1. Abstract

The presented proposal outlines a robust and scalable solution for integrating part numbers from the S1000D-Compliant Bill of Materials (BOM) with Sage100 Article-Database. Using an Automated ETL pipeline combined with real-time / scheduled synchronization modes, this approach ensures part numbers, assemblies, and component data remain in harmony across databases.

To imitate the current scenario, a Supabase database to act as the Central Development Database (CDD) that stores the BoMs, as well as an SQLite local database serving the ERP system to store the article data, are set up. Using FastAPI for backend/ETL engine, continuous synchronization is achieved. This complete application is developed along with an interactive GUI using the React framework.

2. Proposed Solution as an Executive Summary

An automated ETL pipeline with configurable synchronization modes:

- **Live Sync**: During stable production stages. For lower update frequencies
- **Scheduled Sync**: For rapid, continuous BoM updates. Batching optimizes performance.
- **Manual Sync**: As a failsafe. Also control, verification and approval

Key Benefits:

- Eliminates manual data entry errors
- Ensures ERP always has current BOM data
- Provides complete audit trail for compliance
- Scales from development (high-frequency changes) to production (stable BOM)

3. Recommended Tools & Technology Stack

Table 1. Core Integration Platform

Component	Technology	Justification
ETL Engine	Python FastAPI	- High performance - Async support - Auto API documentation - 10x faster than
Communication	Websocket	- Instant change detection

4. Integration Approach

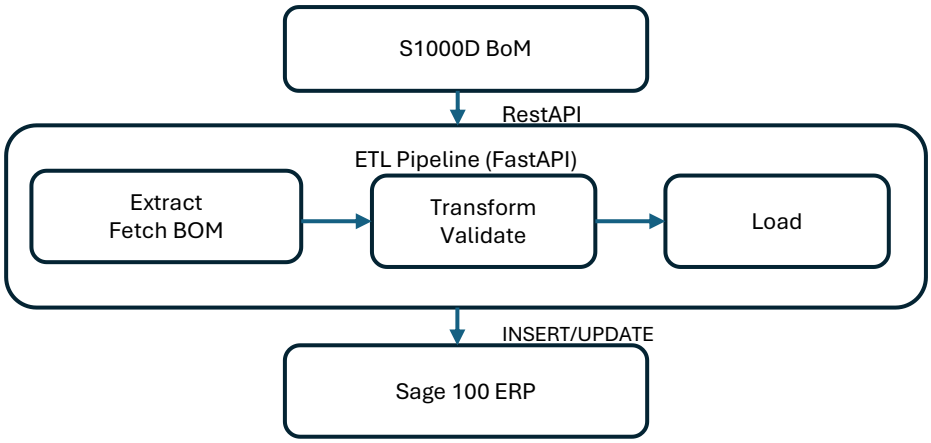

Figure 1. System Architecture

ETL Pipeline Process

EXTRACT Phase:

- Connect to CDD via REST API
- Fetch all assemblies (parent nodes)
- Fetch all components (leaf parts)
- Build hierarchical tree structure recursively
- Flatten tree for processing

TRANSFORM Phase:

- Validate part number format
- Validate description length
- Check unit price is non-negative
- Verify supplier information for components
- Log validation errors for review

LOAD Phase:

- Check if article exists in Sage 100
- If new: INSERT with all metadata
- If exists: UPDATE with new values (preserve stock quantities)
- Log operation result (inserted/updated/error)
- Record sync history for audit trail

Synchronization Modes

Mode 1: *Live Synchronization*: INSERT/UPDATE/DELETE actions are detected in real-time. WebSocket sends event to frontend. Frontend triggers immediate sync via API. Latency: < 2 seconds and best for: Production environment with infrequent changes

Mode 2: *Scheduled Synchronization*: Timer runs periodically (configurable). Batches all changes since the last sync and a single API call processes all changes. Best for: Development phase with rapid changes

Mode 3: *Manual Synchronization*: Full control over timing. Best for: Critical releases, testing

5. Expected Difficulties and Mitigation Strategies

Challenge 1: High-Frequency Changes During Development: Engineers may make many parallel rapid changes during development, causing system overload if each triggers a sync.

Solution: Implement configurable sync modes using Debounced/Scheduled sync groups changes together

n-rapid changes = 1 sync operation

Assumption: Development phase has high change frequency; production phase has low frequency.

Challenge 2: Data Validation & Quality: Invalid data (missing part numbers, negative prices) could corrupt ERP system.

Solution: Strict validation in TRANSFORM phase by rejecting invalid records with detailed error messages. Log all validation failures for review and never load invalid data into Sage 100.

