AI Meets SAP: Building Real-Time Pricing Optimisation Pipelines Beyond Rule-Based Systems

How to transform static ERP pricing logic into intelligent, continuously evolving decision systems using modern data pipelines, AI, and DevOps practices.

By Sirisha Ayyagari
IBM corporation
Conf42 Database DevOps 2026

The Problem

Static Pricing in a Dynamic World

Rule-Based Constraints

SAP pricing rules are authored manually, deployed infrequently, and optimised for yesterday's market conditions not today's signals.

Lagging Market Response

Price lists and discount tables cannot react to competitor moves, demand spikes, or inventory shifts in real time.

Manual Intervention Overhead

Pricing teams spend disproportionate effort on exceptions, approvals, and overrides that intelligent automation could handle continuously.

Pricing: The Most Under-Modernised Enterprise Component

Organisations have modernised ERP finance, procurement, and supply chain with cloud and DevOps practices yet pricing pipelines remain one of the last areas still driven by static configuration and manual governance cycles.

Session Agenda

What We Will Cover

01

Why Rule-Based Pricing Breaks at Scale

Structural limitations of condition records, pricing procedures, and manual overrides in SAP SD/MM.

02

Hybrid Architecture Design

SAP as the transactional backbone; an external AI engine as the optimisation layer clean core compliance maintained throughout.

03

Real-Time Data Pipelines

Ingesting enterprise signals demand, inventory, competitor pricing, and customer behaviour continuously and reliably.

04

Operationalising ML in Production

Model deployment, drift detection, observability, and rollback strategies aligned with Database DevOps principles.

Chapter 1

Why Rule-Based Pricing Breaks

Rule-based pricing in SAP works well for stable scenarios, but condition records and pricing procedures were never built for constant market volatility. As the logic grows, change cycles create version lag and the system starts reflecting yesterday's assumptions.

At the same time, rule trees remain blind to live signals like demand shifts, inventory pressure, and competitor moves. Manual overrides, special approvals, and exception handling then pile up into operational debt, making pricing brittle at scale.

SAP SD PRICING | STATIC RULES

The Limits of SAP Condition Records

SAP SD pricing is powerful but fundamentally static. Condition records, access sequences, and pricing procedures encode business rules at configuration time not at execution time.

- **Version Lag**

  Pricing table updates require change requests, testing cycles, and transports typical lead time is days to weeks.

- **Context Blindness**

  Rules cannot incorporate real-time signals like live inventory levels, competitor feeds, or predictive demand models.

- **Exception Proliferation**

  Edge cases accumulate as manual overrides, creating audit debt and brittle logic that breaks under new business scenarios.

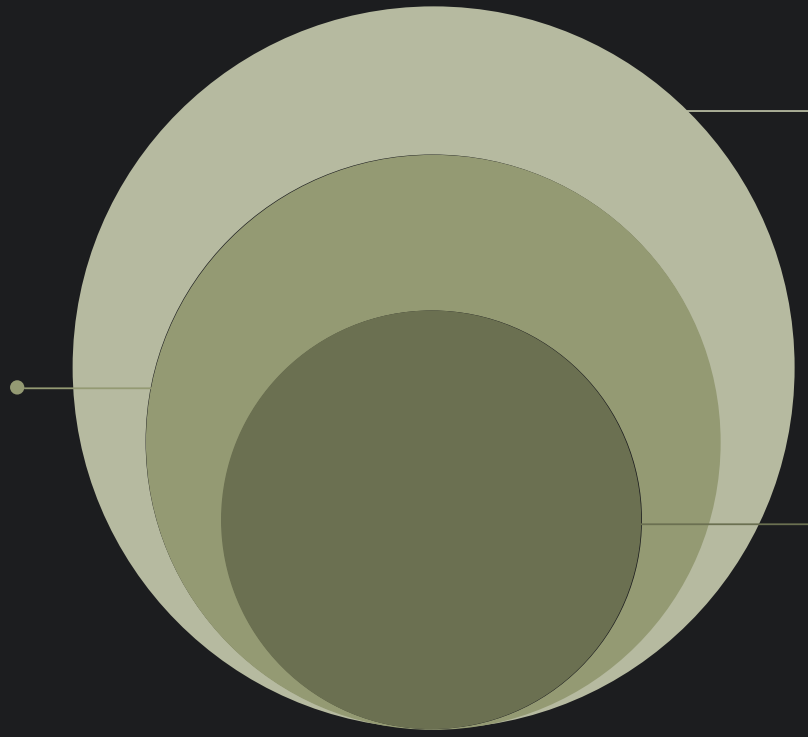
Chapter 2

The Hybrid Architecture

SAP remains the transactional system of record for pricing execution, order processing, and auditability. An external AI layer adds optimization, recommendation, and decision support without disrupting core operations.

