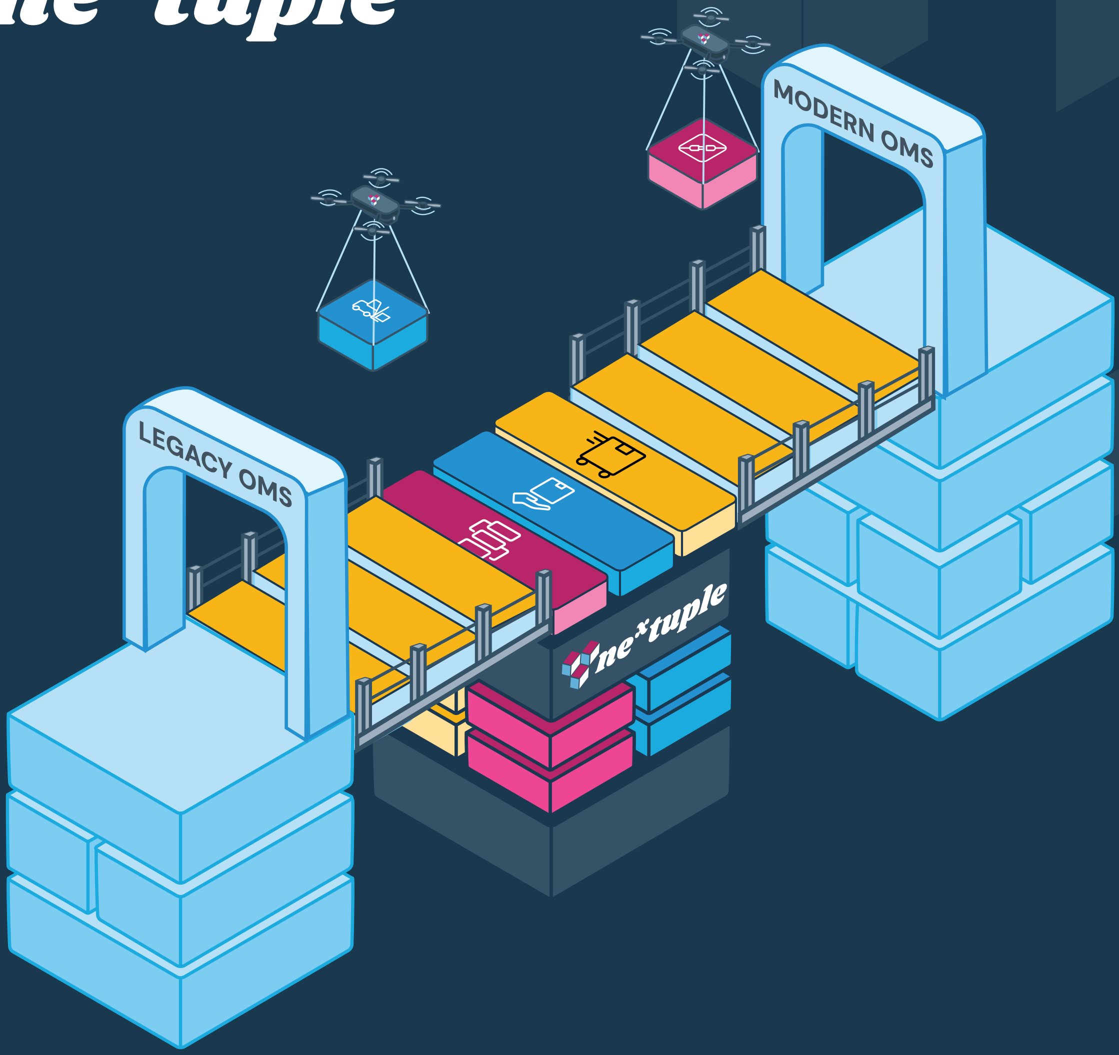


Composability for Order Management Systems: Beyond the Hype

www.nextuple.com

COMPOSABILITY FOR ORDER MANAGEMENT SYSTEMS: BEYOND THE HYPE

Executive Brief

The Limitations of Legacy OMS

For many retailers, their current, legacy Order Management System (OMS) acts as a roadblock on the fast-paced track of eCommerce and order fulfillment. These traditional systems, often monolithic and inflexible, hinder the agility needed to keep pace with evolving customer demands. This document explores the limitations of legacy OMS and introduces composable microservices as a powerful alternative that unlocks a new level of flexibility.

