CASE STUDY

Building a Role-Based ERP Training System from Developer Test Scripts

How a State Agency ERP Rollout Was Transformed into a

Structured, Role-Based Learning Ecosystem

Jason Boursier

Learning Designer & AI Content Systems Specialist

March 2026

Executive Summary

A state regulatory agency was replacing its decades-old legacy systems with a modern cloud-based ERP platform (Microsoft Dynamics 365). The rollout affected hundreds of employees across two fundamentally different work environments: retail-facing field operations and headquarters administrative functions. Each group used entirely different modules, workflows, and terminology within the same ERP system.

The challenge was not the technology itself. The challenge was that the only source material available to build training from was developer test scripts—highly technical, step-by-step system validation documents written for QA engineers, not for end users. No training curriculum existed. No legacy materials were transferable. Everything had to be built from the ground up, under aggressive timelines, with content changing weekly as the system moved through configuration and user acceptance testing.

> *This case study documents the process of transforming raw developer test scripts into a comprehensive, role-based training system—including job aids, user diagrams, presentations, workbooks, process animations, structured VILT and ILT sessions—and the methodology that made it repeatable and scalable.*

The Situation

The Organization

A state-level regulatory agency responsible for oversight of a controlled retail industry. The agency operates a dual structure: a network of retail locations managed by individual operators across the state, and a central headquarters handling compliance, procurement, finance, and reporting. The ERP migration touched every operational function across both environments.

The Problem

The organization had committed to a full ERP migration with a firm go-live date. Training development was running in parallel with system configuration, which meant the source material was a moving target. The specific challenges included:

- **No existing training curriculum.** The legacy system had been in place for so long that institutional knowledge lived in people's heads, not in documented procedures. Retirements and turnover had eroded even that.
- **Developer test scripts as the only source of truth.** The implementation partner produced detailed UAT (User Acceptance Testing) scripts covering every system

transaction. These scripts documented exact field names, button sequences, expected system responses, and validation criteria—but they were written for QA testers, not learners.

- **Radically different user populations.** A cashier processing a daily transaction needed completely different training than a headquarters analyst running month-end reconciliation reports, yet both were "using D365."
- **Content changing weekly.** System configuration, business rules, and even screen layouts were being adjusted throughout UAT. Training materials had to be built in a way that allowed rapid updates without rebuilding everything from scratch.
- **Compressed timeline.** Training delivery needed to begin well before UAT was complete, requiring a development methodology that could produce high-quality materials from incomplete and evolving source content.

What Was Available

The raw materials I had to work with included:

- Developer/QA test scripts with step-by-step system transactions, field names, expected outcomes, and validation criteria
- SME working sessions and recorded system walkthroughs (often informal, conducted via Teams)
- Microsoft and third-party D365 documentation covering standard platform functionality
- Early-stage PowerPoint decks and partial process documents created by the implementation team
- Direct access to the D365 sandbox/test environment for screen captures and system exploration

The Methodology: From Test Scripts to Training System

The core innovation of this project was developing a repeatable methodology for converting developer test scripts—documents never intended for learning—into structured, role-based training assets. This section documents that process in depth.

Phase 1: Deconstruct the Test Scripts

What a developer test script looks like

A typical test script reads like a technical recipe. It specifies a transaction (for example, "Receive a purchase order shipment"), then lists every click, field entry, dropdown selection, and expected system response in sequence. It includes validation checkpoints ("Confirm status

changes to Received"), error paths ("If quantity mismatch, system displays warning"), and prerequisites ("User must have Inventory Manager role assigned").

These scripts are invaluable for system validation but completely unusable as training materials. They assume technical literacy, provide no context for why a process matters, skip over real-world decision points, and include dozens of system details irrelevant to the end user.

The deconstruction process

For each test script, I followed a structured extraction process:

1. **Identify the business process.** Strip away the QA validation language and ask: what is the employee actually trying to accomplish? "Receive a purchase order shipment" becomes "A delivery truck arrives at the store. Here is how you receive that inventory into the system so it shows up on your shelves and in your counts."
2. **Map the process to roles.** Determine which user roles actually perform this transaction. A test script might cover a single end-to-end process, but in reality, a store clerk initiates it, a manager approves it, and a headquarters analyst reconciles it. Each role needs different depth and context.
3. **Extract the critical path.** Identify the steps the end user must perform, separating them from system-level validations, background processes, and administrator-only configurations. The test script might have forty steps; the end user might only touch twelve of them.
4. **Flag decision points and error paths.** Note where the process branches based on real-world conditions: What happens when a shipment is short? When a product is damaged? When a price does not match? These become the scenarios and knowledge checks in the training.
5. **Capture screen context.** Use the sandbox environment to take annotated screenshots of every screen the user will encounter. Map these to the extracted steps so every instruction is paired with a visual reference showing exactly what the learner should see.

