, locality, region, postal code) with the postal authority references; the geocoding gave latitude-longitude pairs to the proximity features and the service-area rules. Item description text was degenerated of most boilerplate, harmonized in the use of standard abbreviations, and tokenized to support character and word n-gram similarity. Formulas, manufacturer names, and brands were standardised based on synonym tables.

Before any curated loads, structural soundness was imposed in staging. Surrogate keys were constructed deterministically based on business keys and both controlled namespaces to negate cross-system collisions and to allow idempotent upserts. Referential integrity had identified purchase order lines with no header

record, inventory transfers based on invalid material-plant pairs, open AR items mentioned in customers not present, and bills of material with invalid components. Quantitative values were brought to the same level of precision and magnification; currency conversions required a period-specific FX rate, with rounding being audited to under-cent levels. Timestamp properties were normalized to UTC, and the source time zone was captured during audit. PII protections enforced early in the process: sensitive columns hashed or masked on pre-production, and columns restricted by privileges [13]. Pre-processing also generated conformity indicators (unit validity, code-set validity, address parsability), which were acted upon as quality KPIs and as features in subsequent record-linkage models. All transformations were described as versioned, testable artifacts to ensure reproducibility and traceability by reviewers.

3.3 Visual Data Profiling

Visual analytics enhanced the pace of grasping data fitness, guided blocking design, and control selection. The user was provided with a profiling workspace that rendered completeness, uniqueness, conformity, and drift metrics on all the entities and attributes over the entire time horizon. Heatmaps of nullity and uniqueness were able to show sparse areas or over-keyed areas, such as optional tax identifiers in particular jurisdictions or over-constrained item codes in specific plants. Distribution charts showed heavy distributions, multi-modal lead times, and skewed payment term usage that would otherwise bias similarity measures and rule thresholds. The unsuspected dependencies exhibited by the correlation matrices led to selective standardization in certain areas, such as the family of payment terms correlating with the family of incoterms in individual sales channels.

Outlier detection combined robust z-scores with interquartile fences and domain rules that were used to identify negative values, improbable weights, or outlandish unit costs. Flagged records were directed to stewards along with contextual evidence so that they could be corrected at an upstream point where possible. Duplicate clusters were formed by aggregating candidate records that matched in blocking keys and ranked in the top-k in a group of similarity metrics; the review interface included juxtaposed comparison, subsplitting, and merging of clusters, and recording of adjudication rationale to guide retraining of heuristics. Temporal panels show where value-distribution drift exists along with known changes to the accounting system, such as chart-of-accounts extensions or unit-of-measure policy changes, allowing isolation of reversing the changes to the accounting system and actual business changes. The profiling also involved rule coverage views that indicated the number of records that would be covered by a claim validation or mapping rule that eliminated edging controls. Reconciliation cards - counts and monetary amounts split by entity and period - were available in the same workspace where the sampling and filtering decisions could be checked before formal cutover rehearsals.

3.4 Canonical Data Model (CDM) & Mapping Rules

The CDM was a stable semantic agreement that could break the coupling of legacy peculiarities and downstream applications and analytics. The core entities—Party, Location, ContactMethod, Item, GLAccount, DocumentHeader, and DocumentLine — had their semantics, valid kinds of values, annotations of lineage, and tests that can be executed. Subtyping was able to manage regional or line-of-business variance without schema forks: Party subtypes reflected special-purpose varieties of supplier and customer with shared identifiers and contact structures. Relationships maintained business semantics: Party-Location join tables represented the ship-to and bill-to connection with validity; DocumentLine points to Item and GLAccount with quantity, unit, and currency; many-about-one and one-about-many connections were direct and controlled, with cardinality enforcement.

As illustrated in figure 4 below, a canonical data model unifies semantics on heterogeneous sources to direct CRM/ERP/legacy feeds through cleansing, format conversion, and schema mapping into controlled canonical schemas. The

core entities, such as Party, Location, ContactMethod, Item, GLAccount, DocumentHeader and DocumentLine, contain lineage annotations and executable tests. Subtyping supports variance in the region or lines of business, without schema forking. Relationships provide business semantics: Party - Location join tables encode ship-to/bill-to validity, and each DocumentLine references Item and GLAccount with quantities, unit, and currency, which provides explicit cardinalities to ensure useful downstream analytics and applications.

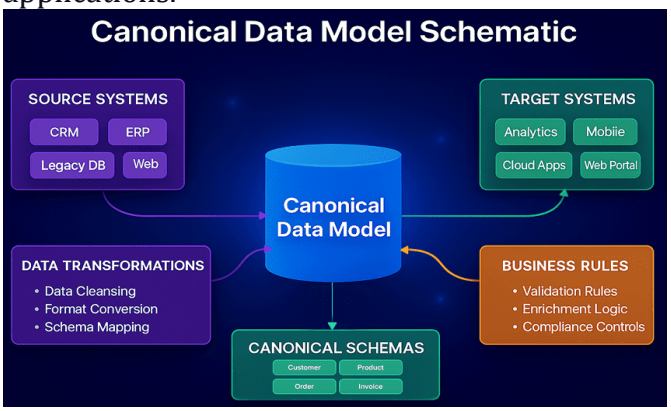

Figure 4: CDM centralizes semantics, transformations, and rules across systems

Mapping rules were merged with deterministic transformations and curated lookups. Code-set harmonization, payment terms, and incoterms, as well as types of taxes, were also mapped to governance-owned tables; unit conversions utilized audited conversion factors to help add the directionality of the conversion. In cases wherein the legacy systems differed on granularity, rules were implemented to create synthetic headers (e.g., consolidated delivery notes) or break out composite fields into normalized columns. [6] Survivorship policies were scripted per attribute with precedence chaining based on source authority (bank details in the SAP vendor master out-ranked the Dynamics on last-updated statuses and stewardship status within the thresholds). Disputes would cause exception routes, which would have to be confirmed by stewards before being accepted. Each of the mappings was written as a modular, test-supported transformation node with traceability to the underlying tables and commit SHAs, aiding reproducible builds and before/after comparisons. CDM and rule library were developed using change proposals that were versioned to allow revision history downstream consumers to determine whether new versions of the CDM or new rules were backward compatible.

