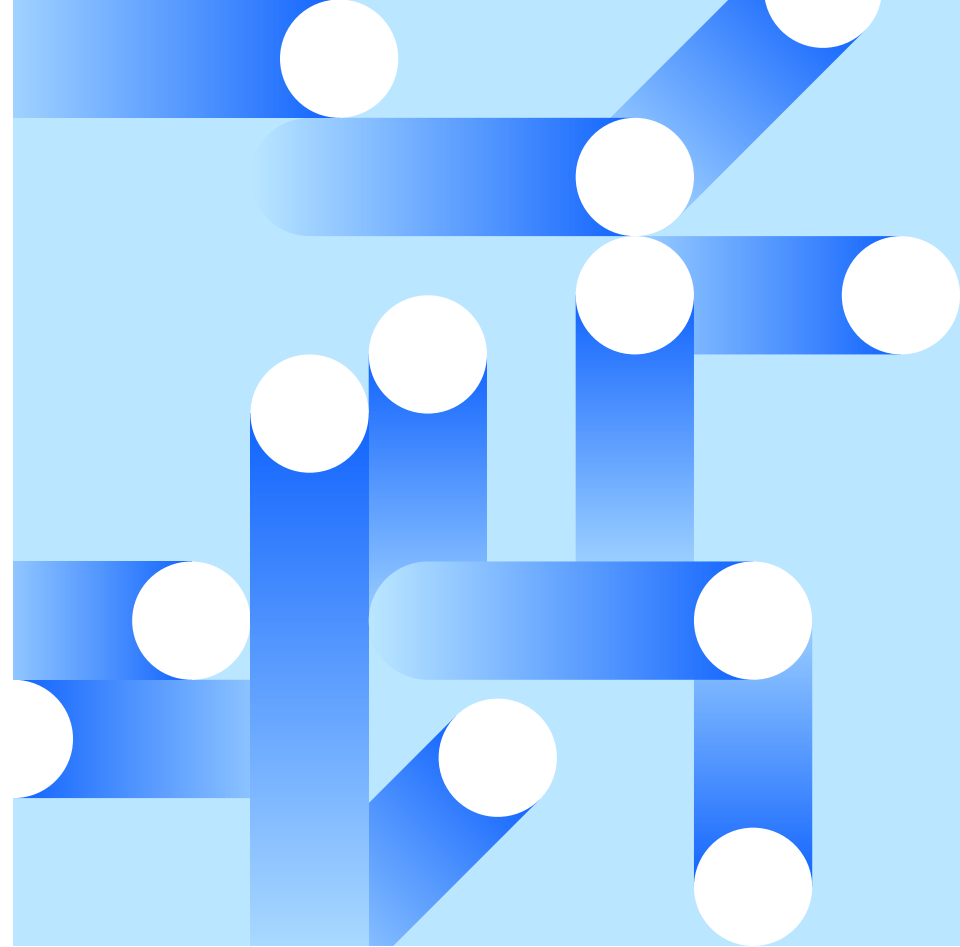


IBM Sterling Order Management

Q1 2023
Demo Series



Demo Agenda

Modernization

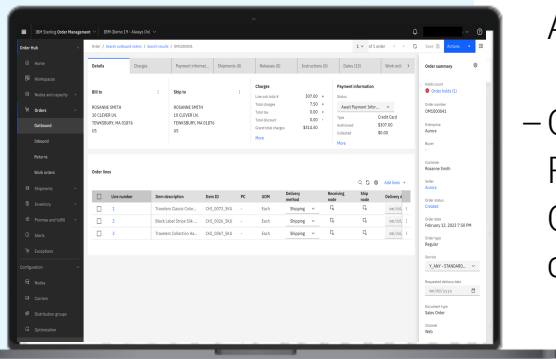
- Introduction to NextGen Call Center
- OMS Foundation alternate OIDC Integration (Okta, ADFS)
- OMS Foundation Prioritize processing of Orders based on dynamic conditions

Commerce Connectivity

- SIP Inventory Sync Transparency (No more a black box)
- SIP Promising based on inventory transfers and dynamic inventory rules
- SIP Inventory Availability and reservations for specific dates

Any Cloud Anywhere

- Order Hub (Next Gen OMS UI) as Containers
- Order Service as Containers



Order Management System

Speakers

Order Management System

Moderator

- Nina Li

Modernization

Call Center

- Mansi Tiwari
- Nisha Kukken

OMS Foundation

- Brian Lima
- Selvakumar Govindarajan
- Bobby Thomas
- Vikas Agrawal

Commerce Connectivity

SIP Inventory

- Christopher Lemay
- Kavita Prasad

SIP Promising

- Tejaswini Ranadive
- Karuna Barla

Any Cloud Anywhere

Order Hub

- Howard Borenstein

Order Service

- Sachin Sethiya



Modernize Business User Tools

- Call Center

Modernization

- Call Center redesigned
- User experience to be modern, intuitive, efficient and user-centered.



Requirements

- Simple, clean and consistent experiences employing modern digital design best practices.
- Content will be easy to find, leveraging search, navigation and wayfinding to meet the unique needs of Call Center users.
- Common tasks will be quick to accomplish, allowing CSRs to move through their queues faster and more effectively.



Solution

- New Call Center with consistent, modern and intuitive interface to be made available in SaaS, Traditional Install, and Containers
- Call Center to support multiple customization options (configuration, differential and override) using Angular and JSON.

OMS Modern UI

Modern & Consistent developer experience across all OMS UI Apps

Modern UI

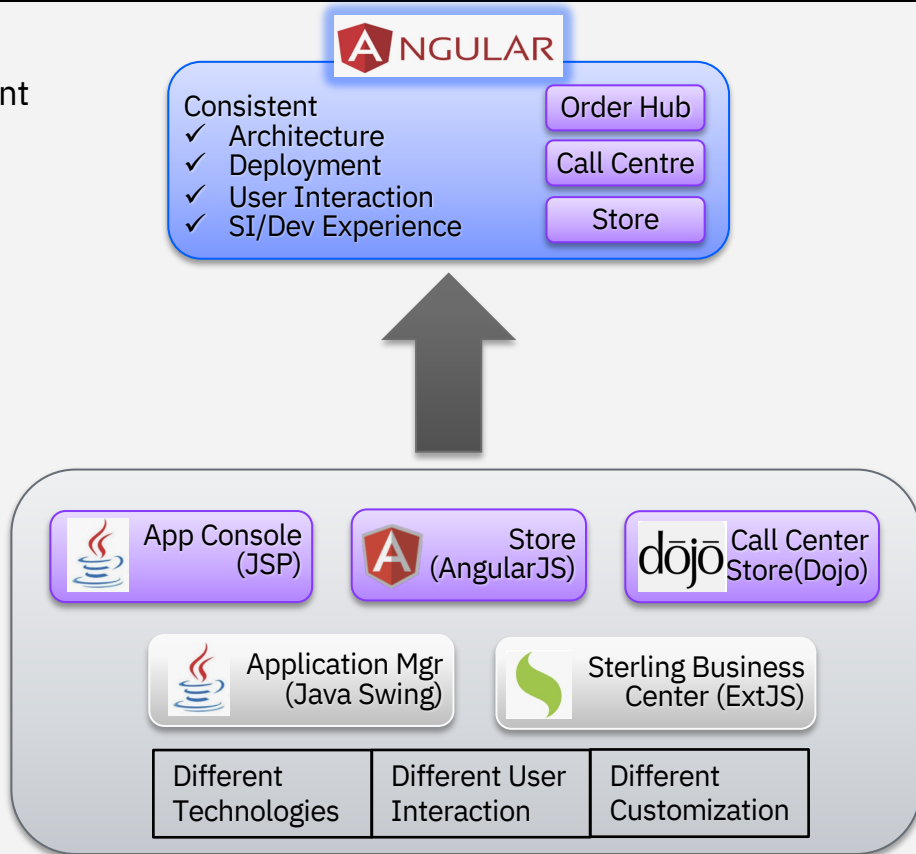
- Order Hub – Business Tooling | App Console Replacement
- Call Center – Efficient custom service
- Store – Mobile first | Increased productivity