Phase 2: Build the Role-Based Architecture

Once the test scripts were deconstructed, the next step was organizing the extracted processes into a training architecture organized by role, not by system module. This is a critical distinction: the ERP vendor organizes documentation by software module (Inventory, Procurement, Finance), but employees think in terms of their jobs (opening the store, receiving a delivery, closing out the register).

Role-based learning blueprints

I created learning blueprints for each major user role. Each blueprint defined:

- The specific D365 processes that role performs, in the sequence they encounter them during a typical workday or work cycle
- Bloom-aligned learning objectives for each process (what the learner needs to be able to do, not just know)
- The modality best suited for each objective: job aid for quick reference tasks, interactive walkthrough for complex multi-step processes, scenario-based assessment for decision-heavy situations
- Prerequisites and sequencing: which processes must be learned before others, and which can be learned independently

User Role	Core Processes	Primary Training Modalities
Field Staff (Retail)	Daily transactions, inventory receiving, cycle counts, returns, register operations	Step-by-step job aids, Storyline process animations, quick-reference cards
Store Manager	Ordering, inventory oversight, reporting, exception handling, staff coordination	VILT sessions, workbooks with scenario exercises, detailed job aids
HQ Operations	Procurement workflows, financial reconciliation, compliance reporting, system administration	ILT workshops, process diagrams, advanced workbooks, reference documentation
HQ Analysts	Data extraction, report generation, trend analysis, audit support	Guided practice in sandbox, reference job aids, scenario-based assessments

Phase 3: Produce the Training Assets

With the role-based architecture in place, I built out a full suite of training assets for each process and role. Each asset type served a specific purpose in the learning ecosystem.

Job Aids

These are the backbone of the training system. For every process a user performs, there is a corresponding job aid: a concise, visually annotated, step-by-step guide that can be printed, bookmarked, or pulled up on a second screen while working in D365. Each job aid follows a consistent structure:

- Process title and one-sentence purpose statement (“Use this guide when a delivery truck arrives and you need to receive inventory into D365.”)
- Prerequisites (what must be true before you start)

- Numbered steps with annotated screenshots showing exactly which buttons, fields, and menus to use
- Decision-point callouts highlighting where the process branches ("If the quantity does not match the PO, see the Discrepancy Handling section.")
- Common errors and how to resolve them
- A "Who to contact" reference for issues beyond the scope of the guide

User Process Diagrams

For complex, multi-step processes or processes involving handoffs between roles, I created visual process diagrams showing the end-to-end workflow. These diagrams serve two purposes: they orient the learner within the bigger picture ("You handle steps 3 through 7; headquarters handles the rest"), and they give managers and trainers a tool for explaining how work flows through the organization. Each diagram maps the process across swim lanes by role, highlights system touchpoints versus manual steps, and marks the decision gates where human judgment is required.

Presentations (Facilitator and Learner Decks)

For VILT and ILT sessions, I built presentation decks in two versions. The facilitator deck includes detailed speaker notes, timing cues, discussion prompts, and transition guidance. The learner deck is a streamlined visual companion that reinforces key concepts without duplicating the job aids. Both decks follow a consistent visual language tied to the organization's branding and use progressive disclosure—starting with the "why" of a process before revealing the "how."

Workbooks

Workbooks are printed or digital exercise guides that accompany VILT and ILT sessions. Each workbook includes the learning objectives for the session, space for notes, guided practice exercises that mirror the sandbox environment, scenario-based activities where learners work through realistic decision points, and self-assessment checklists. The workbooks are designed so a learner can return to them after training as a reference, bridging the gap between structured classroom time and on-the-job application.

Process Animations (Storyline and Rise)

For processes that are difficult to convey through static screenshots—particularly multi-screen workflows or processes where timing and sequence matter—I built interactive process animations in Articulate Storyline and Rise. These include:

- Animated screen recordings that walk the learner through the D365 interface with callouts and narration
- Branching scenarios where the learner makes a decision and sees the system's response (for example, choosing how to handle a shipment discrepancy)

- Try-it-yourself simulation layers where learners click through a replica of the D365 screens in sequence
- Knowledge checks embedded at decision points to reinforce correct procedures before the learner moves forward

Storyline was used for complex, branching interactions that require custom triggers and states. Rise was used for linear process walkthroughs, microlearning modules, and quick-reference sequences that could be accessed on mobile devices.

Structured VILT and ILT Sessions

The training delivery model combined virtual instructor-led training (VILT) for geographically distributed field staff and in-person instructor-led training (ILT) for headquarters teams. Each session was built using a consistent structure:

- Pre-session: Learners complete a short Rise orientation module introducing the process context and key terminology (fifteen to twenty minutes)
- Live session: Facilitator-led walkthrough using the presentation deck, with guided practice in the D365 sandbox environment. Learners follow along in their workbooks. Sessions are kept to sixty to ninety minutes to manage cognitive load.
- Post-session: Learners receive the corresponding job aids and a short reinforcement quiz delivered through Rise. Results feed back to the training team to identify where additional support is needed.