Breaking the OMS Monolith

The traditional perception of OMS as a single, all-encompassing application overlooks a key fact: its functionality can be separated into four distinct subdomains – **Inventory Management, Order Orchestration, Promising & Sourcing, and Store Fulfillment.** Each of these subdomains can operate effectively as an independent service, offering greater flexibility and customization potential. This shift in perspective allows retailers to unlock the true potential of their OMS.

The Power of Composability for OMS

Composable microservices offer significant advantages. Retailers can pick and integrate specific OMS components that best suit their needs, fostering easier customization and adaptation. The modular design allows for adding or removing components **without disrupting the entire system.** Additionally, APIs and microservices ensure efficient communication between the OMS and other critical systems. As businesses grow, composable OMS scales on demand, **eliminating costly overhauls.**

True Composability vs. Marketing Hype

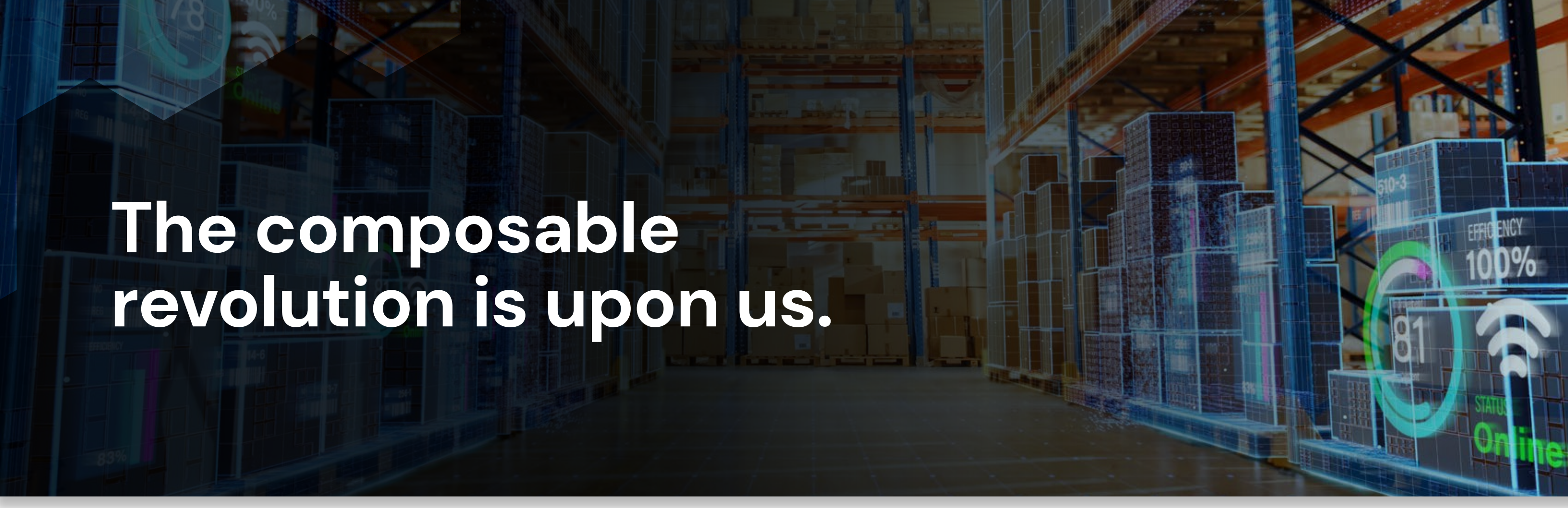
Many vendors offering “composable” solutions often fall short of that claim. Limited customization options and pre-packaged modules can force retailers to settle for a less-than-ideal fit. Integration with other vendors’ microservices can also be complex.