Modern Architecture

- All OMS UIs – Order Hub, Store, Call Center - are developed in **Angular** <https://angular.io/>
- UI & Backend are separated – faster deployments
- Micro-frontend architecture - every page is an independent angular app
- Seamless Angular version updates

Consistent UX - Carbon Design System

- Based on IBM's Open-source design system <https://carbondesignsystem.com/>
- Well-researched UX patterns and guidance
- Responsive and ready-to-use widgets





Session 1

Home

Session 1

Home

+

New session

All enterprises

Find an order

For best results, search by order number.

You can also find an order by using the email or phone number of the customer.

Search

[Advanced search](#)

Find a customer

Search for a customer by using their email or phone number.

You can also search by customer name but might get multiple results.

Search

[Advanced search](#)

Find a return

Search by using the return number, if available.

You can also find a return by using the order number or the contact information of the customer.

Search

[Advanced search](#)

Modern UI Design

- Organized, intuitive and easy UI
- Answers & actions right there - View all information on same page
- Improved Productivity - Fewer number of clicks for any task.

Modern UI Architecture

- Built using latest UI tech – Angular (popular framework supported by Google – always up-to-date, always secure)
- Micro-frontend UI
- UI & Backend separately deployed

Easy Customization for SI

- No code** - Configuration based customization
- Low Code** - Differential customization capability
- Drag & Drop** - Carbon UI builder for developing new screens.
- Faster deployment** - Only a minute (as compared to 20 min in Dojo Call Center)

Call Center Review

Handle Every Customer Interaction

Orders:

- **Capture ship/pickup orders**
- **View order history/notes**
- **Track shipments**
- **View invoices**
- **Appease customer (future discount/credit note)**
- **Cancel products**
- **Reship products**
- **Apply/resolve holds**
- **Change fulfillment option**
- **Change address**
- **Apply coupons/ promotions/ discounts/ charges**
- **Add products to an order**
- **Add/remove payment**
- **Price match**
- **Manage service center**

Product:

- **Product Browsing**
- **View in-store availability**
- **View related products – cross-sell/up-sell/ substitutions**

Returns & Exchanges:

- **Create returns**
- **Create returns without an order receipt**
- **Even/uneven exchanges**

Customer:

- **Manage customer profile**
- **Manage addresses**
- **Manage customer classification/notes**
- **Manage customer preferred Payment methods**

Alerts:

- **Exception workflow using configurable alerts**
- **Order alerts**
- **Payment related alerts**
- **Customer alerts**

Customization & Integration:

- **Customize any screen as per business requirement**
- **Integrate with any payment system provider**
- **Open Call Center UI from other CRM**

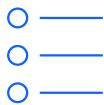




Self Service Enhancements

Modernization

- Allow OMoC Next Gen environments to change their OIDC providers
- Provide developers and organization administrators insight into recurring maintenance schedules



Requirements

- OMS environments support different OIDC providers and on-cloud environments require similar functionality
- OMoC Next Gen environments undergo recurring maintenance such as DB2 and IKS maintenance, the schedule needs to be documented



Solution

- Self Service provides the ability to configure environments to use OIDC providers such as ADFS and okta
- IBM will populate an organization's event calendar with DB2

OIDC Configuration

The new OIDC configuration section allows an environment to be configured to use okta or ADFS

Environment details

Monitoring

Processes

Customizations

Configuration

Certificates

Server configuration

Queue configuration

OIDC configuration

No OIDC provider configured: edit the configuration form to add a provider

Show alternative provider configuration

On

Sync status:

Out of date

Apply changes

OIDC authentication preferences

IBM Sterling Order Management supports configuring environments with your choice of an alternative authentication provider from a list of approved providers that are compliant with OpenId Connect (OIDC). You are encouraged to configure your authentication credentials for every environment. Applying the changes redeploys the environment with your latest customization, which will be shown in the table below. When applying changes, the latest saved OIDC configuration is used.

OIDC provider

Okta

Client ID

abcde

Client secret

abcde

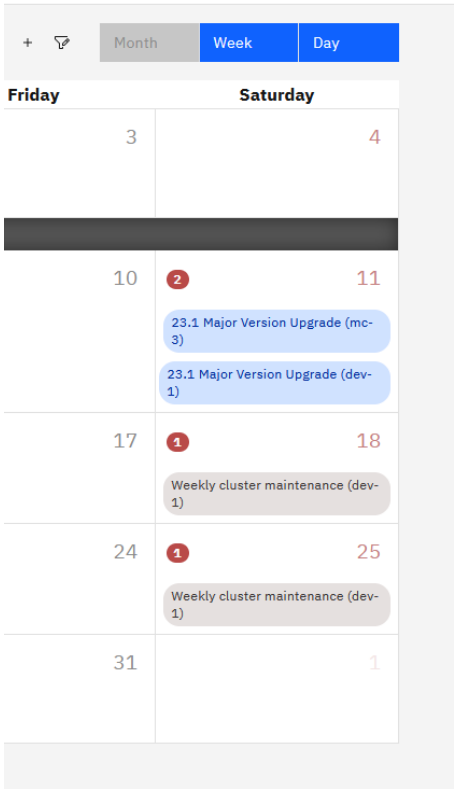
Provider discovery endpoint URL

https://discovery.com

HTTPS URL

Event Calendar

The event calendar will include recurring maintenance events





OMS Foundation

Workload Segregation

Modernization

- Prioritize processing of orders based on dynamic conditions
- Ease of implementation



Requirements

- Provide an easy way to prioritize agent processing
- Ability to define fulfillment workflows based on the size of a sales order
- Optimum utilization of hardware resources allocated for task-q agents