3.5 Integration & Migration Architecture

The topology of the integration was landing-raw-curated zones on a cloud data platform. Landing ingested extracts- via secure file transfer, authenticated APIs, and using log-based change data capture. Files were checksummed, cataloged, and timestamped; schemas and row counts were recorded to reconcile. Raw saved in the original form of bytes with minimum metadata to retain evidentiary value—curated, fully mapped, quality-checked CDM tables, as well as reconciliation marts and quality telemetry. Transformations applied in-warehouse that performed an ELT pattern orchestrated through workflows that ordered a dependency, retries, and environmental promotions [30]. Dress rehearsals executed full pipelines against production-like snapshots, and run manifests were pinned to commit SHAs and configuration hashes.

Incremental movement was based on an event-driven spine. Changes to data in source ERPs were broadcast on topics partitioned by entity and business key; consumers used idempotent upsert by deterministic hashes to emulate exactly-once semantics in the face of retries. Backoff with jitter was used to cope with transient failures, where, instead, poison messages were moved to a dead-letter topic to be examined by a steward. High-water marks and checkpoint tables allowed backfill and live flow to co-exist. Fault tolerance measures are Jittery retries, circuit-breaker patterns, replayable logs, schema-evolution protection, and partition rebalancing, which were chosen to maintain integrity in the face of node drops or bursts of messages, as advised about resilient event-driven systems [5]. The validation checkpoints imposed complex stop rules (referential integrity, surrogate-key uniqueness) at communication

boundaries, but soft warnings issued stewardship tickets without blocking the pipeline. This cutover is a timed window: sources frozen, final delta run, financial and operational reconciliation, flip integrations, and alerts with monitoring of hypercare dashboards of latency, error budgets, and incident rates. Rollback plans could still be carried out, although the reversal scripts and snapshot restores were pre-computed and tested during rehearsals.

4. Data Quality Assurance, Validation, and Reconciliation Framework

4.1 Data Quality KPIs, SLAs, and Monitoring Architecture

The program sets realistic goals, which hold the migration and integration units accountable for specific results and on-demand responsibility. The Core KPIs are completeness of required attributes 99% or greater, meta-compatible conformity to canonical code sets 98% or greater, referential consistency for the coding across master and transactional entities, error rates of 1% or less based on sampling against source records, and change data capture latency times ≤ five minutes threshold at the ninety-fifth percentile. Monitoring is superimposed. Checks at the column- and row-level are both compiled into dbt and scheduled by Airflow, failures trigger incidents and halt subsequent loads.

Time-series metrics, including null ratios, distinctness, constraint violation counts, and drift scores, are cached in the warehouse and replicated to an observability store that is used to trigger alerts via Slack and PagerDuty. Thresholds on SLAs can be used as deployment gates: model deployments cannot move across DEV, SIT, UAT, or PROD until a threshold is achieved on the affected entities and partitions [1]. Modeling cannot move on until it meets some of these thresholds. A control matrix links each KPI to a particular automated test, escalation path, and RACI of data owners, stewards, and platform engineering. Data contracts serve as agreements between producing ERPs and the consuming services whereby the schemas and permitted value spaces are defined together with backward-compatibility assurances, coordinating the conflicting oversights with the needs of the stakeholders [15].

4.2 Validation Rules and Automated Test Suites

Deterministic validation rules operationalize the canonical data model and ensure that contaminated records do not become a part of your operational and analytical consumers. The domain, pattern, semantic, and business constraints enforce ISO currency/country codes; verify VAT/TIN formats and fields, using regional regular expressions; normalize units of measure and currencies, using authoritative conversion tables; and restrict credit limits, payment terms, and incoterms by region and customer types. There would be tiers to the rules, hard-stops block block loads, warnings would have to be acknowledged by a steward within defined SLAs, and informational checks would signal improvement opportunities.

Each table (Customer, Vendor, Item, GLAccount) includes a set of FIN/dt, Great Expectations, with partition knowledge of the company code, plant, and the fiscal period. Staging checks are performed during pre-load, curated model, and downstream mart checks after load. Referential integrity checks affirm that all transactional foreign keys can point back to a golden master. Other controls are row counts and hash totals, stratified sampling of spot audits, and lineage validation, which traces the original column that an attribute was transformed to and the transformation operation [3]. Artifacts, including green test runs, are a necessity before rehearsal and cutover approvals.

4.3 Financial and Operational Reconciliation Procedures

Reconciliation can show that the migrated state is sufficiently matched with the legacy environment so that business operations can resume as usual to agreed tolerances. Financial processes involve trial balance tie-out by company code; subledger general ledger parity in accounts receivable, accounts payable, and inventory; maintenance of aging buckets; carry forward of open items; and foreign-exchange revaluation parity in pre-selected ledgers.

Purchases and inventory processes encompass open purchase order line parity, continuity of three-way matches, stock valuation using valuation by classes and location, and stock movement history spot checks.

Figure 5: Order-to-Cash lifecycle checkpoints for reconciliation and operational validation

The order to cash process encompasses order creation, order confirmation, picking, packing, shipping, generating an invoice, delivery, collection of receivables, cash payments, cash application, reconciliation and reporting, as illustrated in Figure 5. Controls produce reconciliations of the counts and monetary amounts within tolerance ranges, check off handoffs using checksums, record diffs using lineage, and use IoT logistics telemetry to support timestamp alignment and event continuity. The Order-to-Cash will verify the open sales orders, delivery, backorders, and the pricing conditions; the Manufacturing will check on the BOM versions, routings, and work-in-process balances. Delivered totals (counts and monetary sums) are compared against partition totals on tolerance bands (such as monetary deltas l= 0.01 after currency rounding). Handoffs based on files are validated using checksums; record-level differences are created when the handoffs create exceptions, and have lineage snapshots and trace transformations made available per case. Reconciliation reconcepts of IOT on operational telemetry based on large-scale asset tracking can be applied to logistics data, including: timestamp alignment, event continuity, and reconciliation window inference [24].