The Nextuple Solution

Nextuple offers a truly composable OMS solution. Deep customization capabilities empower retailers to tailor the system to their specific needs. Each subdomain can be deployed and customized independently.

Additionally, Nextuple offers deployment flexibility for single-tenant, multi-tenant, on-premise, or cloud-based implementations.

Don’t settle for limitations. If your current OMS hinders agility, explore composable solutions and embrace a more adaptableOMS to keep pace in the e-commerce race.



The composable revolution is upon us.

Here at Nextuple, we've been anticipating the composable revolution since we started our journey in 2017 by building solutions for customers one microservice at a time. But we've noticed in the last couple of years that our competitors have shifted messaging from Order Management System (OMS) platforms to composable microservices that can help you achieve improved speed and agility.

And you can't blame them. A recent IDC survey reveals **67% of companies are considering a composable architecture** for the front end.[1] Gartner predicts that by 2026, organizations with mechanisms to reuse digital commerce modules will see a significant boost—a **60% improvement in digital innovation speed compared to 2022."**[2]

The message is clear: more companies are recognizing that composable microservices are key to unlocking agility and innovation.

However, a gap exists between vision and reality. Many retailers remain stuck in a mindset that packaged OMS applications cover the full set of use cases necessary for distributed order management. In a recent survey we conducted with IT stakeholders – a staggering 97% of customers are "very likely or somewhat likely" to stay with their current OMS platform and if they are not – they are still looking at other OMS "platforms" to replace it.

Meanwhile, industry leaders like Target, Walmart, Best Buy, Dick's Sporting Goods, Lowes and others have already adopted composable OMS microservices to replace legacy systems. The results speak for themselves. These retailers boast exceptional customer experiences, lightning-fast fulfillment options, and top-notch inventory visibility.

What's holding the rest of the market back? We believe three key factors exist:

- Perception
- IT Maturity
- Composability with Extensibility

[1] *SaaS Commerce Platforms: The Future of Simplified Business Operations, IDC, Filippo Battaini & Cristiano Quattrini, January 2024*

[2] *Increase Organizational Composability by Reusing Composable Commerce Technologies, Gartner, Sandy Shen, April 2023*

Perception

While the traditional view considers the OMS domain a set of interconnected functions best managed by a single application, a closer examination reveals a different reality. There are, in fact, four distinct subdomains within the OMS space that can function effectively as independent services.

4 Key OMS Functions as Independent Services:

01 *Inventory Management*

Inventory Management plays a crucial role in supporting OMS functionality. However, these functions and their data don't necessarily benefit from being tightly coupled with the core OMS. In fact, **a separate architecture designed for high availability and scalability better serves inventory needs.** This trend is gaining traction in the market. We recently helped a major Wholesale Club successfully replace Sterling Inventory with a dedicated enterprise inventory service. Similarly, Blue Yonder has deployed its inventory solution at several retailers like Tractor Supply to sit alongside Sterling OMS.

02 *Order Orchestration*

Order Orchestration, the original purpose of the OMS, remains central. It conducts the order journey from fulfillment and returns to payments and customer care. This core domain thrives as a unified whole, ensuring efficient management. But complementary services like order monitoring and repositories can seamlessly integrate alongside it, providing **deeper insights and supporting adjacent functionalities.** Many retailers, for instance, leverage external CRMs to access order data, demonstrating the flexibility of this approach.