Solution

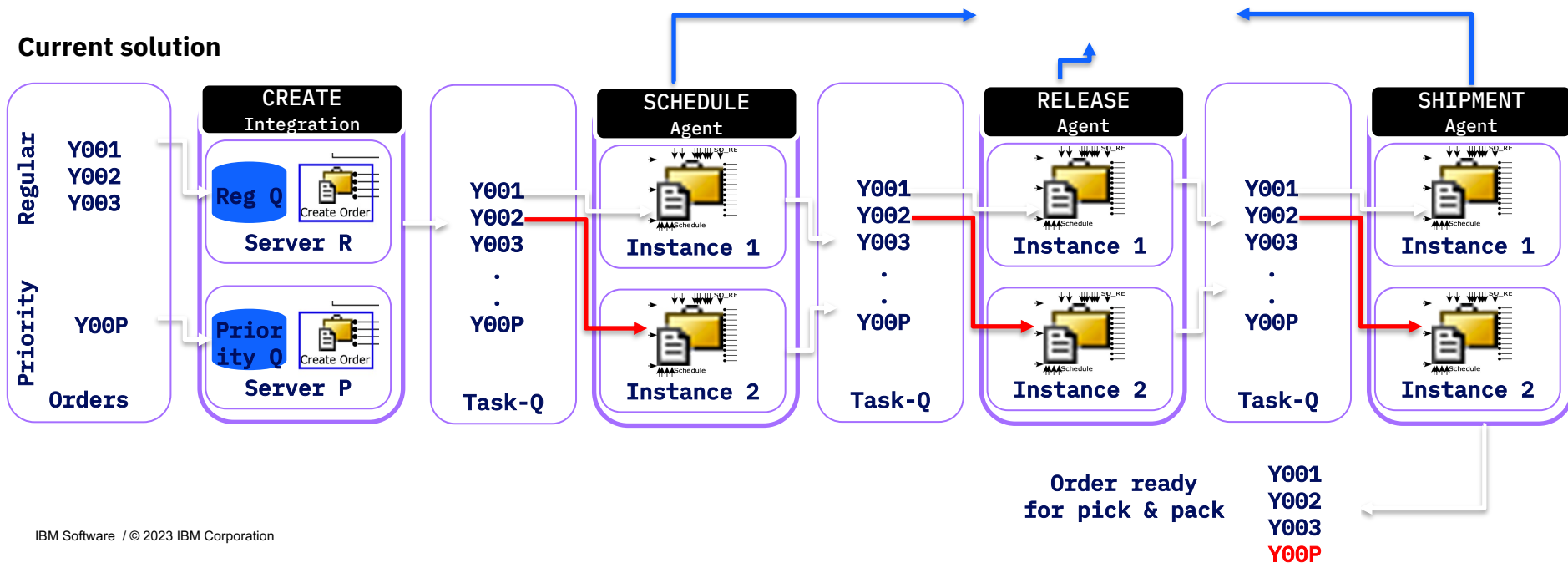
- Workload segregation of task-q agents
- Segregation indicators in condition builder to route the workload

Operational Problem

Aggressive shipping SLA

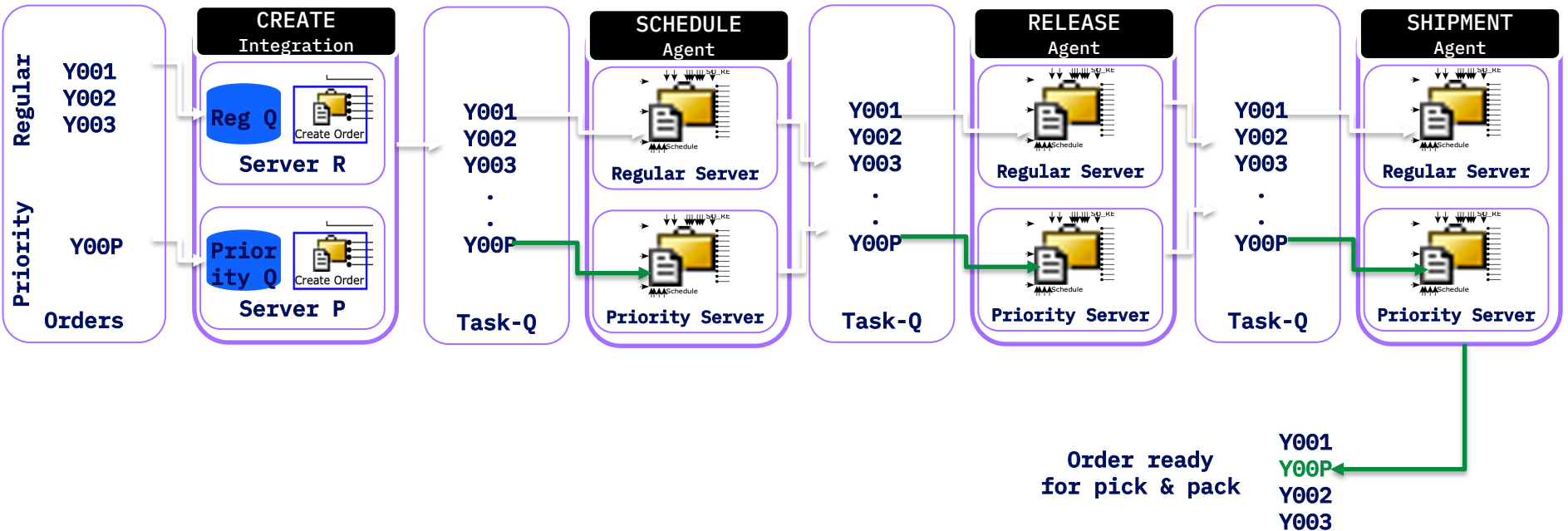
A retailer provides **30-minute express delivery** of medicines with an additional premium. Once the order is received, it must be picked, packed, and made available to the delivery partner **within 5 minutes**. They were not able to **meet this SLA**, as the orders do not get released to the store quickly by OMS agents. They scaled up the schedule, release, etc. agents, but the express delivery **orders wait for its turn** to get released.

Current solution



Operational Problem

With workload segregation



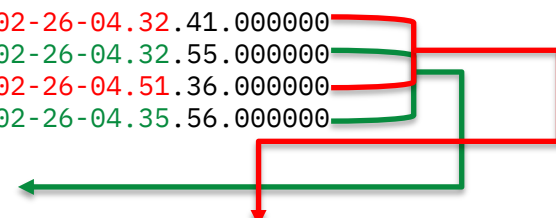
Operational Problem

Sample workload segregation result

OrderHeaderKey	CreateProgid	Status	Createts
202302260432413821637224	CreateStoreOrderServer	1100.001	2023-02-26-04.32.41.000000
202302260432533821652384	HighPriorityPaidSOCreateServer	1100.001	2023-02-26-04.32.55.000000
202302260432413821637224	ConsolidateShipmentRETAIL	3350.001	2023-02-26-04.51.36.000000
202302260432533821652384	HighPrioritySOConsolidateDefault	3350.001	2023-02-26-04.35.56.000000

Order ready for pick and pack in **3 minutes**

Order ready for pick and pack in **19 minutes**



Regular Task-Q Agent Criteria

- Schedule – ScheduleSalesOrderRETAIL
- Release – ReleaseSalesOrderRETAIL
- Consolidate – ConsolidateShipmentRETAIL

Priority Task-Q Agent Criteria

- Schedule – HighPrioritySOScheduleDefault
- Release – HighPrioritySOReleaseDefault
- Consolidate – HighPrioritySOConsolidateDefault

How does workload segregation work?