4.4 Exception Management, Stewardship Workflows, and Remediation

The system of exception management is a closed loop that includes detection, triage, assignment, remediation, retest, and closure. Triage sorts based on the control criticality, monetary exposure, regulatory risk, and blast radius within processes. Cases are created in an enterprise ticketing system with logical templates that record entity, rule broken, affected partitions, lineage snapshots, and rollback plans [10]. Workbenches with deterministic rule hits and interpretable model signals and confidence indicators where ML-confirmed can assist matching; mass corrections can be made safely in large quantities with maker-checking in sensitive fields such as bank details and tax identifiers.

Remediation patterns are rule refinement, transformation updates, upstream or master maintenance, and one-time biasing backfills using idempotent scripts. All such fixes are cross-linked to regression tests to avoid repeats. Mean time repair and rate of reopening are tracked rule-wise and entity-wise on the ceremony based on quality debt burndown every week. As part of the new trends charts, the systemic problems have been highlighted. The backlog prioritization, as well as training of data owners and stewards regarding the standard error modes, and the preventive measures could be made against the systemic problems.

4.5 Compliance, Security, and Auditability Controls

Compliance requirements are implemented in the language of testable, continually-verified controls that endure internal and external audit. Role- and attribute-based access controls restrict privileges to data zones and least privilege; field-level masking to protect personally identifiable information like contact details and bank accounts; and environment isolation to avoid privilege creep. Run manifests make model versions, commit hashes, seeds, environment variables, dependency graphs, and warehouse sizing immutable, creating an audit trail where model identifiers can be traced to canonical surrogate keys.

Quarterly control testing samples reconciliation evidence, change approvals, access reviews, and user, time stamp, and change

context are recorded for read and write access [19]. Personal data generated on legacy ERPs is subjected to Data retention and right-to-erasure workflows. Integration platforms and warehouses' disaster-recovery goals are called into account during restore drills. In the case where model-driven matching is employed, privacy-preserving involves practices as well as minimization of sensitive attributes, which is in line with distributed analysis model patterns that minimize centralization of identifiable data [27].

5. Experiments and Results

5.1 Experimental Setup

The coursework of the experiments tested the complete data migration and integration chain whilst decoupling the contribution of the modeling components. Compute resources were composed of a cluster of eight m5.4xlarge instances used to generate features and blocking, and a Snowflake warehouse scaled to XL when generating candidate pairs and L in routine ELT. The toolchain consists of dbt 1.6 to transform and test data, Airflow 2.8 to perform the orchestration, and a Kafka cluster comprising three brokers to replay the change data capture [7]. Three areas were covered by source data, including customers, vendors, and materials, with open AR/AP, open POs, and transactional fact tables.

Table 1: Experimental setup for ERP data migration and integration pipeline

Aspect	Configuration / Tools	Scale / Parameters	Purpose / Notes
Compute	8× m5.4xlarge	16 vCPU, 64 GB RAM each	Parallel feature generation and blocking
Warehouse & CDC	Snowflake; Kafka (3 brokers)	XL for candidate pairs, L for ELT	Elastic ELT and CDC replay for increments
Orchestration & Transforms	Airflow 2.8; dbt 1.6	Versioned DAGs and tests	Reproducible pipelines and data quality checks
Data Scope	Customers, Vendors, Materials; AR/AP, POs, facts	36-month horizon	Covers O2C, P2P, R2R, P2P processes
Dataset Size & Labels	3.2M customers; 1.1M vendors; 14M materials; 1.4B transactions	240k labeled pairs; 9.7% positives	Historical cross-refs + steward adjudication
Candidate & Reproducibility	Blocking with standardized fields (name, address, tax ID, phone)	Recall ≤ 98%; pair inflation ≤ 5%; pinned images/manifests	Efficient candidate generation; auditable runs & registry artifacts

The training corpus consisted of 3.2 M rows of customers, 1.1 M vendor rows, 14 M materials, and 1.4 B records of transactions over 36 months across systems. As highlighted in Table 1 above, labeling involved a combination of historical cross-references and steward adjudication of the remaining ambiguous pairs; a stratified sampling scheme used code and geography to provide 240,000 labeled pairs with 9.7% positives. Standardized fields, including name, address, tax ID, and phone, were used to generate the blocking keys; the candidate generation process used no more than a 98% recall budget with a maximum pair inflation

budget of 5 %. Experiments were run on reproducible manifests and containerized images with pinned dependencies and states. The metrics were also recorded as artifacts in a registry to allow comparisons and auditability.

5.2 Baselines vs. Proposed Models

The baseline included deterministic rules and a logistic regression (run over string and numeric similarities) that was calibrated. The approach proposed compared the use of gradient-boosted trees (XGBoost) to a Siamese architecture, which shared encoders on names and addresses, using contrastive loss and threshold calibration optimization. Similarity features used were JaroWinkler, cosine similarity over character n-gram (TF-IDF), phonetic keys, numeric delta, geospatial distance, and business rule flags. The blocking of generated candidates on composite keys, compared to 98% recall, constrained the comparison to keep probable matches. On the test held-out partition, the rules-plus-logistic baseline provided PR-AUC 0.914 and F1 0.887 at the decision threshold optimized to reduce false merges. PR-AUC was increased to 0.954 and F1 to 0.921, whereas latency was 8.3 ms per pair on the CPU with XGBoost. On pure GPU inference, it shows PR-AUC 0.963 and F1 0.931 in 5.1 ms per pair using the Siamese model. Ablations revealed that dropping phonetic characteristics decreased F1 by 0.012, whereas removing geospatial proximity decreased F1 by 0.007 in regions where addressing was ambiguous. The attention memory devices guided the Siamese architecture in prompting the model to consider token alignments when the attributes are partially inconsistent through experience, which is also similar to the dynamic memory inference network in pairwise judgments (Raju, 2017).

