

Transforming Inventory Visibility for the Omnichannel Consumer

www.nextuple.com



A new approach to the modern demands of inventory visibility.

In a time where the sources of inventory supply (both physical and systemic) seem to be growing, retailers face an increasingly complex challenge: how to provide accurate, real time inventory data higher up in the shopping funnel to meet the demands of today's Omni Channel fulfillment expectations. They are improving but still have a long way to go. [For example today, 59% of brands allow shoppers to filter or refine their selection to show store inventory availability \(up from 50% in 2022\).](#)

Additionally, in today's retail landscape, customers are embracing a wide variety of shopping methods, emphasizing the critical need for accurate and connected inventory systems. Customers demand not only convenience, but consistency in their shopping experiences. [One study](#) shows more than half of customers engage with three to five channels during each journey they take towards making a purchase, underscoring the importance of seamless integration to meet their evolving demands. A recent [McKinsey article](#) points out that while 75 percent of consumers want a seamless omni channel experience, only 25 percent are satisfied with the experience they are getting from retailers.

Estimated Delivery Date (EDD) has emerged as a pivotal factor in customer satisfaction, becoming a make-or-break element for retailers. When it comes to EDD promising, multiple aspects of that promise rely on an accurate and real time inventory foundation. For grocers, the ability to avoid costly substitutions is directly tied to how fast inventory is updated across the business process.

The bottom line is that as more and more fulfillment data is brought upstream in the purchasing funnel, to allow for precise EDD promising and fulfillment choices, the need for real time inventory data continues to become more important.

A scaled, accurate inventory picture is the foundation to OMNI fulfillment excellence.

- EDD promising can't happen without it
- The right sourcing decisions
- Missed online sales
- Substitution avoidance
- Bad CX with cancels or backorders
- Share inventory position with marketplaces
- Make better decisions on sourcing related to inventory age.
- Timely markdowns based excess stock visibility.
- Faster inputs to planning systems.

Yet we find many retailers struggling with how to scale, replicate and ultimately provide accurate availability to promise pictures across the sales channels and to other internal teams and systems.

How did we get here?

Legacy inventory masters such as ERP's or WMS systems were not designed to scale to support the rapid inquiries and commits required of today's OMNI channel commerce experiences. For example, a retailer with 100K+ SKUs attempting to show estimated delivery dates on product list or details pages would require millions of inventory availability calls over a day. Retailers have been building scaffolding around these systems through replicated inventory pictures in the OMS and e-commerce engine. Safety stock calculations and virtual segmentation are additional tools that have been used to protect against inaccuracy or velocity.

But this scaffolding has brought its own set of challenges.

As more and more inventory data is replicated and shared, reconciliations across these inventory pictures have become challenging. **Consider these examples:**



CLICK HERE
to view our infographic

01 **Cloud latency:**

Inventory discrepancies can arise as a result of network latencies at the cloud provider's infrastructure. This can cause inconsistent inventory pictures across multiple shopping sessions which leads to overselling in some instances and underselling in some others.

02 **SKU volumes driving up full sync time windows:**

Many of the synch jobs surrounding inventory load are not designed for high performance and when the SKU volume is high it is common to see inaccurate inventory pictures due to long load time issues.

03 **Complex calculations across systems:**

The more systems that are involved in calculating inventory pictures, the more room for error. If you don't have a clear inventory master each system may calculate ATP differently resulting in a skewed inventory picture.

04 **Poor monitoring of inventory syncs:**

Many of the inventory sync jobs lack proper monitoring on the functional side. For example, an Enterprise Resource Planning (ERP) system is responsible for capturing periodic inventory snapshots and publishing these feeds to the Inventory service. This feed includes inventory statuses of positive counts, zero, and negative numerical values. While the Inventory service handled the processing of positive and zero inventory entries, it did not provide any alerts or acknowledgement for negative inventory instances. This resulted in an overcommitment situation, as the Inventory service continued to maintain positive inventory values for these specific items, despite the presence of negative figures.

05 **Performance issues:**

A common problem is the performance of inventory calls to the source systems. A typical scenario is the e-commerce application relying on a periodically updated inventory cache sourced from the inventory service. The website initiates frequent inventory service calls during various stages of the shopping session. These calls occur during adding items to the cart, progressing through the checkout process, and finalizing at order placement. Challenges arise when the cart sizes expanded significantly or when there were numerous customers placing orders at the same time. The API calls start timing out, leading to disruptions in the order placement or overpromising.

What is the answer?

As a result of these common issues, many retailers are starting to think about an inventory service that can support the modern demands of inventory visibility and management as well as limit the associated challenges with inventory accuracy across the different views.