Task-Q agents can now filter the workload

- Segregation Policy – OrderLineSize, OrderAttribute
ReleaseLineSize, ReleaseAttribute

Properties to define Segregation Policy

```
yfs.taskqueue.segregation.order = orderattribute
```

```
yfs.taskqueue.segregation.order.orderattribute = OrderType
```

```
yfs.taskqueue.segregation.release = releaselinesize
```

```
yfs.taskqueue.segregation.release.releaseattribute =
```

Task-Q Records

YFS_TASK_Q						
SELECT TASK_Q_KEY , TRANSACTION_KEY , DATA_KEY , DATA_TYPE , FILTER Enter a SQL expression to filter results (use Ctrl+Space)						
	ABC TASK_Q_KEY	ABC TRANSACTION_KEY	ABC DATA_KEY	ABC DATA_TYPE	ABC FILTER_CRITERIA	
1	2022042204344826019	SCHEDULE.0001	20_PRE_ORDER39	OrderHeaderKey	Large	
2	2022042204350626031	RELEASE.0001	20_PRE_ORDER39	OrderHeaderKey	Large	
3	2022042705244528209	SCHEDULE.0001	20_PRE_ORDER52	OrderHeaderKey	VeryLarge	
4	2022042705313928249	RELEASE.0001	20_PRE_ORDER52	OrderHeaderKey	VeryLarge	
5	2022042205263526387	RELEASE.0001	20_PRE_ORDER46	OrderHeaderKey	[NULL]	
6	2022042205262526378	SCHEDULE.0001	20_PRE_ORDER46	OrderHeaderKey	[NULL]	

YFS_TASK_Q						
SELECT TASK_Q_KEY , TRANSACTION_KEY , DATA_KEY , DATA_TYPE , FILTER Enter a SQL expression to filter results (use Ctrl+Space)						
	ABC TASK_Q_KEY	ABC TRANSACTION_KEY	ABC DATA_KEY	ABC DATA_TYPE	ABC FILTER_CRITERIA	
1	2022041812060622103	CONSOLIDATE_TO_SHIPMENT	2022041812060522093	OrderReleaseKey	Ent1	
2	2022042711545128807	CONSOLIDATE_TO_SHIPMENT	2022042711544928797	OrderReleaseKey	Large	
3	2022050407214830863	CONSOLIDATE_TO_SHIPMENT	2022050407214630852	OrderReleaseKey	Store	
4	2022041809524121586	CONSOLIDATE_TO_SHIPMENT	2022041809524021572	OrderReleaseKey	VeryLarge	
5	2022041809524421682	CONSOLIDATE_TO_SHIPMENT	2022041809524421676	OrderReleaseKey	[NULL]	

Agent Criteria Configuration

Transaction ID: SCHEDULE.0001 Transaction Name: Schedule

Externally Triggered **Time Triggered** **User Triggered** **Others**

☒ This transaction is time triggered (an agent) Java Class: com.yantra.omp.agent.YFSScheduleOrderAgent

Agent Criteria Definitions

Criteria ID	
SCHEDULE.0001	schedule
scheduleRegular	scheduleOrder_Regular
scheduleOrder_Large	scheduleOrder_Large
scheduleOrder_VeryLarge	scheduleOrder_VeryLarge

Applications Manager

Agent Criteria Details

Criteria ID: scheduleOrder_VeryLarge

Runtime Properties **Criteria Parameters** **Jms Security Properties** **Advanced Scheduling**

Criteria Parameters

Parameter Name	Parameter Value
Action	Get
Collect Pending Jobs	Y
Number of Records To Buffer	0
MaximumRecords	5
Is Base Version	N
Enable Multi-Version Support	N
Colony Id	DEFAULT
Task Queue Filter Criteria	VeryLarge
OptimizationType	
OrderFilter	
ScheduleAndRelease	

Agent Server Level Segregation

Running all the services in an Agent Server for a particular segregation filter only

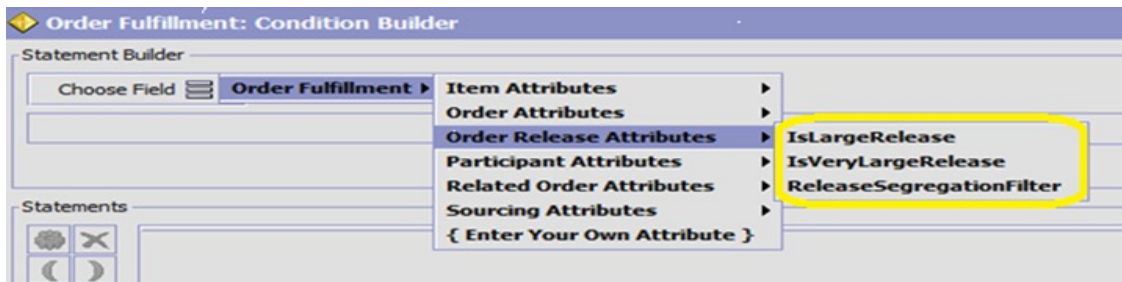
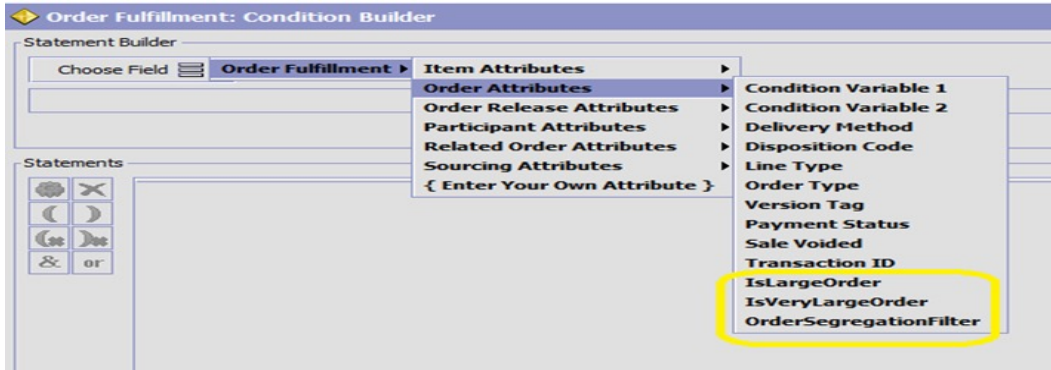
```
./agentServer.sh -jvmargs "-Dfiltercriteria=Large,VOID" scheduleLarge  
OR  
./agentServer.sh scheduleLarge -filtercriteria "Large,VOID"
```