03 *Promising and Sourcing*

Much of the business value in a traditional OMS comes from complex node selection and accurate delivery data for sales channels. These functions act as optimizations and configurations within the order orchestration process, similar to an external fraud system evaluating risk and feeding back into the order flow. Interestingly, the "promising" function, recently gaining traction, has seen a rise in independent software vendors (ISVs) like Shipium and Fenix Commerce offering solutions across multiple OMS systems. This trend likely stems from their development during the recent rise of composable architecture. Potentially, the same would be true for sourcing if it had been developed during this time. This shift reflects a **growing market demand for improved decision-making, conversion gains, and cost-effective solutions compared to traditional offerings.**

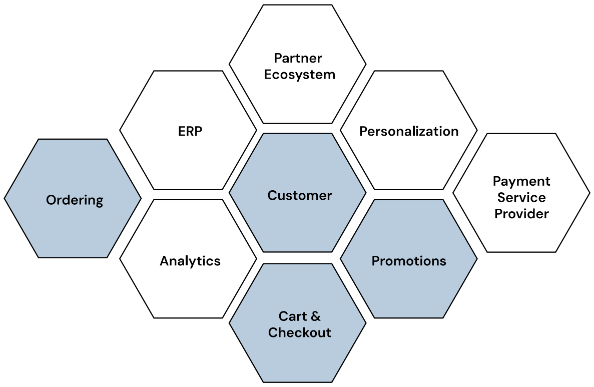
04 *Store Fulfillment*

A store fulfillment request is the same as a request an OMS might send to a Warehouse Management System (WMS). We all agree that a WMS provides many more features than an OMS does as it relates to physical inventory tracking, picking complexity and productivity management. The OMS domain picked up this functional footprint as it became a "one-stop" shop for omnichannel fulfillment. But we also all agree that the same challenges a WMS solves for are now popping up in the store domain. The market already contains standalone vendors offering store fulfillment solutions and the OMS has been integrated numerous times into other packaged SIM applications that provide store fulfillment functionality.

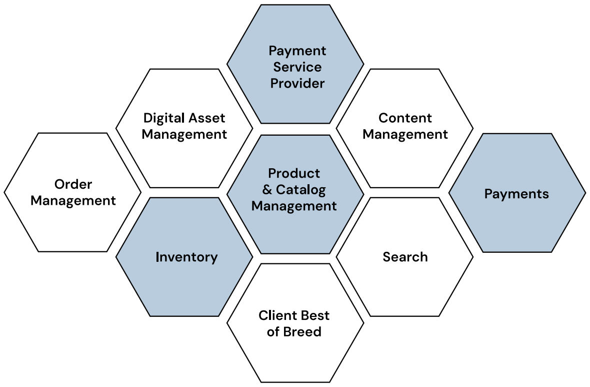
While the commercial data clearly shows the OMS domain is actually four distinct functions bundled together, a significant perception gap remains. Many composable front-end vendors, for example, ironically depict the OMS as a monolithic "back office" entity in their diagrams.

As you can see, the front-end is broken down into 11 microservices, yet **order management** remains *a single, unyielding block*.

And much of the industry still views the OMS domain this way. Perhaps the nature of actual customer commitments via orders and financial transactions being at stake prevents enterprises from viewing the OMS domain as a set of composable services. They view breaking this up as risky. But let's look at the upside being left on the table.



eCommerce Platform Vendor



Advantages of OMS Composability That Businesses Can't Ignore:

01 ***Flexibility:***
Composable commerce allows businesses to select and integrate OMS components that best fit their specific requirements and workflows. This flexibility enables customization and scalability as business needs evolve.

02 ***Modularity:***
By breaking down the OMS into smaller, interchangeable modules, businesses can more easily add, remove, or replace components as needed without disrupting the entire system. This modular approach facilitates agility and innovation.

03 ***Integration:***
Composable commerce encourages the use of APIs and microservices architecture, which streamlines integration with other systems and third-party services. This seamless integration capability is essential for OMS to communicate effectively with other parts of the e-commerce ecosystem, such as inventory management, payment processing, and customer relationship management (CRM) systems.

