Rethinking supply-chain and information systems: building the operational governance tier

Supply-chain information systems typically combine systems of record, operational execution, planning, analytics and increasingly AI services. As recommendations begin to influence actions, architecture needs a clear place for business decision authority. Data access, model governance and system permissions are all necessary, but they do not automatically answer who owns a consequential operating choice.

Distinguish the layers

An architecture review can separate four functions:

- Operational data: orders, inventory, supplier, shipment, production and quality facts.
- Analysis: forecasts, constraints, scenarios, optimisation and model recommendations.
- Decision governance: authority, policy, approval, exception routing and evidence.
- Execution: the systems that carry out approved transactions or instructions.

In practice, these functions may be distributed across several products. The purpose of the separation is not to force a particular vendor stack. It is to make the hand-offs and ownership visible. A recommendation should not become an execution command simply because it was returned by an API.

Keep decision logic connected to operating policy

When rules are embedded only in a model prompt, a dashboard or custom code, it can be difficult to know which policy applied and who approved an exception. Define policy ownership, versioning and change control. Specify the context required, the threshold for escalation, the permitted action and how overrides are recorded.

Decouple policy from an underlying transaction database where the architecture benefits from doing so, but preserve the links to source evidence and execution records. Ensure local sites can retain the authority required by their operating conditions. A reusable policy should not be activated everywhere without validation and local acceptance.

Evaluate the whole chain

Trace a real decision end to end: signal, data sources, analysis, policy check, approval, execution and outcome. Ask what happens when each stage is late, unavailable or inconsistent. Confirm identity and permissions at the technical layer, and confirm decision authority at the business layer. Make manual fallback and reconciliation part of the design.

Where Aidecis fits

Aidecis keeps one authority model and one decision record across your people, your own algorithms and every vendor’s agents. Aidecis never writes into your systems: your people, systems or bots execute the approved action, and Aidecis reads each execution back and reconciles it against the approval, so SAP stays your single source of truth. With enforced control, your own integration checks Aidecis's single-use grant before it executes. Agents and algorithms that write in their own systems are checked against their registered role and limits afterwards, and flagged within minutes if they went outside them: observed control, which does not prevent an action but shows it. It does not replace source systems, analytics or AI governance. A pilot should validate the architecture around one decision before the layer is extended across a network.

See how Aidecis works and explore the use cases.

Copyright 2026 Aidecis