ARCHITECTURAL CASE STUDY

High-Speed Shop Floor Automation

Bridging the gap between an internal commercial ERP and a mandated Tier-1 customer platform (Plex) to automate 600 daily transactions with perfect traceability.

The Operational Friction

In high-volume manufacturing, manual data entry is a silent capital drain. Our client, a critical supplier, was tasked with processing approximately 600 subcontract receiving transactions daily into their customer's mandated Plex ERP system.

A conservative time-study placed manual processing at 45 seconds per transaction. Beyond the pure labor cost—representing 7.5 hours of continuous data entry every day—this manual rate fundamentally failed to keep pace with physical production, creating a severe operational bottleneck on the shop floor.

Furthermore, manual transactions in this environment carry an unacceptable risk of human error. Because many part numbers differ by a single alphanumeric character, they are highly prone to being mis-selected from dropdown menus. A mis-selected part number results in a completely mislabeled physical container, creating severe quality control issues. Un-finalized web transactions trigger customer complaints, which require costly investigation and response, ultimately dragging down the supplier's quality scorecard.

Business Impact

- Reduced transaction processing from 45 seconds to under 20 seconds
- Eliminated a 7.5-hour daily administrative bottleneck
- Created 100% bidirectional inventory traceability
- Reduced labeling errors caused by manual part-number selection or failure to finalize the transaction.
- Established a collaborative, data-driven feedback loop between the client and their Tier-1 customer
- Improved customer compliance and shipment accuracy
- Returned approximately 1,875 labor hours annually to higher-value work

The Software Gap

Twelve years prior, we built the client's original automation bridge to handle this exact process. However, that legacy system relied on Internet Explorer technology, which was rendered obsolete when Plex released a completely new, modernized interface. Compounding this shift, the client was actively migrating to a new internal commercial ERP.

A complete rewrite was required. The new architecture needed to seamlessly pull data from their new internal ERP and push it into the customer's Plex environment. The challenge? Plex did not offer an accessible API for this workflow, necessitating highly complex, headless web automation that could execute reliably at the speed of production. Furthermore, strict inventory control required bi-directional traceability—the serial numbers generated by the customer's Plex system had to be actively captured and linked back to the physical tags generated by the client's internal ERP to ensure payment and compliance.

> **The Operational Software Gap**
>
> The client's production processes evolved,
> their ERP evolved,
> and their customer's Plex environment evolved.
>
> The original automation could no longer support
> the way the operation actually functioned.
>
> This misalignment created an **Operational Software Gap:** a disconnect between how the business needed to operate and what the software environment was capable of supporting.

The Architectural Execution

1. Sustainable Technology & Vendor Independence

Our technology stack was selected with long-term sustainability in mind. By engineering the local interface using mainstream, open-source technology (Python and Tkinter), we ensure the client will always have plentiful choices for future development and support, eliminating the risk of being tethered to a single developer. Furthermore, because this open-source architecture scales completely free of software licensing friction, the client is seamlessly deploying the solution across at least two of their manufacturing plants. At Intrinsic Systems, we believe our clients should choose to work with us because they want to—never because they are held captive by proprietary, closed-loop technology.

This local application queries the internal ERP for a queue of finished goods containers. The operator simply selects a container from the on-screen list—or scans the physical barcode on the internal tag—and the automation executes, completing the transaction in under 20 seconds. Crucially, because this execution is entirely automated, the operator is completely free to focus on other physical tasks during this window, returning the full 45-second block of labor back to the business for every transaction.

2. Modular Data Architecture: ODBC & API Independence

To drive the automation, we defined the exact business specifications and authored the complex SQL queries required to extract production data from the client's internal ERP. We engineered the architecture to be highly modular, building the ODBC and API connectors as completely separate, interchangeable components. We initially deployed the ODBC module to immediately unblock the

workflow. As the system matured, the commercial ERP vendor integrated our SQL logic into their proprietary API framework, allowing us to toggle seamlessly to the native API connection.

This compartmentalization maximizes sustainability. If the client ever changes their ERP, or if a new business requirement emerges that the API cannot support, the architecture can instantly toggle back to the ODBC module. Major changes can be engineered and tested in isolation while absolute continuity is maintained across the rest of the application.

3. "Invisible" Headless Execution & Poka-Yoke

Because the automation drives a web browser to interact with Plex, we deployed headless execution, completely hiding the browser from the user. This *poka-yoke* (mistake-proofing) design prevents users from accidentally clicking or interfering with the transaction mid-flight. However, the system includes a surgical debugging toggle, allowing our engineers to make the browser visible during execution to rapidly investigate anomalies and deploy surgical engineering changes.

The Physical & Digital Flow

01

Production & Local Tagging

Product comes off the line. The client's internal ERP generates a local Finished Goods tag with a unique internal serial number.

↓

02

Intrinsic Automation Engine

The operator scans the FG tag. The custom Python architecture queries the ERP and executes the headless Plex web transaction in under 20 seconds.

↓

03

Tier-1 Plex Integration

Plex receives the data, validates the subcontract receipt, and returns a Plex Serial Number. A compliant Plex label is printed automatically alongside the FG tag.

↓

04

Compliant Shipping

The product ships with a custom Bill of Lading scansheet, explicitly correlating the Internal Serial to the Plex Serial for guaranteed payment tracking.

Strategic Error Handling & Alternate Tags

When a transaction fails due to physical or data anomalies on the shop floor, the system does not simply crash. Instead, it triggers a failure-mode protocol that prints an "Alternate Tag." This is not merely an error slip; the alternate tag maintains full physical part traceability while injecting the specific failure code into the physical label, the database record, and the system log. By sharing this failure-mode data with the client's customer, we turned compliance requirements into a collaborative feedback loop to optimize the broader supply chain.

Process Control & Continuous Improvement

A true architectural solution doesn't just automate a task; it establishes ongoing process control.

To achieve this, we engineered granular step-timers directly into the Python workflow. **Telemetry**—the automated collection and transmission of system measurements—allowed us to track exactly what the automation is doing, how long each step takes, and the cumulative transaction time. This micro-metric telemetry was instrumental during initial deployment to identify and eliminate latency. Today, it serves as a permanent, data-driven control mechanism, ensuring the automation consistently monitors its own health and maintains peak throughput.

Finally, to completely close the loop, we collaborated directly with the commercial ERP vendor to help them utilize our data architecture to generate a custom form that accompanies each bill of lading to the customer. This form explicitly correlates every internal client serial number to its newly generated Plex serial number. The result is absolute bi-directional traceability that guarantees compliance and ensures our client is paid without dispute.

Key Takeaways

- → Operational bottlenecks are often caused by software-workflow misalignment.
- → ERP replacement is not always necessary.
- → Sustainable architecture should avoid vendor lock-in.
- → Automation should improve both throughput and traceability.
- → Process control requires integrated telemetry; you cannot continuously improve what you do not measure.

Continue Exploring

Operational Software Gaps
https://intrinsic.systems/software-gaps

The Blueprint
https://intrinsic.systems/approach

Solutions
https://intrinsic.systems/solutions

Schedule a Strategic Assessment
https://intrinsic.systems/contact