5.3 Operational Quality Outcomes

Operational quality was measured as a result of pre- and post-load test suites, reconciliation reports, and time-series monitoring according to data contracts. The existence of mandatory fields rose to 99.6% after migration (up to 96.8%). In contrast, conformance to the canonical model at the field level increased by 92.1% to 98.7% through region-specific normalization. The most obvious effect was the growth in unique customer parties, which rose 23.4% when experts merged duplicates, and the reduction in the entity sprawl of vendor golden records, which declined 19.1%. Survivorship rules that merged the source trust tiers with recency produced 97.9% verified accuracy on sampled attributes such as bank accounts, payment terms, and tax regimes. Referential integrity rules eradicated orphaned records in order-to-cash and procure-to-pay; the orphan rate decreased by 1.6% to 0.2%. The highest residual risk lay in the normalization of addresses for regions with atypical formats. 0.6 percentage points of conformity were retrieved by performing actions to directly address this risk, including remedial work and supplementing with postal reference tables. Open AR/AP aging, open POs, and valuation of inventories achieved reconciliation parity with monetary deltas of less than or equal to 0.01 - after currency rounding. The time-to-detect for data-quality regressions has been reduced from hours to minutes since dbt tests and warehouse queries sent their metrics to the alerting pipeline [23]. The quality metrics were consistent across the six refreshes, indicating that the boundaries and monitoring of the contract did not show any degradation when increasing load.

5.4 Cutover & Stabilization Results

Cutover had rehearsed steps including specified exit points and rollback check-off points; dry-run one took 11.0 hours with 64 defects, dry-run two in 8.6 hours with 21, and the dress rehearsal was in 7.9 hours with nine minor problems, all fixed before go-live. The time of cutting over production ended in 7.6 hours or within the eight-hour SLA without any severity-one incident. Rollback was demonstrated by returning to a parallel environment using pre-cut snapshots and carrying out checksum checks on control totals, using 30 and 70% completion decision gates. Stabilization was founded on layered control: DBT tests on model boundaries, warehouse queries to detect drift, testing of API endpoints to check API health, and reconciliation of financial and operational objects. Alerts were routed to an on-call rotation, and data-quality and

pipeline streams were monitored on dashboards at a five-minute timescale. The orchestration graph was used to perform idempotent work and compensating actions to contain failures without necessarily halting the rest of the domains. These techniques reflected broad guidelines for how communication systems that scale can be designed- fault isolation and backpressure that minimize the mean time to recovery and the blast radius [12, 25] At the hypercare stage, the incident rate was 0.3 incidents per thousand jobs, the remediation time was 42 minutes, and no data re-loads at all were necessary beyond the targeted replay of three Kafka partitions.

5.5 Business Impact

Quantitative assessment of business impact was achieved using the KPIs of finance and operations and validated using stakeholder surveys. The monthly close cycle was reduced by 1.8 days as subledger to GL reconciliations were unified and manual system tie-outs were eliminated. The ability to leverage procurement consolidated a 14.6% reduction in the amount of spend placed within preferred contracts, as a result of deduplicated payees in conjunction with harmonized payment terms, as highlighted in the table below. Stockouts were reduced by 9.2% quarter-on-quarter because of materials master unification, rectifying unit-of-measure mistakes, and duplicate SKUs that formerly splintered safety stock.

Table 2: Key business impact metrics from ERP unification across finance, operations, and technology

Area	Improvement	Metric	Key Driver
Finance	Faster close cycle	**1.8 days shorter**	Unified reconciliations
Procurement	Spend efficiency	**14.6% contract compliance gain**	Deduplicated payees
Inventory	Reduced stockouts	**9.2% decrease**	Master data unification
Invoicing	Higher accuracy	**+2.7%**	Standardized tax/address
Operations & Analytics	Faster cycles & insights	**6.4% faster orders; 41% quicker insights**	Golden records; consistent semantics
Deployment	Greater agility	**Failure rate 2.1% (down from 7.5%)**	Automated testing

The standardized tax and address data allowed for a 2.7 percentage point improvement in the accuracy of customer invoices, eliminating credit memos and duplication of work. The integration also saved 6.4% of order generation time by eliminating cross-system lookups and using golden records cached in the API edge. Analytics teams noted reductions in time-to-insight of 41% in spend and margin analysis by using data with consistent semantics and lineage, and thereby reusing previously developed models. Deployment frequency increased quarterly to monthly due to automated testing and manifest reproducibility to reduce the risk of changes; deployment failure rate decreased to 2.1% (from 7.5%). These benefits are consistent with the evidence that close integration of predictive analytics with business intelligence and DevOps practices shortens feedback cycles and enhances business agility when instrumentation and automation are a part of the delivery process [17].

6. Discussion

6.1 Effective Levers and Impact

This discussion summarizes how architecture and governance helped speed up the multi-ERP fusion and silence errors. The blocking design restricted the pairwise comparison of

candidates based on the splitting of sets of candidate records on keys related to the business domain, standardized names, postal codes, country-specific tax identifiers, and phonetic representations, thus reducing the combinatorial expansion of the set of candidates, and recalling, against legitimately similar versions. The keys were profiled using feedback on profiling and back-tests performed against adjudicated match sets and blocking failures provided counterexamples to develop rules further. On areas of disagreement between model confidence and rule-based heuristics, active learning focused on stewarding time.