At Nextuple we designed our inventory master solution specifically with these challenges in mind. We'll discuss our solution across 4 aspects:

1




Purpose Built Design

There are 3 separate microservices that can run independently or as a composite:

 Demand and Supply Service

 ATP Service

 SAVR Service (*Snapshot, Audit, Visibility and Reconciliation*)

2



Functional Features

3



Supporting Architecture

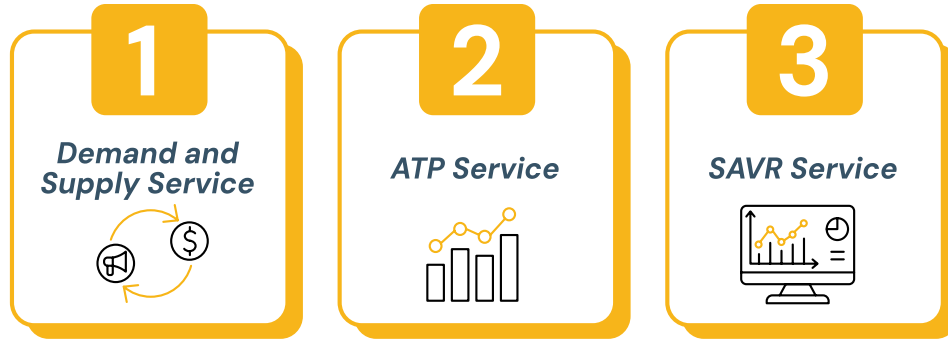
4



Business and technical outcomes enabled

Purpose Built Design

Our inventory solution was designed for true composability. There are 3 separate microservices that can run independently or as a composite.



A key aspect of our solution is decoupling the ATP and demand/supply updates to address performance issues we see with other inventory solutions. In addition to performance benefits, this decoupling allows you to incrementally build towards a full solution. For example, you may only choose to deploy our SAVR service (Audit function) against your existing inventory system or leverage the demand and supply service to feed your existing ATP function.

1

Demand and Supply Service



Designed specifically for fast “writes”. Often 10s of millions of inventory snapshots are required to be synced within an hour.

Raw Supply and Demand Service updates – a service for every supply and demand at the SKU / node / transaction level. We track every single update with the associated message ID and references (such as order #, line ID or PO #). Aggregate Supply Demand – a service that maintains aggregate information of Supply and Demand for items at different levels.

2

ATP Service



Designed for fast “reads” that can happen at peak shopping periods or in “hot SKU” scenarios.

The Available to Promise computes ATP for different availability rules and uses aggregated supply/demand, Safety Stock and Availability rules for computations.

This implementation follows the CQRS (Command and Query Responsibility segregation) design pattern which allows the commands (the writes of supply and demand) to not impact the queries (the reads of an ATP). Now a full synch of inventory data coming into the system will not impact a sales channel’s ability to get a fast response on the ATP query.



The 3rd microservice was purpose built specifically for traceability.

As we shared earlier – the most problematic parts of inventory management come from trying to troubleshoot inventory pictures across systems or tracking down the snapshot of the inventory picture your website or order sourcing engine was looking at when it made an ATP call or node selection.

3 SAVR Service



We designed an extensive audit feature we call the SAVR Service. (Snapshot, Audit, Visibility and Reconciliation) Think of this as a time machine function for your inventory data.

SAVR service provides a unique ability to look up any information related to inventory at any time in the past. This applies to supply, demand, ATP, Safety Stock, and any configuration related to inventory. As we shared above, because of the detailed data captured for each raw update, the SAVR service has in-depth traceability, including the ability to trace back the exact ATP returned for each instance when inventory availability was queried.

A Snapshot is the capability to go back in time and look at supply, demand and availability numbers from the past. The audit feature displays a list of all the transactions that have happened in a timeframe selected. Both of these give you complete visibility to how the inventory picture changed over time down to the raw transactions.

Reconciliation is a comparison capability, where one can look at comparing the supply, demand and availability numbers from any inventory (External or internal) system with those in the inventory management service.

This SAVR service is designed as NoSQL document DB for quick searches and quick retrieval of large amounts of data. This powerful tool can be deployed on its own against your current inventory master for functional reasons or as a way to incrementally deploy this full solution.

Functional Features

The inventory microservice encompasses a wide set of features designed to meet the complex requirements across B2C and B2B sales channels. Highlighted features are shown below.