04 ***Scalability:***
As eCommerce businesses grow, they often need to scale their OMS to handle increasing order volumes and complexity. Composable commerce allows businesses to scale their OMS by adding additional modules or resources incrementally, rather than overhauling the entire system.

05 ***Innovation:***
Composable commerce encourages experimentation and innovation by enabling businesses to easily test new OMS components or features without significant upfront investment or risk. This fosters a culture of continuous improvement and adaptation to changing market dynamics.

If that sounded like it was from ChatGPT—it was. And that's the point. ChatGPT sees all the great content OMS vendors are producing content about composability and if we didn't know better – we would say it's a no brainer. Go composable in OMS!

But hey, we're humans here at Nextuple and we know the promise of composability in the OMS domain is not the reality today. With that – let's discuss what else is in the way.

IT Maturity

Microservice-based architectures **unlock incredible flexibility and scalability,** but they also involve new considerations for your IT environment. Clearly there is additional complexity when a composable microservices-based architecture is introduced. Again—if you want to consult ChatGPT you'd find that maturity is needed across cloud infrastructure, DevOps, security and data management.

Imagine managing five microservices from different SaaS vendors. One important step is creating a testing environment where all these components can work together seamlessly. Having the tools to continuously deploy end-to-end testing with automation is critical—yet most organizations underestimate these requirements.

For organizations looking to deploy on their own clouds we provide ready-made DevOps CI/CD pipelines, Docker images, and other tools that can be used as-is or taken as a baseline for customization.

Tools for Streamlined Development & Deployment:

1

CI/CD Pipeline

Our CI/CD pipeline covers code quality checks, builds, automation regression suites, deployments, and more. For instance, one of our clients was using a different pipeline tool than what we provided, yet they were still able to leverage our tools effectively.

2

Infrastructure Templates

We offer infrastructure templates, such as CloudFormation and Kubernetes scripts, that can be used to set up custom clusters. These templates and scripts are compatible with all major cloud providers. We assisted one client in deploying their infrastructure on GCP, even though our products are primarily deployed on Azure.

3

Monitoring and Automation

We provide Grafana dashboards for monitoring, load testing tools, automation scripts, and integrations/abstractions with various monitoring tools. This comprehensive suite ensures our clients can efficiently monitor and maintain their systems.

We also have a set of libraries/tools to deal with common concerns around encryption, performance, infra cost templates etc. For example, we have an encryption framework that helps you extract PII data.

However, even the most robust foundation needs strong building blocks. While composability offers the flexibility to integrate best-of-breed solutions, true value emerges when these components seamlessly work together and can be extended to meet your specific needs. This is where composability meets extensibility.

Composability with Extensibility

Most OMS vendors sell composability, but they don't deliver on the advantages of it.
Over the last year there has been a marked shift in how OMS vendors are describing their solutions. Nearly all use "composable microservices" in their messaging. At Nextuple we are unaware of any of these vendors (outside of Blue Yonder) that have deployed these microservices at scale to sit alongside other OMS products. Said another way, you don't see a Kibo promising microservice at a Sterling OMS client and you don't see a Manhattan store fulfillment microservice deployed alongside a Fluent Commerce OMS. **Why is that?**

We believe this has played out like this because these microservices were not really designed to be used commercially in this way. The vendors are still trying to optimize on a complete OMS footprint at the client vs creating truly flexible, nimble microservices that can be extended and work seamlessly with other products.

On the feature side we see the following limitations:

Microservices are still packaged business capabilities at a module level vs a functional level.

For example, in most OMS products you cannot separate the ATP function and safety stock function within inventory. These modules are typically packaged to include multiple functions that a client may or may not need all of which may defeat the purpose of going composable. We're working with a client who is running IBM Sterling and looking to deploy our SAVR (Snapshot, Audit, Visibility, and Reconciliation) inventory microservice to improve the audit functionality within inventory. That level of composability is not available at other OMS solutions.