This clean-core approach keeps SAP stable while extending it through controlled BAdI hooks and sanctioned API-based integration points that let real-time context shape decisions.

Architecture Overview: Two Planes, One Pipeline



SAP Transaction Core

Financial validation, order execution, compliance

Integration Layer

API gateway and SAP BAdI extensibility bridge

AI Optimization

Model inference, pricing & risk-adjusted recommendations

The AI engine never replaces SAP's transactional authority it enriches every pricing decision with contextual intelligence before SAP commits the outcome to ledger and order.

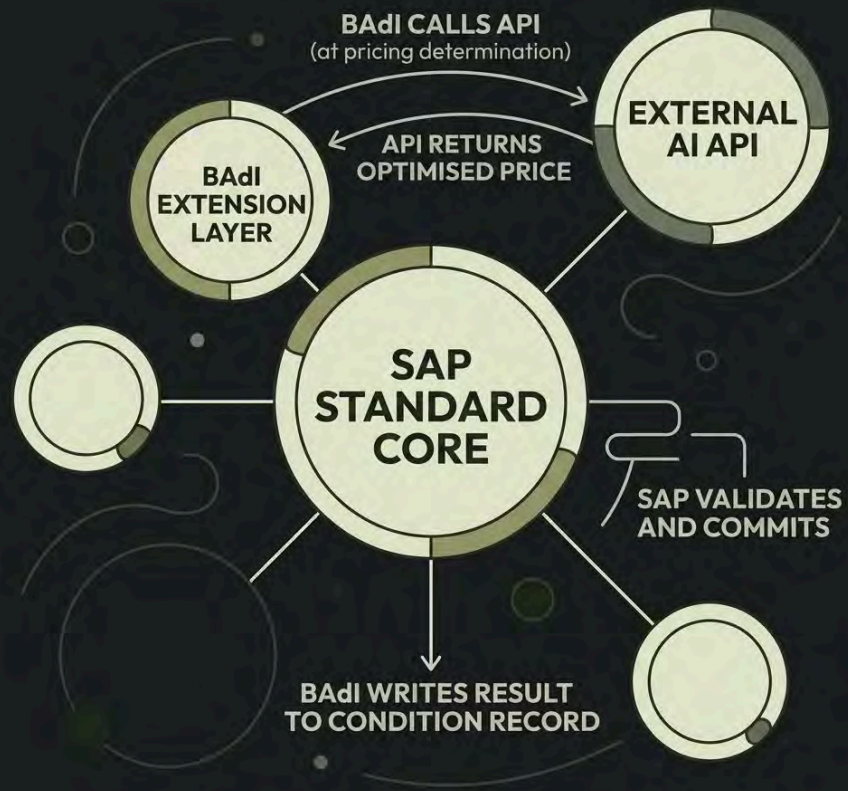
Clean Core Compliance via BAdI Extensibility

What Is BAdI?

Business Add-Ins (BAdIs) are SAP's official extensibility mechanism allowing custom logic to be injected at defined kernel hooks without modifying standard code.

Pricing BAdI Hook Points

- Condition value determination (`PRICING_PREPARE`)
- Net price calculation override (`PRICING_COMPLETE`)
- Manual price control (`V_T682Z` access sequence)



Chapter 3

Real-Time Data Pipelines

Continuously feeding the AI engine with the signals that matter.

This chapter covers the data foundation for responsive pricing: ingesting and normalizing SAP SD/MM events into a streaming architecture built for high throughput and low latency.

We'll also connect external market feeds like competitor pricing and demand forecasts so the AI engine always works from the freshest business context, not stale batch snapshots.

Designing the Pricing Data Pipeline

Enterprise Signal Ingestion

SAP SD/MM order history, inventory ATP, and customer master data streamed via CDC or SAP Integration Suite into the pipeline.