It improved the curation interface, presenting ranked candidate pairs, rationale snippets, and lineage to enforce fast, consistent decisions. Paired-based approvals, Stewardship workflows introduced risk levels, and maker-checker approvals of key deposits and survivorship changes, without halting other workflows at lower risk levels. Automated test that ran continuously as data contract tests and dbt assertions ensured completeness, conformance, referential integrity, and invariants across staging and curated layers [2]. However, a combination of these levers enabled shorter matching latencies, less propagation of duplicates, and more explainability, which allowed achieving quicker cutovers and fewer post-go-live crises without jeopardising trust in golden records.

6.2 Trade-offs & Limitations

The results were made possible through trade-offs between risk, effort, and cost. The thresholds between precision and recall were addressed by cost sensitivity, where false merges were more expensive than false splits on financial parties; parameterising this conservatism led to increased adjudication on ambiguous groups, but avoided amorphous merging of separate counterparties in a single group. The rules walked a fine line between automation and manual follow-ups, depending on the risk levels, as the low-category fields were easily automated. Still, high-risk properties, such as bank accounts, credit limits, and tax numbers, needed manual review.

Interpretability versus raw accuracy emerged as a modelling/feature choice item, whereby the team preferred the simplicity of tree ensembles and rule hybrids in which the complexity was traded off against actionable explanations available at decision time. Legacy customizations constrained schema harmonization; the mitigations relied on transformation adapters and exception tables, but some niche edge cases required manual intervention. Local peculiarities, such as address formats, fiscal years, and local taxation systems, required the customization of the blocking keys and tolerance windows to eliminate, or at least diminish, bias. Cost constraints influenced architectural decision-making; elasticity and service granularity were used to optimize cost whilst maximizing throughput, a finding which highlights tension in distributed architectures between scale and Cost [4, 29].

6.3 Threats to Validity

Several issues are putting internal and external validity at risk. A sampling bias may occur in case labeled pairs tend to over-represent simple duplicates of geographies with high volume and under-represent subsidiaries with matching addresses but varying fiscal status; this could be corrected by sampling stratified by country, industry code, and enterprise size—labeling drift. In the case of stewardship criteria, a drift occurs between rehearsal and cutover, and silent drifts change ground truth; versioned decision rubrics, calibration sessions, and audits were introduced to ensure drift does not occur. Survivorship bias can overestimate precision when historical cross-references detect successful mergers; as a countermeasure, hard negatives and blind adjudication samples were used in the evaluation.

The risk of data drift persists before go-live; policy or vendor onboarding measures upstream can change distributions of value, reducing the blocking selectivity and the effectiveness of thresholds. Drift monitors, therefore, measured changes and acted on re-tuning. The external validity is only weak, as the industry structure may also affect it because manufacturing hierarchies and regulatory

framework constraints are different across service contexts, and the performance of address normalization in various regions also varies. Cutover tactics may be limited by operational constraints such as downtime windows, quarter-end freezes, and change moratoriums. A documentation of assumptions, boundary conditions, and non-goals was done so that consumers can decide what will be transferred and what needs to be adapted.

6.4 Risk & Control Considerations

Layered controls that combined identity, privacy, and change management were essential to lay the foundation of program defensibility. Stewardship was separated by role-based versus attribute-based access, and just-in-time elevation, as well as segregation of duties on build, allow, and deploy activities. Immutable audit logs provided actor, timestamp, before/after value data on sensitive fields, which was a chain of custody between the source identifiers and canonical keys. Masking of sensitive attributes in any non-production environments; secrets were rotated centrally; and right-to-erasure through automated identity-aware deletion were all implemented. Change control was extended to the mapping rules, survivorship logic, and cutover playbooks; each release encapsulated manifests, test evidence, reconciliations, and rollback activities.

Two measures of validating disaster recovery objectives included failover drills. Such practices coincide with the concepts of the zero-trust approach, which prioritizes constant verification, authorization, and telemetry-based enforcement to minimize the risk of lateral movement [21]. Business controls were based on financial and operational reconciliations: trial balance tie-outs, subledger-to-GL parity, open item continuity, PO line counts, and inventory valuation checks. The triaging of exceptions was done through a ticket queue; remediation was done through retest and approval before closure. In combination, the controls yielded audit, inquiry, and post-incident review evidence.

6.5 Comparisons to Prior Work

Compared to the usual vendor playbooks, in this case, the migration formalised data contracts and executable tests as first-class migration artefacts, avoiding relegating acceptance to passive checklists. The approach was in contrast to the big-bang migrations because it entailed running rehearsal waves with measurable gates and rollback exercises, hence, allowing gradualism. However, clean cutovers were still made possible. In contrast to MDM-centric deployments, the design placed ELT orchestration, observability, and financial reconciliation on an equal footing with mastering, recognizing that flows and controls to ensure the correctness of integrations are as crucial as matching logic.

The methodology was also distinct from the API-only integrations, as it included both batch ELT for backfills and change data capture for deltas, simplifying the management of performance and validation. Operationally, the program relied on artifacts that could be tested, such as dbt assertions, reconciliation notebooks, and scripted smoke tests, so stakeholders could verify progress without relying on handcrafted spreadsheets. The stewardship model also transitioned the ad-hoc review into a workbench consistent with surfacing rationale, lineage, and risk tier, providing a shorter cycle time with redundant audit of sensitive modifications. Combined, such decisions transformed ad hoc heroics into repetitive mechanics that can be easily applied in other multi-ERP unifications with minimal adaptation.

