▲ INFLEXIONPOINT · A FIELD GUIDE

THE MES ARCHITECTURE QUESTION

Do You Need an MES?

Rethinking the platform question as the technology landscape changes.

For operations, engineering, and IT/OT leaders evaluating a full MES platform.

▲ THE QUESTION

The answer used to be simple. It isn't anymore.

For twenty years, the answer to "how do we get real-time visibility, execution control, and traceability on the plant floor?" was the same: buy an MES. It was the only architecture capable of unifying production data, execution logic, and enterprise integration in one place.

That's no longer the only answer — and for some manufacturers, it may not be the best one.

> **This isn't an argument against MES. It's an argument for asking a more precise question.**

Instead of "should we buy an MES platform," the more useful question is: **which functions do we actually need, and what's the best way to deliver them today?** Sometimes the answer is still a full MES. Increasingly, it isn't.

▲ WHAT AN MES PROVIDES

Six core functions, one traditional bundle

Strip away the vendor branding and every MES is really delivering some combination of six core functions.

1 Production execution & work orders

Sequencing, releasing, and tracking work orders against the schedule — and enforcing the steps operators must follow to complete them.

2 Data collection & context

Pulling data from PLCs, SCADA, and historians, and giving it enough context — line, shift, batch, product — to be useful rather than just a stream of tags.

3 Quality & genealogy/traceability

Capturing inspection results, non-conformances, and full forward/backward genealogy — which lot went into which batch, on which line, with which material.

4 Performance tracking

Calculating availability, performance, and quality losses (OEE, downtime, scrap) and attributing downtime to specific causes.

5 Operator interfaces & instructions

Giving operators a system of record for what to do next and how to do it — replacing paper travelers and whiteboards.

6 Plant-floor ↔ enterprise integration

Passing production and quality data up to ERP, and pulling schedules and specifications back down.

Traditionally, these six functions were bundled into a single, tightly coupled platform because that was the only practical way to get them talking to each other. **That bundling — not the functions themselves — is the part now being challenged.**

▲ WHAT'S CHANGED

The unbundling of MES

A few shifts in the last several years have made it possible to deliver each of those functions independently, without buying the full platform.

Edge-first data infrastructure

Unified namespace architectures and protocols like MQTT/Sparkplug B now let plant-floor data get contextualized and made available across systems at the edge — before it ever needs to land in a monolithic MES data model. The "integration problem" that used to require MES as a hub is increasingly solved at the infrastructure layer instead.

Modular, composable software

Low-code MOM tools, standalone quality/traceability applications, and purpose-built scheduling tools now each do one job well. A manufacturer can assemble a stack of best-of-breed tools rather than accepting one vendor's version of all six functions.

AI/ML and modern analytics

Performance tracking and anomaly detection — historically a native MES reporting function — can now run on top of contextualized edge data using modern analytics and AI/ML tools, often with better pattern detection than legacy MES reporting modules offer.

Cloud/edge hybrid architectures

Because data can be governed and contextualized at the edge and still made available to cloud analytics, the "single source of truth" argument that used to justify a monolithic MES is weaker than it used to be. You can get a unified view without unifying every function into one platform.

> **It's now genuinely possible to deliver most MES functions through a composed, lighter-weight stack — at lower license cost, with more flexibility to swap components later, and without taking on a full MES implementation.**

▲ WHERE MES STILL WINS

Four conditions that still favor a full platform

None of this means MES is obsolete. There are real conditions under which a full platform still outperforms a composed stack.

A single validated system of record

In life sciences and other GxP environments, CSV requirements and 21 CFR Part 11 controls often favor one validated system over several loosely coupled tools. The audit and compliance overhead of validating multiple point solutions can exceed the cost of validating one platform.

High SKU or recipe complexity

When execution logic needs to be tightly coupled to work orders — many product variants, frequent changeovers, complex recipe management — the native coupling inside a full MES reduces the integration risk of keeping several disconnected tools in sync.

Multi-site standardization

For manufacturers running the same processes across many sites, a common platform can reduce long-term support burden more than a composed stack saves in flexibility. Standardization has real value at scale, even if it costs some agility at any single site.

Limited internal integration capability

A composed stack shifts integration and maintenance onto the manufacturer or a partner. Without internal capability — or ongoing partner support — to keep several systems talking reliably, the "lighter-weight" stack can quietly become more expensive to run than a single platform.

▲ The pattern across all four: full MES still wins when the **cost of coordinating multiple systems** is higher than the **cost of losing flexibility** in one system. That's a real tradeoff, not a default.

▲ A DECISION FRAMEWORK

Work the question function by function

Rather than answering "MES: yes or no" as a single question, work through it function by function and cost by cost.

1

Function-by-function need

WHICH FUNCTIONS MUST BE BUNDLED?

For each of the six core MES functions, ask: do we need this tightly bundled with the others, or can it be solved as a standalone tool at lower cost and complexity? Most organizations find they need some functions bundled — usually execution + genealogy in regulated settings — and others are fine standalone, usually performance tracking and operator interfaces.

2

Complexity thresholds

FOUR QUESTIONS TO SIZE THE PROBLEM

- How many lines and sites are involved, and how similar are their processes?
- How significant is the regulatory and traceability burden?
- How frequent are SKU changes, recipe updates, and changeovers?
- How mature is the existing data infrastructure — is contextualized data already available at the edge, or does that layer not exist yet?

3

Total cost of ownership

COMPARE THE WHOLE LIFECYCLE, NOT THE STICKER

Compare full MES license, implementation, and long-term maintenance against the composed alternative's license, integration, and ongoing maintenance — including who is responsible for keeping it running two, three, five years out. The sticker price of a composed stack is often lower; the integration and maintenance cost over time is the part that's easy to underestimate.

4

Time to value

HOW URGENT IS THE SPECIFIC GAP?

Composed stacks typically deliver partial value faster — a traceability tool live in weeks, an analytics layer live shortly after. A full MES has a longer time-to-value but delivers deeper cross-functional standardization once it's live. Which matters more depends on how urgent the specific gap is.

▲ THE TAKEAWAY

Where this leaves you

There's no universal answer, and any content — including this — that tells you MES is always right or always outdated is selling something. The right architecture depends on your regulatory exposure, operational complexity, and internal capability to integrate and maintain whatever you choose.

What matters is treating this as an architecture decision, not a software purchase decision. That means starting from the functions you actually need, evaluating the real total cost of each path — including the ongoing cost of keeping systems integrated — and choosing the platform or the composed stack that fits your specific environment, not the one the market defaulted to a decade ago.

> **Treat it as an architecture decision — not a software purchase decision.**

A vendor-neutral architecture assessment — one that evaluates function-by-function need before recommending a platform or a composed alternative — is the most reliable way to answer this question for your specific environment.

InflexionPoint provides that kind of assessment as part of our **Design · Build · Manage** approach — helping you land on the right architecture before you commit to a platform.