External Market Feeds

Competitor pricing APIs, macroeconomic indices, and demand forecast services enriched at ingestion time before model inference.

Streaming Infrastructure

Apache Kafka or IBM Event Streams as the backbone low-latency topics per pricing domain, schema-governed via a registry for pipeline reliability.

Chapter 4

Operationalising ML in Production

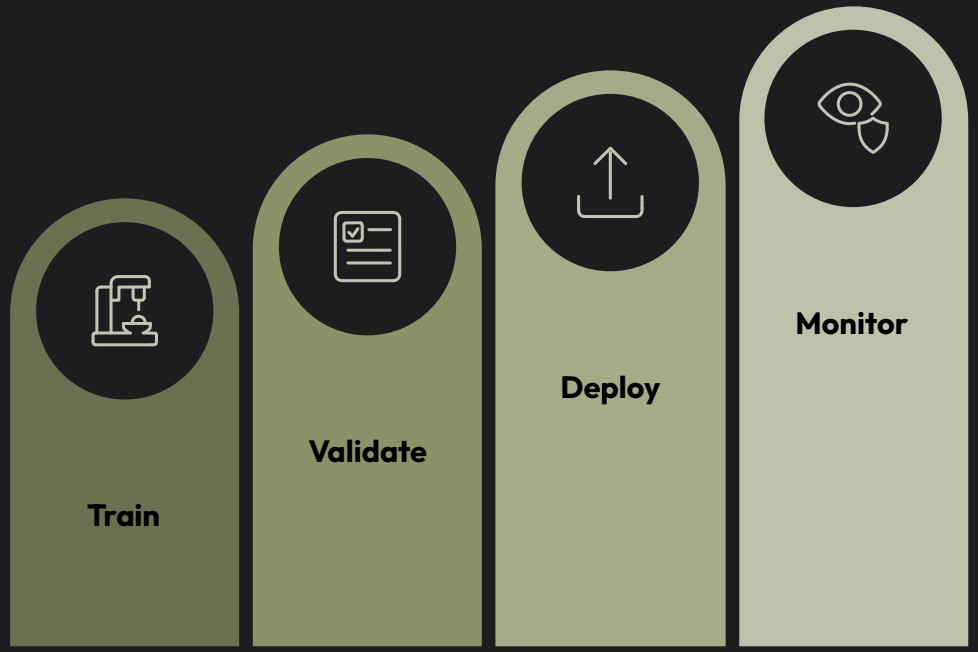
DevOps principles applied to model lifecycle, observability, and rollback.

This chapter shows how pricing models move from experimentation into controlled production release, with deployment patterns that balance speed, safety, and traceability.

We also cover drift detection and observability across data quality, feature stability, prediction behavior, and business outcomes so model performance can be monitored before issues affect pricing decisions.

Finally, we connect Database DevOps practices to ML lifecycle management: versioning, promotion, rollback, and governance become part of a repeatable operating model for pricing in production.

ML Model DevOps for Pricing



Key Operational Principles

Model Versioning

Every deployed model carries a semantic version, linked to the feature schema and training dataset hash.

Drift Detection

Statistical distribution checks on input features trigger alerts before pricing quality degrades in production.

Automated Rollback

Latency SLO breaches or anomaly thresholds revert traffic to the last stable model without human intervention.

Auditability, Consistency & Performance at Scale

- **Auditability**

  Every AI-derived price recommendation is logged with model version, input features, and confidence score satisfying financial controls and regulatory review requirements.

- **Consistency**

  A distributed cache layer (e.g. Redis) ensures identical pricing responses across parallel SAP application servers for the same order within a transaction window.

- **Performance at Scale**

  Inference SLOs of sub-100ms are achievable with model serving frameworks (ONNX, TensorFlow Serving) behind an API gateway with connection pooling toward SAP.

- **Governance**

  Discount floors, margin guardrails, and regulatory price caps are enforced as hard constraints in the AI engine not as SAP condition records keeping both layers coherent.

Key Takeaways

- **Pricing is a pipeline problem**

  Modernising it requires the same rigour streaming ingestion, schema governance, and CI/CD applied to any critical data domain.

- **Clean core is non-negotiable**

  BAdI extensibility and API-based integration let you move fast without accumulating technical debt inside SAP.

- **Operationalise ML like software**

  Version, validate, deploy, and monitor pricing models with the same discipline applied to application code in production.

Thank You!