7. Future Considerations

7.1 Streaming & CDC at Scale

Event-first integration must evolve beyond batch ETL and into streaming change data capture with transactional order and keys. Oracle GoldenGate can feed heterogeneous sources into Kafka, and Kafka can be fed with ERP logs using Oracle GoldenGate, as well as DB2 CCSID. The target warehouse can accept CDC in append-only tables, materialize type-2 dimensions, and feed a reconciliation micro-batch every 60 seconds. Idempotence of consumers, schema registry enforcement, and dead-letter topics safeguard against cutover in the pipeline. Back pressure management is done through Kafka quotas and

consumer lag alerts, and Airflow orchestrates the catch-up jobs after maintenance windows.

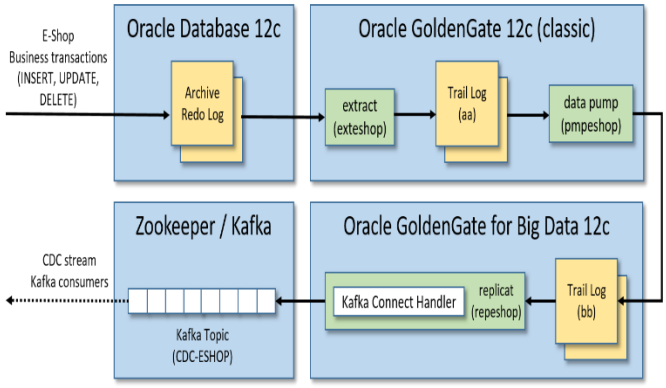

Figure 6: Oracle GoldenGate streaming CDC pipeline integrating ERP logs into Kafka

As shown in Figure 6 above, Oracle GoldenGate 12c agents capture change data in the Oracle Database 12c redo logs and write them into trail files, which are then transported by data pumps and replicated into Kafka topics to be processed in real-time. The Kafka connector-based integration guarantees the reliability of publishing heterogeneous ERP logs, and the schema registry, combined with the dead-letter issues, can reduce data quality risks. Consumers are idempotent to preserve transactional order with back-pressure controlled by Kafka quotas and lag notifications. Airflow scripts catch up on flows after maintenance, downstream warehouses solidify append-only CDC tables and type-2 dimensions, and every 60 seconds, micro-batch reconciliations are performed.

7.2 Programmatic Labeling & Weak Supervision

Such label scarcity in ERP masters can be handled with programmatic labeling, where deterministic business rules are encoded as label functions that a generative model reconciles to output probabilistic training targets. Weak supervision can hasten coverage of vendors, customers, and materials without losing auditability because of the provenance of rules. Steward workbenches are responsible for reconciling clashes between rules and model predictions, suggesting candidate merges or splits, and recording adjudications to use in retraining. The synthesis of the curriculum prioritizes ambiguous pairs, whereas versions of label policies preserve reproducibility. Checks and balances should have maker-checker authorizations and rollback of erroneous mass-edits identified using validation dashboards [14].

7.3 Online Learning & Drift Handling

To maintain the ability of entity resolution and mappings to be resilient over time, online learning must adapt thresholds and update models according to drift indicators. PSI could report a shift in names, addresses, or the description of products by feature and KL divergence on similarity distributions. A blue-green model deploys candidate releases behind feature flags; canary channels direct a proportion of traffic as pairwise accuracy is monitored. Steward actions are used in a series of incremental labels to retain pipelines. Automated fallbacks elevate thresholds, blocking, or use deterministic rules when drift is tolerated beyond established limits to guarantee safe fallback to a seasonal peak or post-schema changes [28].

7.4 Knowledge Graphs & Semantics

Knowledge graphs provide a single language of semantics across the array of ERPs, bringing the canonical keys and the legacy identifiers belonging to the partners, items, geographic locations, accounts, and relationships into a consistent model. Ontology alignment aligns disparate sets of codes, taxonomies, and attribute hierarchies into vocabularies, which can be used to reason about policies, perform lineage queries, and match in the context of additional evidence. Graph-native constraints find cycles, missing relationships, or contradictions in the bill-of-materials, and graph features add a neighborhood signal to improve similarity models. To implement a practical approach, the graph is stored in a scalable engine, such as MapReduce or Spark, which materializes relational projections to serve SQL workloads and offers APIs to validate and search [11]. Governance involves versioned articulated ontologies, change proposals, and automated analysis of effects before publication.

7.5 Continuous Data Quality and Observability

Pipelines that can be observed require contracts that guarantee continuous data quality. Producers publish specifications of necessary fields, values, and deprecation schedules; consumers provide contract tests in CI and during execution. The LAs and SLOs cover CDC latency, success rates, and freshness windows; SLI telemetry is fed into a time-series store used to power automated alerts and dashboards. Automated anomaly detection combines both the approach of rule-based thresholds and seasonality models. The lineage systems can track ERP origins, streaming topics, transformation nodes, and targets to enable blast-radius analysis and further extremely rigorous incident postmortems. The teams within operations departments institutionalize playbooks, quarterly chaotic drills, and metric reviews, stabilizing the unified environment as they allow safe, continuous improvement.

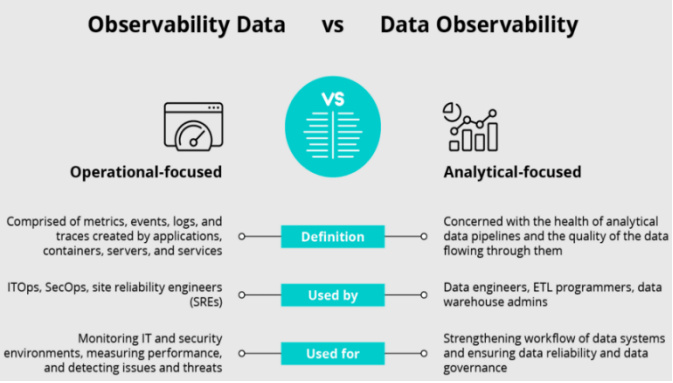

Figure 7: A comparison between observability and data observability for ensuring continuous data quality

The difference between observability data and system and pipeline reliability information brought about by continuous monitoring is illustrated Figure 7 above. The type of observability that is primarily operational is based on measures, logs, events, and traces, utilised by ITOps, SecOps and SREs to identify problems and performance anomalies in servers, applications, and containers. Analytical-oriented data observability instead addresses the fitness of analytical data pipelines and data quality within these pipelines, giving data engineers and the creators of ETLs greater control. It enhances governance with validation of contracts, enforcement of SLAs/SLOs, capture of telemetry, and detection of anomalies, including lineage tracking, allowing safe continuous improvement and reliable, resilient integration pipelines of ERP.