Corresponding SQL Query is

```
SELECT * FROM YFS_TASK_Q YFS_TASK_Q  
WHERE TRANSACTION_KEY = 'SCHEDULE.0001'  
AND AVAILABLE_DATE <= CURRENT_TIMESTAMP  
AND ( FILTER_CRITERIA IS NULL OR FILTER_CRITERIA = 'Large' )  
AND TASK_Q_KEY > '2022050410401131307' AND HOLD_FLAG <> 'Y' ORDER  
BY TRANSACTION_KEY, TASK_Q_KEY, AVAILABLE_DATE
```

Segregation Filter Indicators

Condition Builders



Segregation Filter Indicators

API output / Event data / UE input

```
<Order EnterpriseCode="Ent1" IsLargeOrder="Y" IsVeryLargeOrder="N"  
OrderHeaderKey="20_NEW_ORDER49" OrderNo="20_NEW_ORDER49"  
OrderSegregationFilter="Large"/>
```

```
<OrderRelease EnterpriseCode="Ent1" IsLargeRelease="Y"  
IsVeryLargeRelease="N" OrderHeaderKey="20_NEW_ORDER49"  
ReleaseSegregationFilter="Large"/>
```



Inventory Visibility Supply Sync Transparency

Commerce Connectivity

- Track progress of supply syncs
- Enable customers to manage syncs in batches
- Transparently show status of individual records
- Provide confidence and visibility into supply sync process, thus enabling the customer to understand the status of the process



Requirements

- Transparency feature enabled on a per-tenant basis
- Customer must request a batchId with initial API call, then pass that batchId with subsequent API calls
- Batch will remain in progress until marked complete in final API call

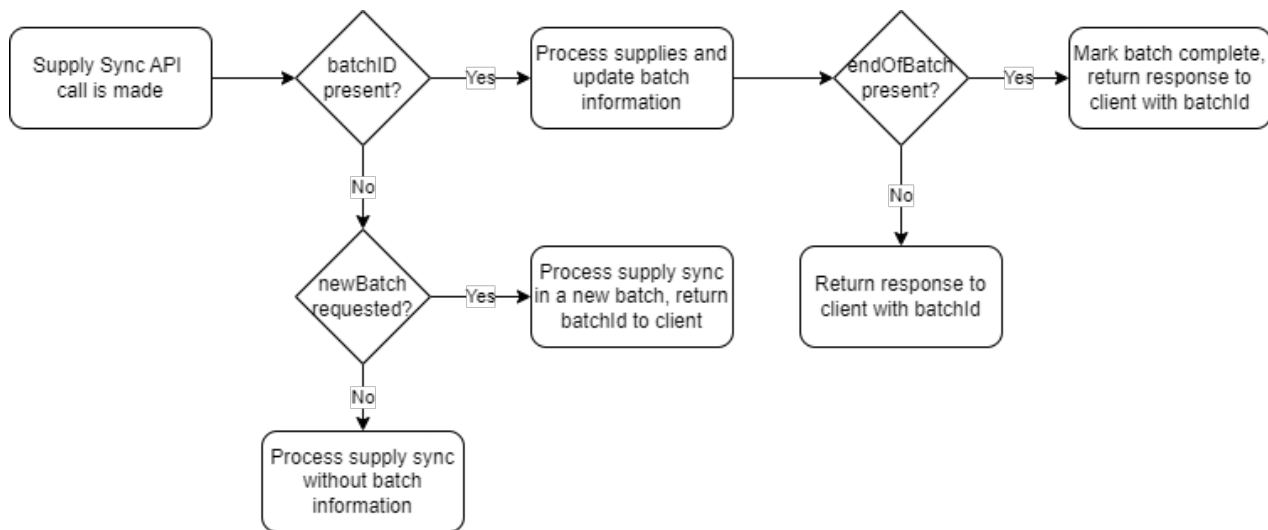


Solution

- Supply sync transparency shows progress of supply sync
- Ensures that customer can reprocess any failed items
- Gives customer confidence that their supply data is being processed
- In addition, a history API is provided, which shows supply history for an item
- Supply Sync process is no longer a “black box” to customers

Requirement

- Customer calls supply sync API to sync supply, requesting a new batchId
- Customer can then view the status of the supply objects sent in that batch
- By adding batchId to subsequent requests, the batch increases in size
- Customer can search by status to see if any records have failed
- Once batch is marked complete, customer is assured that all the supply sync data has been processed

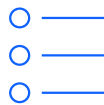


Improve Shopper Experiences and business Profitability



Commerce Connectivity

- Enable fulfillment managers to manage inventory safety stock level dynamically
- Enrich the shopper experience offering accurate ShipToStore promises
- Enable fulfillment managers to dynamically manage the business constraints for customer promises



Requirements

- A shopper can see Store pickup options even if inventory is not available at the location and inventory can be transferred from vendor or alternate location
- SIP Customers should be able to dynamically manage the inventory rules for Safety Stock or for inventory transfers



Solution

- SIP Platform provides rules framework capable of supporting dynamic conditions and results, all at scale
- Core Promising large network graph algorithms are enhanced to support varying optimization and constraint strategies

Promise based on Dynamic Safety Stock Rules

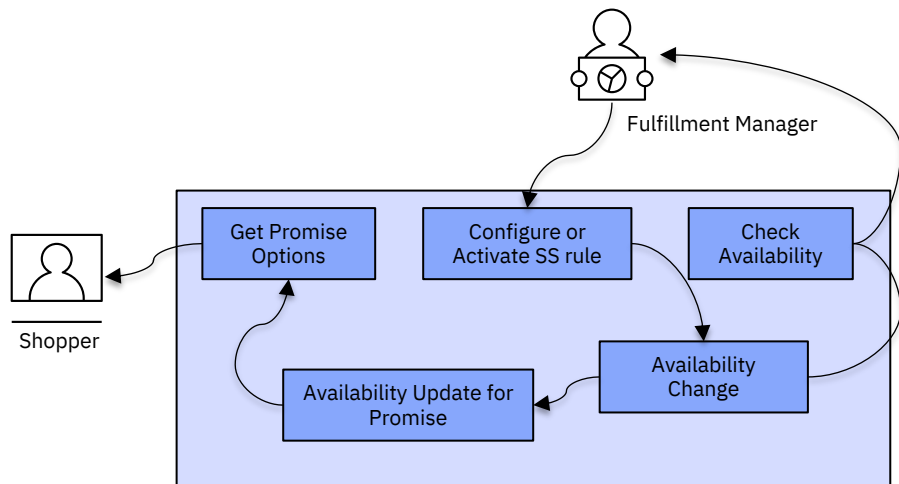
Inventory is complex and not always accurate, so the business needs tools like Safety Stock to manage the inventory thresholds

Safety Stock can be configured based on dynamic conditions and managed at a Node or Network Level, either as an absolute value or as a percentage of entire available inventory

Safety Stock rules take effect real time into promise decisions without having to wait for any availability broadcast out of the platform