Inventory Feature List

On-hand Inventory Tracking

- Inventory On-hand
- By Product, Class, Inventory State
- Stores, DCs, FCs, DSVs, MarketPlace
- Infinite Inventory
- Segmentation (Channels, Customers)
- Kits/BOMS

Safety Stock

- SKU/Node level
- Multiple levels (SKU, Dept, Location type, etc.)
- External service; Safety stock feed
- Safety stock override (ignore)
- Dynamic AI/ML Safety stocks

Available to Promise (ATP)

- SKU/Node level
- Network Availability
- Future availability
- ATP w/ global safety stocks
- Expected fill-rate score
- Overstock indicator score
- Geo, SLA based availability

Future Inventory

- Future Inventory (POs, ASNs)
- Pre-Orders

Configuration & Management

- Visibility dashboards
- Safety stocks & thresholds
- Item, node locks

Integrations & Caching

- Node, network level cache
- Distributed caching
- Inventory full/delta syncs
- File processing, batch feed
- Subscribe to OMS demand updates
- Error handling & out of sequence handling
- Published ATP changes (thresholds based)

SAVR Service

- Audit logs of all changes
- Monitoring

Reservations

- Node level, Node Group, Global
- Soft reservations with expiry
- Hard reservations
- By reservation ID
- Batch #, lot #, Serial #



Supporting Architecture

The solution architecture was created by a team of experts who have built and deployed inventory solutions at scale at Blue Yonder, IBM Sterling, and Manhattan and inside retailers such as Lowes, Walmart, and BJ's Wholesale to name a few.

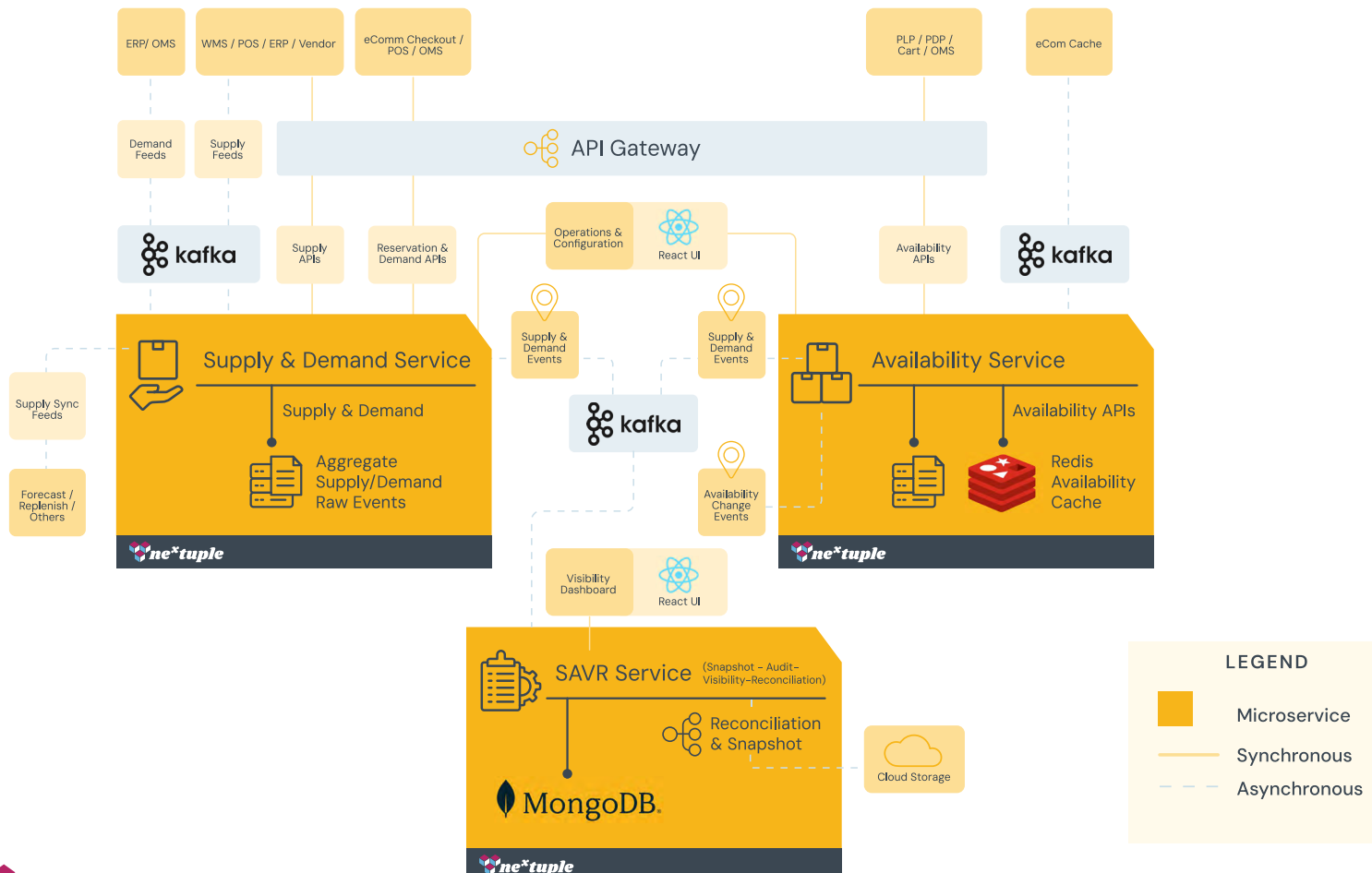
The tech stack components are designed to drive down the cost of infrastructure needed to support the solution. The use of Redis and Postgres for example were chosen to reduce the overall memory footprint of the solution. The architecture is event driven (Leveraging Kafka) to allow for real time inventory and is highly resilient with an active/active distributed architecture for reads and active/passive for writes, cache recovery leveraging persisted event data.