VILT sessions are recorded and archived so field staff in different time zones or shift schedules can access them asynchronously. ILT sessions include additional hands-on lab time in the sandbox, which is particularly important for complex headquarters processes like financial reconciliation.

Phase 4: Managing Content in Motion

The single greatest challenge of this project was not building the initial training materials. It was keeping them current while the system was still being configured. During the UAT period, screen layouts changed, business rules were refined, new modules came online, and process flows were reworked—sometimes multiple times in a single week.

The modular content architecture

To manage this, I designed the entire training system around a modular content architecture. Rather than building monolithic course packages, every asset was constructed from discrete, independently updatable components:

- Each screenshot is a separate, labeled file linked to a specific process step. When a screen changes, only that screenshot is replaced, and it propagates across every asset that references it.

- Each job aid section maps one-to-one to a process step. Updating a step does not require rebuilding the entire document.
- Storyline interactions are built with variable-driven content wherever possible, so that changing a field label or step description requires editing a data layer, not rebuilding the interaction.
- Rise modules use reusable lesson blocks so that a single process explanation can appear in multiple courses without duplication.

This architecture meant that when a process changed—which it did, frequently—I could identify every asset affected, update the specific components, and push revised versions without the cascading rework that typically derails ERP training projects.

Results and Impact

While this engagement is ongoing, the methodology has already produced measurable outcomes:

Metric	Outcome
Training Asset Production	Complete role-based training suite built entirely from developer test scripts, covering all major D365 processes across four user populations
Development Methodology	Repeatable process for converting technical QA documentation into learner-ready content, reducing the typical development cycle from weeks to days per process
Content Resilience	Modular architecture enabling same-day updates when system configuration changes, eliminating the rework bottleneck that stalls most ERP training programs
Role-Based Relevance	Every training asset is mapped to a specific role and task context, so learners only encounter content relevant to their actual job responsibilities
Blended Delivery Model	Integrated pre-work, live sessions, and post-session reinforcement, providing multiple touchpoints for retention without overwhelming learners with marathon training days
Change Management	Training materials kept current through sixty-plus system changes during UAT without a single full rebuild of any training asset

Lessons for Any ERP Training Program

This experience reinforced several principles that apply to any organization building training for a major system rollout:

Developer test scripts are a goldmine, not a finished product. They contain the most detailed, accurate, and current documentation of how the system actually works. But they require deliberate translation—extracting the business process from the technical validation, mapping it to roles, and wrapping it in context that makes sense to someone who has never seen the system before.

Role-based architecture is non-negotiable. Organizing training by ERP module instead of by job role is the most common mistake in enterprise system training. People do not think in modules. They think in tasks: “How do I receive a delivery?” not “How does the Inventory Management module work?”

Build for change from day one. If your training architecture cannot absorb weekly changes without cascading rework, it will collapse under the reality of ERP implementation timelines. Modular design is not optional—it is the only way to survive the gap between system configuration and go-live.

Blend is not a buzzword—it is a survival strategy. No single modality can carry the weight of an ERP rollout. Job aids for the moment of need. Animations for complex sequences. Live sessions for practice and questions. Workbooks for reinforcement. Each serves a different moment in the learner’s journey, and together they create a safety net that no individual asset could provide alone.

Human-in-the-loop governance applies to training content too. Every process extraction was validated with SMEs. Every job aid was reviewed against the live system. Every scenario was pressure-tested with actual end users. The speed of production was enabled by the methodology; the quality was ensured by disciplined human review at every gate.

Looking Forward: Where This Methodology Scales

The approach documented in this case study—deconstructing technical source material, building role-based training architectures, producing multimodal content suites, and designing for continuous change—is not limited to ERP rollouts. It applies to any enterprise system implementation, any large-scale process transformation, and any environment where training must be built from technical documentation rather than existing curriculum.

As AI-powered content platforms mature, this methodology becomes the foundation for automated content generation at enterprise scale. The structured extraction process I perform manually today—identifying business processes, mapping them to roles, flagging decision

points, and producing role-specific assets—is precisely the workflow that AI content engines are being built to accelerate. The human expertise required to design this architecture is what ensures AI-generated content is trustworthy, relevant, and aligned to real performance needs.

> *The future belongs to practitioners who understand both the craft of learning design and the architecture of AI-ready content systems. This case study is evidence that the methodology works in practice, under real enterprise constraints, with real stakes.*

Prepared by Jason Boursier
Learning Designer & AI Content Systems Specialist