Improve the audit functionality within inventory.

Missing tools for microservices composition.

We give you a foundation to compose multiple microservices together (ours and others) through our Nextuple Integration platform. This platform enables a variety of legacy integration types while also acting as the hub of your point-to-point API integrations. NEIP offers composability because of interoperability with other systems. Now you may wonder why create an integration tool between API points. Because our integration platform (NEIP) connects services through configuration and not code, this reduces the overall development effort and improves the quality of the product because we're not introducing new code.

Streamlining development efforts and improving the quality of the product.

Clients also take on composability for greater flexibility, yet customizations are still at the user exit level and not at the component level.

Taking safety stock as an example, you can change the inputs and outputs of the safety stock calculation but can't add new types of safety stock to drive those calculations. This comes with increased technical skills at the client level but this is where our clients are taking us. They want the sweet spot between building and buying that composability gives them.

Businesses **want more** than user-level customization.

Most products are dictating what user exits and public APIs are available to you vs having the ability to expose new user exits on your own.

This includes making direct calls to the database if required. More advanced clients are looking for the next level of flexibility. Our Flexiplug tool exposes our internal APIs to you as well giving you increased capabilities to customize.

Opening up direct access **for flexibility.**

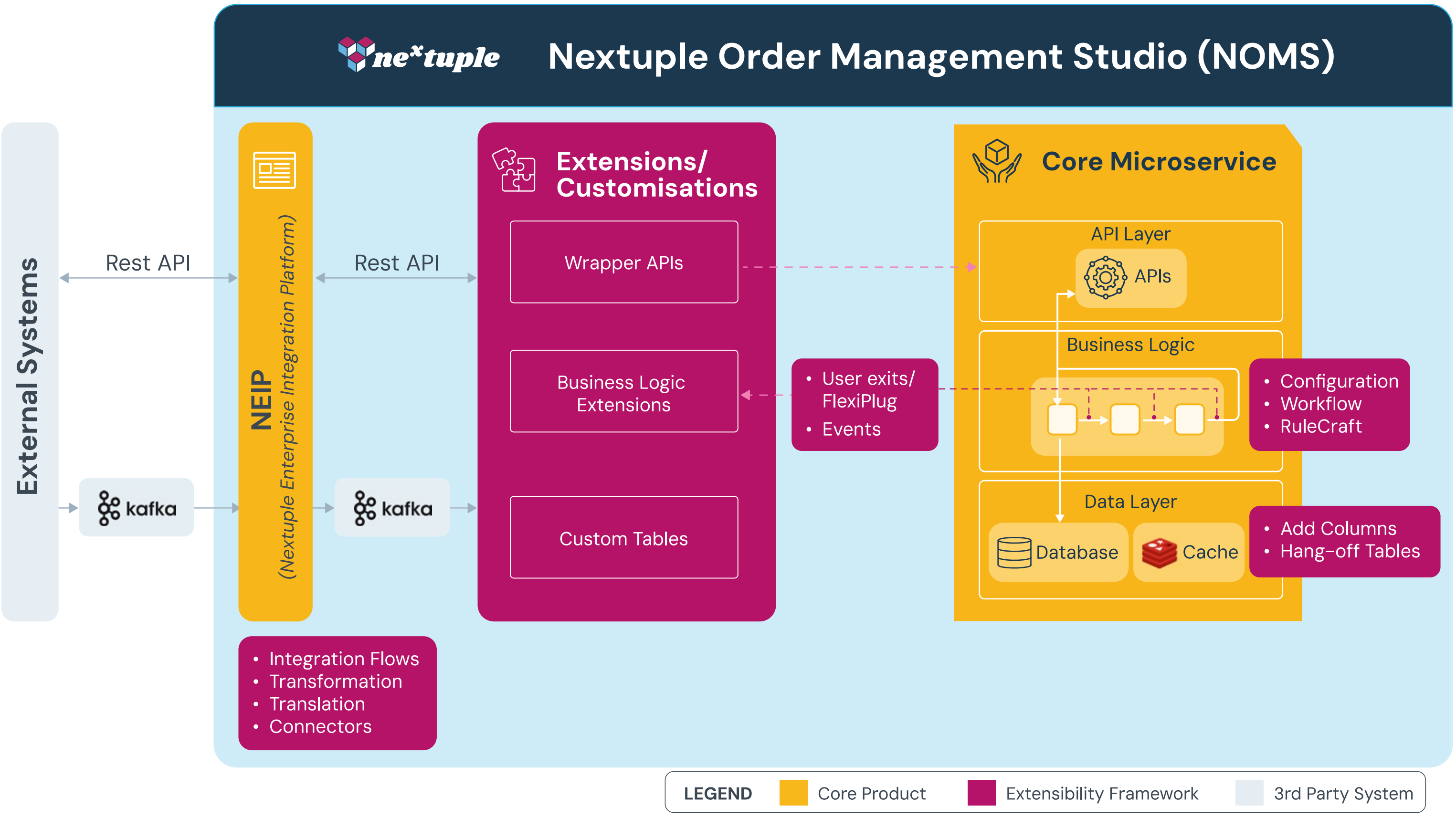
Let's take the following scenario to illustrate the differences.