8. Conclusion

The study demonstrates that it is possible and safe to merge two or more ERP states when semantics, controls, and operations are designed in unison. Duplicate masters, inconsistent charts of accounts, and regional customisations have been removed by standardizing the data model in Party, Location, Item, and GLAccount, as well as document structures using a canonical data model. Consumers were decoupled, and harmonization of meaning between SAP ECC, Oracle E-Business Suite, and Microsoft Dynamics AX was possible using that abstraction. An auditable pipeline was built on an infrastructure of versioned dbt transformations, Airflow orchestration, and Snowflake execution anchored by a layered platform, landing raw and curated. IDC to DCDK into Kafka supported incremental harmonization and upserts that were idempotent, and runbooks practiced proved rollback. The behavior of schemas, code, and controls as deliverables that could be versioned and test-driven changed the brittle ad-hoc integration to a controlled, evolvable system. Operational roles and RACI ensured that ownership, approvals, and escalation during changes were safe.

At the methodological level, deterministic mapping and code-set translation were combined with entity-resolution pipelines built on blocking keys, phonetic encodings, n-gram similarities, geospatial proximity, and business-rule features to generate tentative pairs. Ambiguities became inserted into a steward workbench, where an active-learning loop prioritised informative pairs and recorded adjudications as training signals. This design minimized pairwise explosion, secured sensitive data on critical fields like supplier banking information and tax identifiers, and reduced review windows without costing auditability. Visual analytics measured the degree of completeness, uniqueness, conformity, and

outliers as well as drift and fed these measures into the blocking keys, thresholds, and rule coverage. This produced a reusable scheme: canonical schemas, mapping rules, blocking strategies, feature libraries, and an adjudication UX that transformed expert judgement into repeatable evidence.

Quality assurance was facilitated in the form of executable data contracts and layered validation. Column- and row-level constraints were used to enforce obligatory fields, ISO code-based compliance, pattern validity, referential integrity, and financial invariances between staging and curated models. Statistical monitors tracked skews and uniqueness; reconciliation linked trial balances, subledgers, orphan items, purchase orders, inventory balances, routing, and bills of materials to tolerances. Promotion gates prevented the deployment to DEV, SIT, UAT, and PROD unless the KPIs and SLAs showed green. Controlling compliance- L1 enabled RBAC, ABAC, PII masking, immutable run manifests, and segregation of duties, thereby keeping the pipeline in compliance with SOX and GDPR. Collectively, these actions created an unbroken assurance loop to uncover flaws early and maintain a choke point of evidence.

Results support the strategy both on technical and on economic grounds. A compressed cutover reduced cutover time down to 7.6 hours, which was less than the SLA of eight hours, and resulted in zero severity-one incidents during hypercare. Mandatory-field completeness was 99.6%, and field-level conformity was 98.7%; orphan rates decreased from 1.6% to 0.2%. Matching significantly outperformed a rules-plus-logistic baseline with the gradient-boosted trees and Siamese architecture, improving PR-AUC and F1 on millisecond latency. In finance and operations, the number of days to month-end close decreased by 1.8 days; indirect spend consolidated by 14.6%; stockouts decreased by 9.2%; invoice accuracy increased by 2.7 percentage points; order generation time went down 6.4%; analytics time-to-insight improved by 41%; and deployment failure rate reduced from 7.5 to 2.1%. DC latency achieved five-minute P95 goals in freeze windows with no schema drift.

Various limitations guide adoption. Address normalization is still localized; custom legacy names introduce cases that are difficult to automate, and precision-recall is trying to get away with aggressive thresholds where false merges are something to be avoided. However, artifacts are portable: canonical model, testable transformations, reconciliation packs, observability dashboards, and orchestration graphs can be deployed with little tailoring. The practitioners ought to prepare blocking before modeling, automate clarifying tests, institutionalize maker-checker stewardship, and practice cutover with transparent cutback standards. In the short term, we will develop stream-first CDC, programmatic labeling, thereby broadening its coverage, a learning module that uses drift monitors and knowledge graphs that represent the understanding of the semantics, and speed using lineage. In short, enhanced semantics, designed-in controls, and operational discipline turn multi-ERP consolidation into an event to be repeated and bring the benefits of higher-quality data, less cycle time, and enhanced business agility without compromising compliance and resilience.

References;

[1] Alqahtani, A. M. (2020). *Service level agreement specification for IoT application workflow activity deployment, configuration and monitoring* (Doctoral dissertation, Newcastle University).

[2] Bisegna, A. (2023). Automated Security Testing for Identity Management of Large-scale Digital Infrastructures.

[3] Chanajitt, R. (2023). *Machine learning approaches for malware classification based on hybrid artefacts* (Doctoral dissertation, The University of Waikato).

[4] Chavan, A. (2023). Managing scalability and cost in microservices architecture: Balancing infinite scalability with financial constraints. Journal of Artificial Intelligence & Cloud Computing, 2, E264.

http://doi.org/10.47363/JAICC/2023(2)E264

[5] Chavan, A. (2024). Fault-tolerant event-driven systems: Techniques and best practices. Journal of Engineering and Applied Sciences Technology, 6, E167. http://doi.org/10.47363/JEAST/2024(6)E167

[6] Dhanagari, M. R. (2024). MongoDB and data consistency: Bridging the gap between performance and reliability. *Journal of Computer Science and Technology Studies, 6*(2), 183-198. https://doi.org/10.32996/jcsts.2024.6.2.21

[7] Dordevic, D. (2020). *Data Sovereignty Provision in Cloud-and-Blockchain-Integrated IoT Data Trading,"* (Doctoral dissertation, University of Zurich).

