

When teams embark on ecommerce replatforming projects, they often face a critical decision early on: whether to buy bundled ecommerce modules or adopt a modular, API-driven approach. While bundled modules can seem like a cost-effective shortcut initially, they tend to inflate **year two total cost of ownership (TCO)**—sometimes dramatically. In my experience working alongside marketing, engineering, and finance teams at mid-market and enterprise retailers, tangled bundled systems practically guarantee hidden costs post-launch.



In this post, I'll explain why bundled ecommerce modules create more problems than they solve. We'll explore themes like modular scope discipline, long-term ownership versus one-off delivery, and the importance of clear system boundaries—backed by insights from *Netguru*, *DEPT*, and *Codal*, who understand these challenges firsthand.

The False Economy of Bundled Ecommerce Modules

Bundled ecommerce modules promise a lot: pre-integrated store functionality, fewer vendors to manage, and a “turnkey” experience. But these benefits come with hidden trade-offs that typically surface after the initial launch window.

- **Unused Features Inflate Costs:** Bundled modules often pack a wide range of features, many of which your specific business won't use. Yet they still drive up licensing and maintenance fees.
- **Tightly Coupled Components:** Bundled solutions connect deep internals of commerce functionalities, limiting flexibility for incremental improvements or replacements.
- **Vendor Lock-In and Limited Ownership:** Because you are locked into a package, evolving your platform or swapping out parts becomes an expensive, risky operation.

The result? What looked like a shortcut turns into a sprawling, cumbersome system—causing TCO to balloon in **year two and beyond**.

Why Modular Scope Discipline is Key to Controlling Costs

Companies like Netguru and DEPT advocate for strict modular scope discipline when building ecommerce platforms. This means:

- **Define clear boundaries:** Each module or microservice should serve a distinct, well-understood business function.
- **Avoid feature bloat:** Only build or license modules with clearly documented immediate value.
- **Resist “big bang” solutions:** Plan for incremental growth rather than buying everything upfront.

This disciplined approach helps teams avoid the trap of paying for “bundled ecommerce modules cost” that mostly go unused. Instead, your platform stays lean and maintainable over time.

Case in Point: Unused Features in Ecommerce Platforms

Codal, known for its UX-driven product design, often uncovers that merchants pay for dozens of “features” they never actually use. These features—common in bundled modules—require ongoing support, upgrades, and testing, increasing support overhead. This hidden offering gradually adds up, turning initial savings into bigger budgets down the line.

Long-Term Ownership vs One-Off Delivery

Ever notice how another misconception when dealing with bundled ecommerce modules is treating the project as a one-off delivery. Agencies and vendors sometimes pitch “we can do anything” approaches to finalize the platform quickly, but seldom provide clear guidance on ownership beyond launch.

That’s a problem I constantly raise in vendor meetings—“*Who owns this in year two?*” It’s a question too often ignored, but fundamental for controlling TCO:

- **Ongoing maintenance:** Software requires updates, bug fixes, and security patches.
- **Platform evolution:** Business needs evolve; the platform must adapt.
- **Operational overhead:** Training, monitoring, and efficiency improvements matter.

When bundled module providers don’t clarify roles and responsibilities for ongoing ownership—or lock you into thick layers of interdependent software—you’re left with costly, slow adaptations that pile up year over year.

Clear System Boundaries and Replaceability

One [modular commerce stack design](#) of the biggest gifts of an *API-first architecture* is clear system boundaries. This leads to:

1. **Replaceable components:** If a single module becomes obsolete or costly, you can swap it out without a full platform rewrite.
2. **Focused troubleshooting:** Bugs or performance issues stay localized rather than cascading across the platform.
3. **Parallel upgrades:** Upgrade one module independently, speeding your release cycles.

In contrast, bundled ecommerce modules result in tightly woven systems where boundaries blur, and change requests spiral into multi-month engineering efforts.

Codal highlights the importance of replaceability in their consulting work, noting that platforms with “clear separation and API-driven integrations” enable retailers to control scope creep and dramatically reduce long-term TCO.

API-Driven Integrations Enable Controlled Evolution

Consider a modern headless storefront powered by API-driven integrations. Instead of buying a monolithic bundle, the retailer picks and connects best-of-breed components:

- API-based CMS and PIM systems
- Headless storefronts tailored for frontend experience
- Modular payment, promotions, and inventory modules

This architecture imposes discipline on scope and ownership. Vendors like *Netguru* and *DEPT* have both published case studies emphasizing how controlled, API-first systems enable ongoing evolution without blowing budgets.



With API-driven integrations you can:

- **Benchmark module costs:** Measure and control the operational footprint of each integration.
- **Optimize usage:** Track feature adoption and retire unused capabilities.
- **Plan upgrades:** Smoothly upgrade or replace modules as needed.

Summary Table: Bundled Modules vs Modular API Approach

Criteria	Bundled Ecommerce Modules	Modular API-Driven Approach
Initial Cost	Lower upfront license + implementation	Potentially higher due to design and integration work
Year Two TCO	High due to unused features and vendor lock-in	Lower due to targeted upgrades, replaceability
Flexibility	Limited; tightly coupled modules	High; clear system boundaries
Ownership Clarity	Often unclear post-launch	Defined module-level ownership and responsibility
Evolution & Upgrades	Slow, expensive, risky	Incremental, controlled, quicker

Final Thoughts

Bundled ecommerce modules create a tempting illusion of simplicity and savings. However, without modular scope discipline and clear system boundaries, they rarely succeed in the long run. The hidden costs of unused features and tangled dependencies compound dramatically in year two and beyond.

The smarter approach is to embrace an **API-first, modular strategy** — where you own each component and keep expenses tied tightly to actual business value. Agencies and vendors like *Netguru*, *DEPT*, and *Codal* can help

retailers navigate this transformation to control *year two TCO*, maximize flexibility, and prepare for sustainable growth.

If you're facing a rebuild or headless rollout, ask your team or partners to map out your TCO out to year two and beyond. Insist on transparent costs tied to modular scope, and watch where those bundled ecommerce modules is often quietly inflating your future budget.