Customer experience is maintained throughout the browse experience

Leverage the Rule framework to flexibly configure and manage safety stock rules



Ship To Store - Inventory Transfers

Store Pickup accounts for one of the most important experiences on any order capture channel

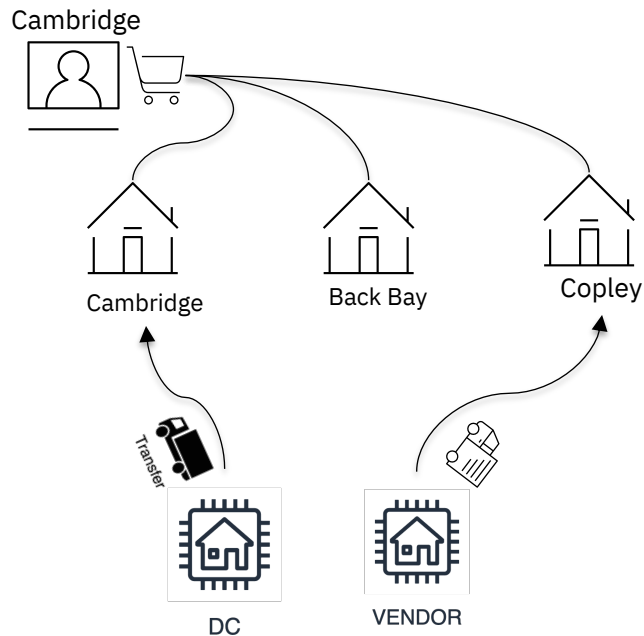
Inventory is not always available at the physical location, but can be transferred from alternate location

Alternate locations can be either in network DC's or external vendors (for items sold directly by vendors)

Platform can account for complex transfer configurations and specific carriers responsible for the inventory transfers

Promises can be made to the shopper for Ship To Store pickups

Flexible transfer rules can be created based on the business operations



Accurate Promise while being Profitable

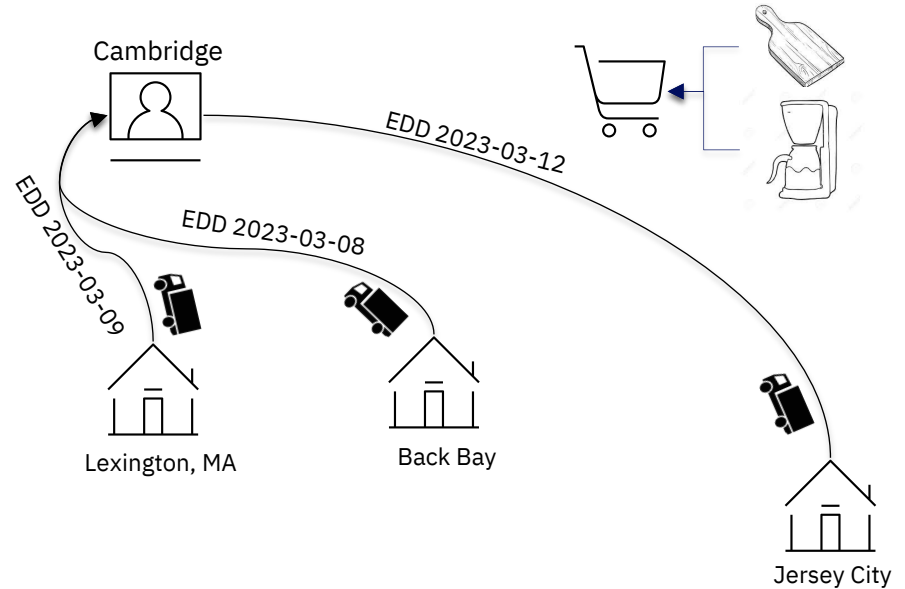
Fulfillment manager should be able to choose between different fulfillment optimization strategies

Enhanced shopper experience based on flexible API options – to either optimize based on earliest delivery or minimizing number of shipments while maintaining service levels

Entire network can be evaluated real time for promise, thus opening endless possibilities of exposing inventory without affecting profitability on the sale

Optimization strategy can be dynamically leveraged to offer more personalized experiences to specific premium shoppers or high value orders

Either two packages can go out from nearby Stores or single package from a farther Store





Availability By Date

Commerce Connectivity

- Ability to tell availability by date or date range
- Allows fulfilment manager to correctly determine the total quantity available for a specific time period and automatically omit unsellable expired inventory.
- Enable customers to pre book future inventory.



Requirements

- New deployments will have the capability available by default
- Existing customers would be migrated from old aggregated availability to this new availability by date based on their business requirements.
- Customers can leverage the new API and event formats.



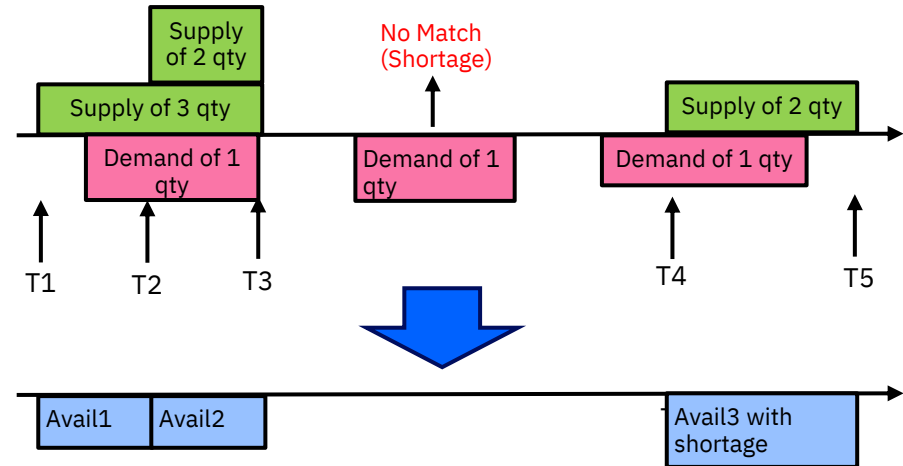
Solution

- Reservations can be placed more precisely against future inventory.
- Network level reservations are propagated to specific nodes based on node priority.
- Availability APIs are provided for both node and network level.
- More granular availability is shown with respect to dates.
- Customer can peek into any future date window to know the availability considering Onhand and POs at that time.