A retailer is implementing a substitution flow within their BOPIS process. They want a nil pick of an item to trigger an external service which recommends a substitution, then kicks off an external SMS message flow with the consumer to validate or reject the substitution. A typical OMS through public APIs and user exists would allow you to accomplish this. However, the lifecycle of that fulfillment request in the OMS would not be able to be changed. Let's say the release could only be filled or killed in the OMS. In this case, killed because of a nil pick and a new request is generated to the picking application once the substitution is finalized by the consumer.

But in this case, the retailer is looking to EDIT the fulfillment release and create a new subline on the original release to better support the store workflow with the substitution. This is a level of extensibility where the retailer is changing the nature of the how the OMS processes fulfillment requests and something no other OMS software provider gives clients. To take this example further, once this subline is created, we can create new logic to the applied to that subline that doesn't exist in the existing OMS.

Said another way, most products offer "run time" customization through hooks. We provide "build time" flexibility inside the code itself. Legacy OMS platforms use a user exit to enable this but it's still outside the core product. It's composability through integration – our composability is inside the module. But of course, this comes with an increased level of expertise from the client's development team. Your teams have access to change how the code works – but you don't own the code.

A certain set of clients are going to be happy with base level extensibility available today which is through public APIs and vendor defined user exits. We offer that level of flexibility AND the next level which allows access to internal APIs and to create new user exits.



Of course, this level of extensibility involves a degree of customization that also requires the support of single tenant deployments. While some legacy OMS providers do support single tenant deployments on the client's private cloud, they don't come with this increased level of extensibility that our customers are asking for.

This also comes with the need for increased deployment flexibility.

Beyond the flexibility of multi-cloud and Single or multi-tenant deployments, true composability creates value when you deploy something that specifically addresses your issue, and nothing more. In addition, Clients are constantly looking for ways to reduce infrastructure and licensing costs. Yet OMS providers are creating modules that force usage or license costs that are not necessary by bundling solutions that must be deployed together.

The inventory domain is a great example of this.

Recently we supported a client who had previously deployed a V1 of our inventory availability service. Our inventory service is divided into three separate deployable components. Supply and Demand, ATP (Availability) and SAVR (Audit). In this single tenant private cloud deployment, the client had modified functions of the service in V1. V2 of the availability service was now available and the client wished to upgrade it (while keeping their customizations intact). To mitigate the risk, V2 was deployed into production alongside V1 and received copies of the supply and demand updates which were fed into the SAVR service for comparison. Essentially, V2 was in shadow mode and its availability picture could be compared with V1. Once confidence was gained that V2 was working correctly, processing was switched to V2.