Our two-level decoupling provides many benefits on the performance as well. Because updates to supply table occur asynchronously it reduces system latency.

In addition, allows for parallel processing of supply updates making the system more capable of handling large volumes with no compromises in performance. This translates to better fault tolerance and reliability as well. If an issue arises during supply table updates, it won't directly impact the supply line or other prior updates/computations.

As you would expect, it's cloud native and leverages open-source technologies for improved maintenance.

Inventory Architecture



Business and Technical Outcomes

Based on these design patterns, feature sets and architecture – the Nextuple inventory solution provides benefits for the business and technical teams while executing at industry leading performance levels.

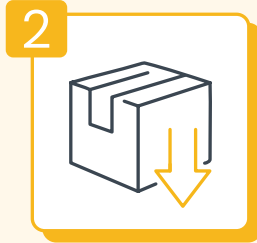


Business Benefits



1 Increase Sales by 1-3%

- 10-20% improved conversion on-line with improved visibility and accuracy
- Improved availability through just in time replenishment



2 Reduce Inventory by 2-3%

- Improved forecasting and replenishment through more accurate and real time data
- Faster decision making on taking mark downs



3 Improved DC and Store Associate Productivity by 2-4%

- 10% reduction in adhoc cycle counts (DC and Store) through improved accuracy and smarter cycle count triggers
- Store fulfillment productivity improvements of 10-20% with fewer no-picks and research activity



4 Improve Inventory Cancellations by 25-50%

- Inventory related cancellations improvements between 25-50% (fill rate and backorder related)
- Reduced cancellations will drive improved CSAT and reduce call center expenses

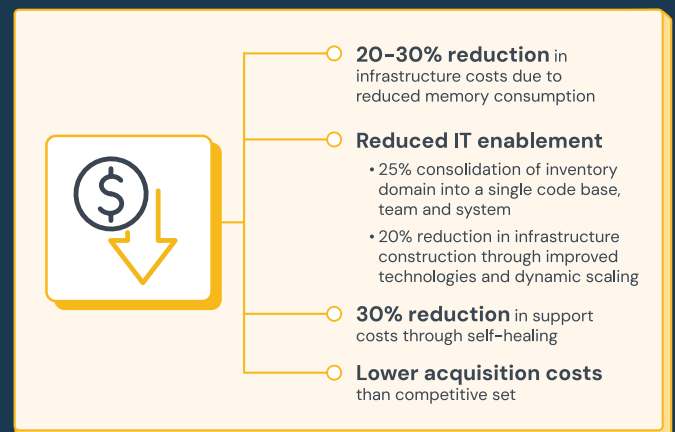
Performance Benchmarks

	TPS	Avg RT (ms)	P95 (ms)
Get ATP	16.1K	4	29
Create Reservation with wait time (100-200 ms)	60	20	27
Full Sync supply	7.4K	6	26

No of instances	Redis	Postgres
10 instances (2 CPU and 4GB RAM per instance)	9 (4 master and 5 slaves). (CPU 500m and 4GB per pod)	256GB SSD cluster, 4 CPU, 8 GB Memory

Cost of Ownership

The inventory solution also delivers significant reductions in the cost of ownership.



The rapidly evolving retail landscape demands a fresh perspective on inventory management. Retailers must adapt to meet the challenges posed by Omni Channel fulfillment while leveraging technology solutions that empower them to thrive in the modern commerce era. By selecting technical solutions that align with their needs, retailers can pave the way for success in the ever-evolving world of retail.

Our inventory master is a perfect example of modern architecture meeting the needs of modern commerce.



We'd Love To Talk To You

www.nextuple.com

OMNI Channel promising is a huge opportunity that requires tooling, operating discipline, lots of data and insights to make it profitable.

[Schedule a Demo](#)

You feel the need for speed, but you don't want to bleed for it. We hear you. Come talk to us about a different approach to OMNI channel promising.

Try Our Promise Engine ROI Calculator

[Try Calculator Now!](#)

Calculate the benefit you could experience with Nextuple's microservices-based Promising Solution.