Availability By Date (V2 Availability)

An availability V2 record shows:

- The total available to sell quantity for a given time period for a requested item and accounting
 - Safety stock
 - shortage
 - fulfillment option
 - Availability Type (SELL, SCHEDULE, RELEASE)
- Availability = Supply – (Demand or Reservation)
- Example – Inventory snapshot of ‘Center table’ for a furniture store



Availability By Date (V2 Availability)

Feature Name: V2 Availability

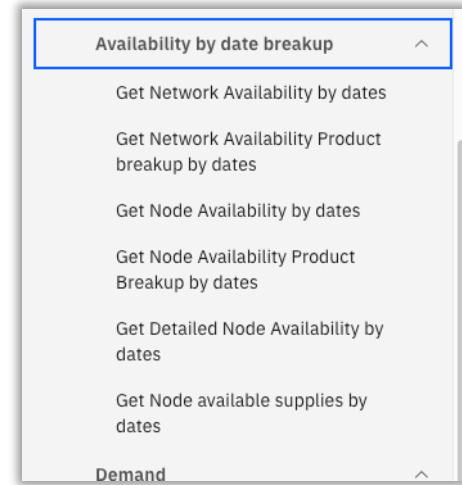
V2 Availability APIs

API Documentation

<https://developer.ibm.com/apis/catalog/inventoryvis--inventory-visibility/Introduction/>

KC link

<https://www.ibm.com/docs/en/inventory-visibility?topic=data-availability-by-date>



V2 Availability events

Feature Name: V2 Events

When user migrates to V2 availability model, they must update their event consumer to understand concepts of availability by date.

The following event ID are there in V2:

The V2 format details are defined in:

<https://www.ibm.com/docs/en/inventory-visibility?topic=formats-new-event>

IBM Sterling Inventory Visibility

IBM Sterling Inventory Visibility /

New event formats

Last Updated: 2022-12-09

IBM Sterling Inventory Visibility has introduced a new event format. Any new events going forward will be delivered in the new format. Existing events will continue to be delivered in older format which is now being referred to as legacy format. Eventually all older events will be migrated to the new format; but that will be done in future and appropriate notice will be sent to all the users.

The following table lists events delivered in newer format.

Event ID in Sterling Inventory Visibility	Event code in Payload
ProductAvailabilityToSell.DistributionGroupSnapshot	dgAvailabilitySnapshot
productAvailability.v2	productAvailability.v2
dgAvailabilityChange.v2	dgAvailabilityChange.v2
productAvailabilitySnapshot.v2	productAvailabilitySnapshot.v2
dgAvailabilitySnapshot.v2	dgAvailabilitySnapshot.v2
Demand.Change	demandChange

V2 Reservation

Feature Name: Availability by date : Reservation

Reservation at Node:

- The V2 reservation API enables inventory to be booked against a specific period
 - `requestedReservationTs <==> requestedEndTs`
- This enables future inventory to be blocked without affecting onhand inventory

Reservation at Network:

- Network level reservations are propagated to specific nodes based on the node priority.
- If multiple nodes of a DG are having sufficient inventory to fulfill the incoming reservation, then the highest priority node is selected.

API Documentation-

<https://developer.ibm.com/apis/catalog/inventoryvis--inventory-visibility/api/API--inventoryvis--ibm-sterling-inventory-visibility-apis#post868743680>

Reservation V2

Create v2 Reservations

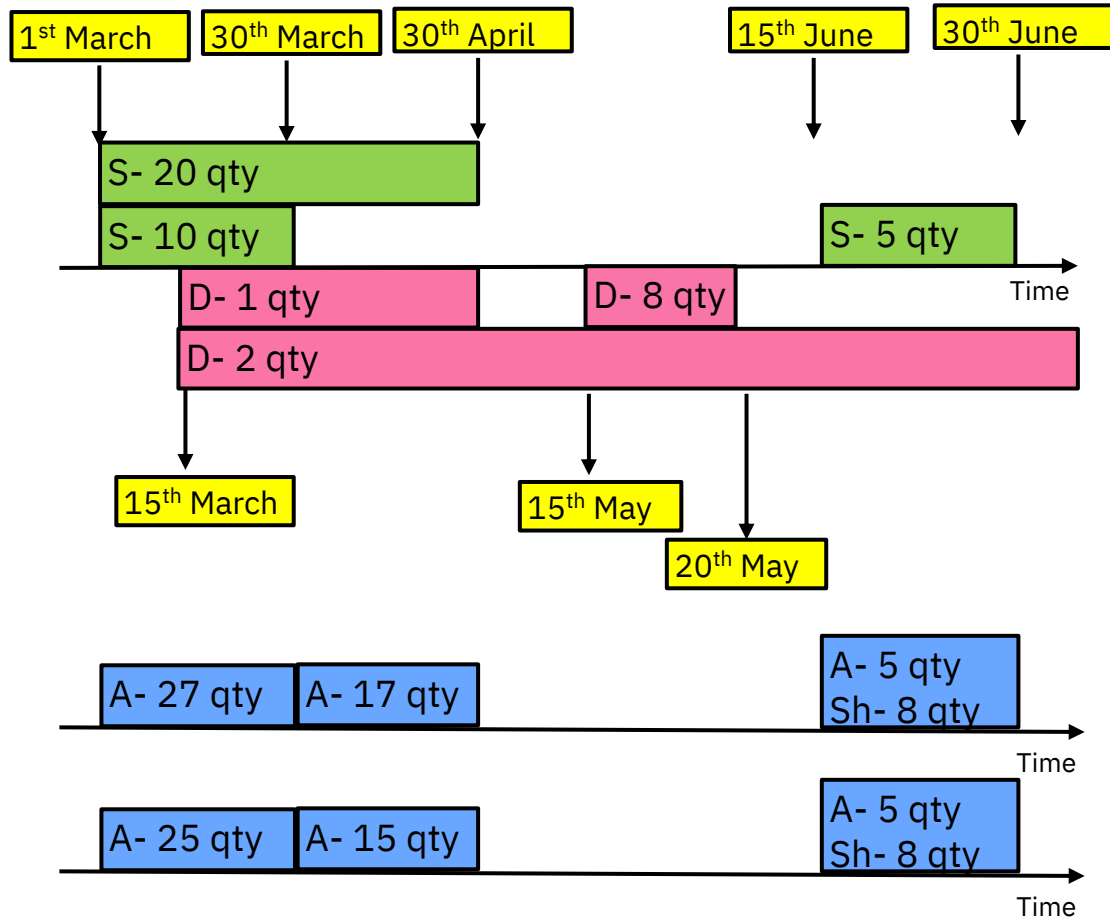
Get Reservations

Delete Reservations

Update Reservation

Demo Scenario

Stores – IV_Store1 (Priority 2),
IV_Store2 (Priority 1)
Distribution Group – IV_STORE_DG



Reservation – 2 qty for 17th April

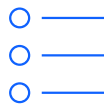


Modernize Business User Tools

- **Order Hub**
- **Store**

Any Cloud Anywhere

- User interface for fulfillment and order management professionals to manage their fulfillment network
- Translate business goals into actions with an intuitive interface, contextual data, and key performance metrics