Business Impact: Prevents procurement errors, maintains data integrity.

Challenge 3: Network Reliability (WebSocket Disconnection): Realtime connection may drop, missing critical changes.

Solution: Multi-layer approach wherein WebSocket monitors connection status and schedule sync as backup (runs regardless of WebSocket). Also, syncing on app startup ensures latest data. Audit log tracks all sync operations

Assumption: Network is generally stable but occasional disconnections occur. Eg: HomeOffice, Travelling

Challenge 4: Hierarchical BOM Structure: S1000D uses nested hierarchy (Drone → Systems → Sub-assemblies → Components). ERP needs flat article list.

Solution: Recursive tree traversal algorithm builds complete hierarchy from database and flattens to single list for ETL. Also preserves parent-child relationships and calculates assembly costs from component prices (if needed).

Technical Detail: BOM levels tracked (L0=product, L1=system, L2=sub-assembly, L3+=components).

Challenge 5: Audit Trail & Compliance: Need to track what changed, when, and by whom for regulatory compliance.

Solution: Comprehensive logging through logging sync operations with timestamp. Records: total parts, inserted, updated, errors, duration. Error messages preserved for debugging.

Business Value: Meets aerospace industry documentation requirements.

6. Example demonstration

An example application is developed to simulate the CDD/ERP environment at AVILUS. Feel free to watch the 3-minute pitch-in video as well as go through the GitHub repository.

Video-link:

GitHub repository:

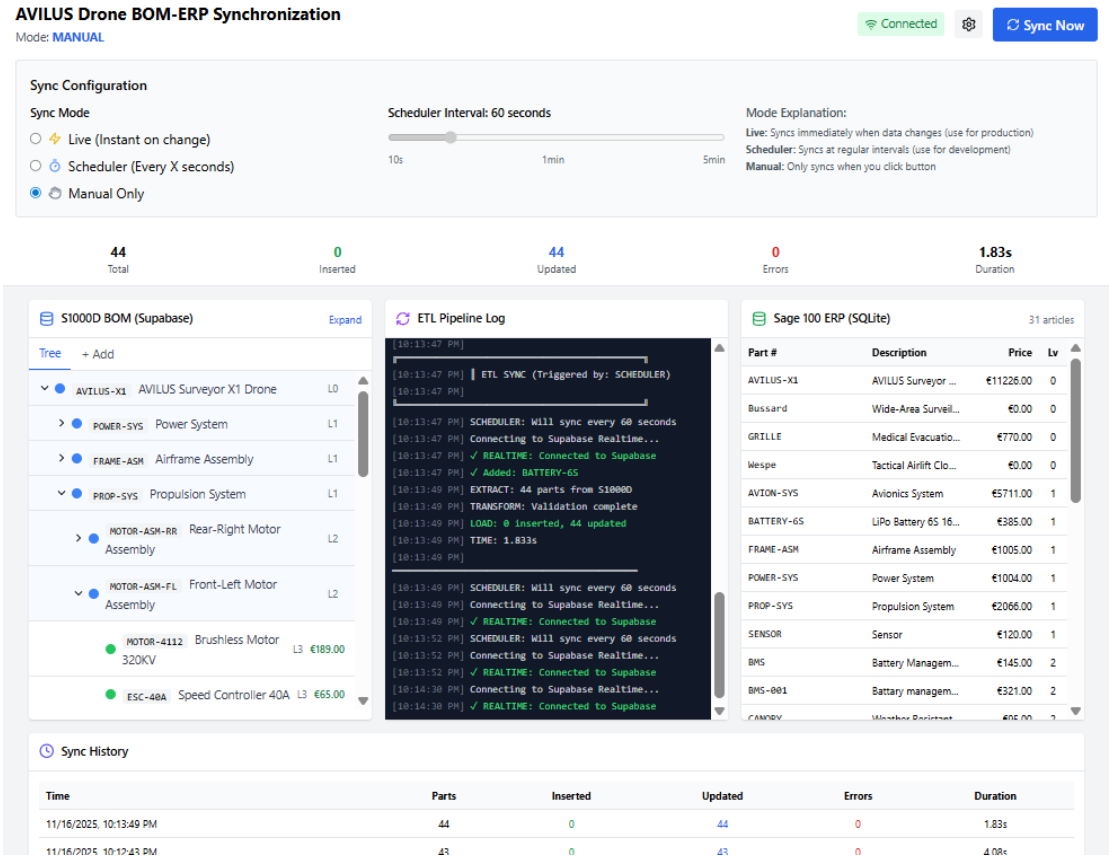

Figure 2. ETL Integration Dashboard (E2E Working prototype)