Could this be done with a traditional OMS inventory microservice? Technically yes, but since the inventory service is deployable only as a full module, it would have required a separate license to deploy another instance of the inventory service which would have been cost prohibitive. Being able to run the systems in production at the same time is a game changer in the inventory domain where retailers typically see long stabilization periods when attempting to make large changes to inventory processes.

Deployment flexibility translates into greater functional capabilities.

In a recent engagement we worked with a client who had multiple source systems of inventory data via mainframe applications which each talked in a different inventory language. One system tracked inventory via sales variants (SKU), one in a logistics variant and the other by UPC. These systems then all talked to downstream systems in a point-to-point integration architecture, and all ran at different batch cycles. There were over 40 integrations to be dealt with. Meanwhile 40-50MM inventory records travelled through these pipes every day.

There was no application where visibility across all 3 variants existed and typically there were discrepancies between them created by the batch cycles and myriad of issues created by the point-to-point architecture and sheer volume of inventory traffic. In essence, the more "copies" of inventory truth you have, the more sources of failure.

From an IT perspective, a modernization project would need to allow them to replace upstream and downstream systems over a long time frame. From a business perspective, the goal was to create centralized inventory visibility across the three variants and to add deep audit capabilities to resolve issues where those quantities across the variants were off. This would lead to improved accuracy that replenishment and forecasting systems could use.

The goal was to have every inventory update in source systems reflected downstream in the variants that those systems required but still have centralized inventory visibility where you could query on any variant and see real time inventory status.
In addition to functional improvements (mapping logic across variants and audit features, etc.), a flexible deployment was also required.

We deployed three different copies of the supply microservice. One supply service was built for sales variant (SKU), the second was for a logistics variant and the third was for UPC. These instances of the supply microservice were connected to the corresponding downstream solutions. However, the client deployed only one copy of our audit service connected to all three supply microservices.

This audit service provides the business not just with an audit history of transactions in and out but also of its own processing. For example, if system was showing a negative five UPC transaction it would be able to tell you what source system created it and what type of variant created it. A user could now look at the supply across the 3 variants in this one system by using any variant ID to query the audit system. Of course, the audit system was used to greatly improve inventory integrity in the downstream systems. Without this deployment flexibility the client would have had to deploy 3 full inventory modules each equipped with its own full set of features which would have been cost prohibitive. And they would not have a centralized view of the inventory across all 3 variants which was the main goal.

This kind of deployment provides IT and business with a significant lever in reducing risk and cost while on the journey of modernizing capabilities.

Embrace Agility with Composable OMS

Is the OMS really a back office meant for a rigid monolith architecture? Or has its services become as important to sales enablement and profitability as your front-end eCommerce solution?

- **Digital influences 80% of offline sales. Showing your real time inventory positions drives foot traffic to your stores.**
- **EDD's are an expectation of consumers that drive conversion online**
- **Fulfillment optionality is a new conversion lever**
- **Shipping costs are eating profitability and store fulfillment has to scale more efficiently.**

All these reasons and others are forcing retailers to evaluate how they can move faster in this domain. And the OMS vendor landscape has responded with a great deal of "Marketecture" but very little "Architecture" as it relates to composability.

If you've already embraced a composable approach on your front-end, you're well-positioned to lead the back-end revolution. You understand the benefits and the operational discipline required for success.

Composable OMS is not a one-size-fits-all solution. However, for retailers seeking greater speed and agility, innovative options are emerging. Challenge your current provider, explore new entrants like Nextuple, and most importantly, don't settle for the status quo. There's a better way to run your OMS.

Begin your future-proofing journey with composable OMS microservices.

nextuple

We'd Love To Talk To You

www.nextuple.com

You feel the need for speed, but you don't want to bleed for it. We hear you. Come talk to us about a composable, microservice-based approach to omnichannel order management.

Schedule a Demo

Experience the Nextuple difference. Unlock the power of composability and schedule a consultation now.