Requirements

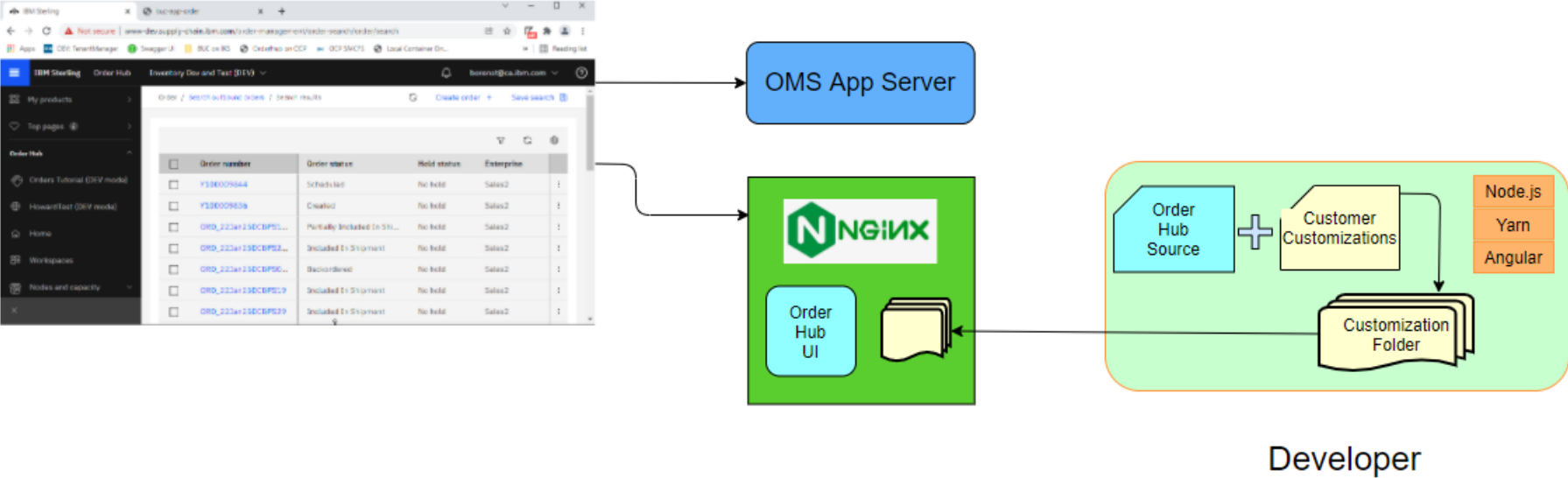
- Optimize the flows in the business user tooling
- Business User Tooling must work on all modern web browsers
- Customizations should be developed using modern UI development technologies



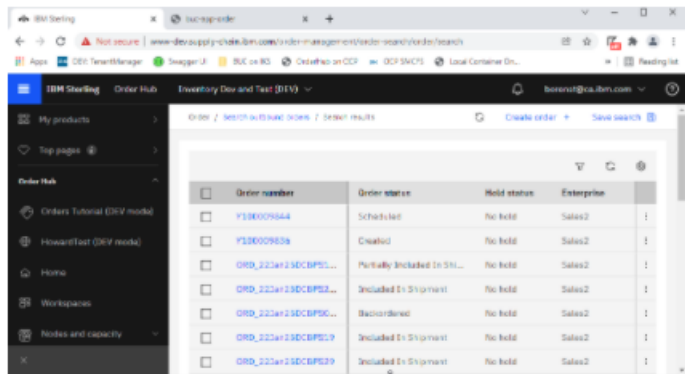
Solution

- Order Hub delivers a consistent, modern and intuitive interface that is supported on Chrome, Firefox and Safari
- Order Hub is available in SaaS, Traditional Install, and Containers
- Workspaces and node metrics
- Order Hub supports multiple customization options (configuration, differential and override) using Angular and JSON.

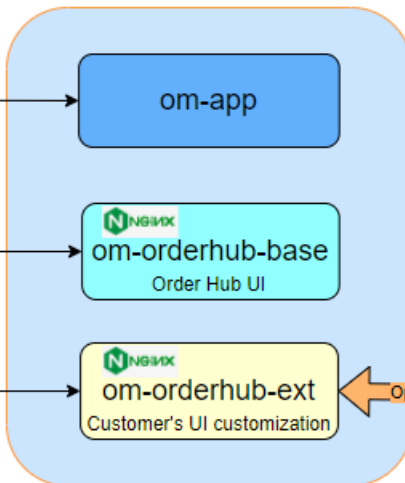
Order Hub - Traditional



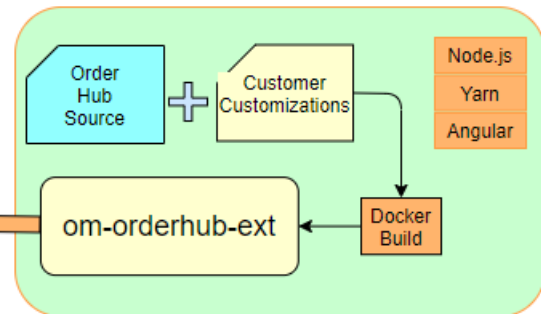
Order Hub - Containers



Order number	Order status	Hold status	Enterprise
P138009844	Scheduled	No hold	Sales2
P138009838	Created	No hold	Sales2
ORD_2234723DCBPSC...	Partially Included In Sh...	No hold	Sales2
ORD_2234723DCBPSC2...	Included In Shipment	No hold	Sales2
ORD_2234723DCBPSC...	Backordered	No hold	Sales2
ORD_2234723DCBPSC9	Included In Shipment	No hold	Sales2
ORD_2234723DCBPSC9	Included In Shipment	No hold	Sales2



RHOCP/Kubernetes



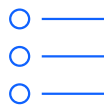
Developer - DTK



Order Service

Any Cloud Anywhere

- Single order repository for all channel orders
- Deploy on any Cloud



Requirements

- Independent deployable component for Order data
- Provides flexible and scalable Interfaces for data ingest and data retrieval
- Ability to do advanced searches on the Order data
- Retrieve Order data without any dependency on OMS



Solution

- A new Order microservice, **integrated** with Sterling Order Management system
- **Order Search** supports lookup of active and history orders
- **Order Archive** allows OMS to offload historical orders

Available as Containers for deployment anywhere

Detailed Webinar and deep dive session done on Jan 19th

<https://ibm.co/3mrxjjB>

References and Links

Modernization

OIDC <https://www.ibm.com/docs/en/order-management-sw/10.0?topic=software-extending-application>

OMS Foundation Workloads Concepts <https://www.ibm.com/docs/en/order-management-sw/10.0?topic=reference-workload-segregation-task-queue-agents>

- Scenarios <https://www.ibm.com/docs/en/order-management-sw/10.0?topic=agents-scenarios-using-segregation>
- Task-Q Agent Criteria (Sample) <https://www.ibm.com/docs/en/order-management-sw/10.0?topic=transactions-schedule>

Commerce Connectivity

Sterling Intelligent Promising <https://www.ibm.com/docs/en/intelligent-promising?topic=overview>

APIs <https://developer.ibm.com/apis/catalog/inventoryvis--ibm-sterling-intelligent-promising-apis/>

Any Cloud Anywhere

Order Hub <https://www.ibm.com/docs/en/order-management?topic=hub-order-overview>

Order Service <https://www.ibm.com/docs/en/order-management-sw/10.0?topic=features-order-service>