[8] Fiseha, A. (2024). Constitutional Adjudication and Constitutional Governance. In *Federalism, Devolution and Cleavages in Africa* (pp. 321-395). Cham: Springer Nature Switzerland.

[9] Goel, G., & Bhramhabhatt, R. (2024). Dual sourcing strategies. *International Journal of Science and Research Archive*, 13(2), 2155. https://doi.org/10.30574/ijsra.2024.13.2.2155

[10] Gupta, M., Jalote, P., & Serebrenik, A. (2019). *Improving software maintenance ticket resolution using process mining* (Doctoral dissertation, IIIT-Delhi).

[11] Hartzell, K. (2023). Comparison of Big Data SQL Engines in the Cloud.

[12] Kangda, M. Z. (2022). Blast protection techniques: a review. *Archives of Computational Methods in Engineering*, *29*(5), 3509-3529.

[13] Kansara, M. (2021). Cloud migration strategies and challenges in highly regulated and data-intensive industries: A technical perspective. *International Journal of Applied Machine Learning and Computational Intelligence*, *11*(12), 78-121.

[14] Karwa, K. (2023). AI-powered career coaching: Evaluating feedback tools for design students. Indian Journal of Economics & Business. https://www.ashwinanokha.com/ijeb-v22-4-2023.php

[15] Karwa, K. (2024). Navigating the job market: Tailored career advice for design students. *International Journal of Emerging Business*, *23*(2). https://www.ashwinanokha.com/ijeb-v23-2-2024.php

[16] Konneru, N. M. K. (2021). Integrating security into CI/CD pipelines: A DevSecOps approach with SAST, DAST, and SCA tools. *International Journal of Science and Research Archive*. Retrieved from https://ijsra.net/content/role-notification-scheduling-improving-patient

[17] Kumar, A. (2019). The convergence of predictive analytics in driving business intelligence and enhancing DevOps efficiency. International Journal of Computational Engineering and Management, 6(6), 118-142. Retrieved from https://ijcem.in/wp-content/uploads/THE-CONVERGENCE-OF-PREDICTIVE-ANALYTICS-IN-DRIVING-BUSINESS-INTELLIGENCE-AND-ENHANCING-DEVOPS-EFFICIENCY.pdf

[18] Laakkonen, T. (2023). Transforming integrations of an organization from point-to-point to iPaaS.

[19] Lee, L., & Sawyer, R. (2019). IT general controls testing: Assessing the effectiveness of user access management. *AIS Educator Journal*, *14*(1), 15-34.

[20] Malik, G. (2025). Business continuity & incident response. *Journal of Information Systems Engineering and Management, 10*(45s), 451–473. https://www.jisem-journal.com/index.php/journal/article/view/8891

[21] Malik, G., & Prashasti. (2025). Implementing zero trust architecture: Modern approaches to secure enterprise networks. *International Journal of Networks and Security, 5*(1), 22–45.

https://www.academicpublishers.org/journals/index.php/ijns/article/view/3734

[22] Meier, S. (2024). *Message History Logics and Callback Control Flow Models for Automatic Event-Driven Application Analysis* (Doctoral dissertation, University of Colorado at Boulder).

[23] Narayanan, S., & Zephan, P. (2024). Real-Time Monitoring of Data Pipelines: Exploring and Experimentally Proving that the Continuous Monitoring in Data Pipelines Reduces Cost and Elevates Quality. *EAI Endorsed Transactions on Scalable Information Systems*, *11*(4).

[24] Nyati, S. (2018). Transforming telematics in fleet management: Innovations in asset tracking, efficiency, and communication. International Journal of Science and Research (IJSR), 7(10), 1804-1810. Retrieved from https://www.ijsr.net/getabstract.php?paperid=SR24203184230

[25] Sardana, J. (2022). Scalable systems for healthcare communication: A design perspective. *International Journal of Science and Research Archive*. https://doi.org/10.30574/ijsra.2022.7.2.0253

[26] Seenivasan, D., & Vaithianathan, M. (2023). Real-Time Adaptation: Change Data Capture in Modern Computer Architecture. *ESP International Journal of Advancements in Computational Technology (ESP-IJACT)*, *1*(2), 49-61.

[27] Singh, V. (2023). Federated learning for privacy-preserving medical data analysis: Applying federated learning to analyze sensitive health data without compromising patient privacy. International Journal of Advanced Engineering and Technology, 5(S4). https://romanpub.com/resources/Vol%205%20%2C%20No%20S4%20-%2026.pdf

[28] Singh, V. (2024). AI-powered assistive technologies for people with disabilities: Developing AI solutions that aid individuals with various disabilities in daily tasks. *University of California, San Diego, California, USA. IJISAE*. https://doi.org/10.9734/jerr/2025/v27i21410

[29] Suleiman, N., & Murtaza, Y. (2024). Scaling microservices for enterprise applications: Comprehensive strategies for achieving high availability, performance optimization, resilience, and seamless integration in large-scale distributed systems and complex cloud environments. *Applied Research in Artificial Intelligence and Cloud Computing*, *7*(6), 46-82.

[30] ToYou, R., & Arabia, S. (2024). A Conceptual Framework for Enhancing Data Ingestion and ELT Pipelines for Seamless Digital Transformation in Cloud Environments.

[31] Wahab, R. A. S. R., & Bakar, A. (2021). Digital economy tax compliance model in Malaysia using machine learning approach. *Sains Malaysiana*, *50*(7), 2059-2077.