Oracle® Retail Enterprise Inventory Cloud Service

Inbound and Outbound Integration Guide

Release 24.1.401.0
G17614-01
October 2024

ORACLE

Oracle Retail Enterprise Inventory Cloud Service Inbound and Outbound Integration Guide, Release 24.1.401.0

G17614-01

Contents

Send Us Your Comments

Preface

Audience xvii
Documentation Accessibility xvii
Related Documents xvii
Improved Process for Oracle Retail Documentation Corrections xvii
Oracle Retail Documentation at the Oracle Help Center xviii
Conventions xviii

1 Introduction

2 Retail Home Integration

Launch SIOCS from Retail Home 2-1
Tile Reports 2-3
EICS Endpoints 2-3
    Shop Floor Out of Stock Items 2-3
    Stock Counts - Ready to Authorize 2-4
SIOCS Operational Views 2-4
    Launch SIOCS Operational Views from Tile Report 2-5
Subscription Usage Batch 2-5

3 Retail Integration Cloud Service (RICS) - based Integration

Security Considerations 3-1
Customer Orders 3-2
Picking 3-2
Deliveries 3-2
Reverse Picking 3-2
Multi Leg 3-3
RIB Payloads 3-3

Purchase Orders and Vendor Deliveries 3-3
Inventory Adjustments 3-4
Items 3-5
Stock Counts 3-5
Transfers 3-5
Transfer Creation 3-5
Transfer Messages 3-6
Transfer Shipment Creation 3-6
Transfer Receiving 3-6
Transfer Doc 3-9
Transfer Shipment 3-10
Transfer Receiving 3-10
Vendor Return 3-10
RTV Creation 3-10
RTV Shipment 3-10

4 REST Web Services

REST WEB Services Security Considerations 4-1
REST WEB Services Basic Design Principles 4-4
Requests and Responses 4-4
API Versioning 4-4
Content-Type 4-4
JSON Validation 4-5
Synchronous vs Asynchronous 4-5
Online Documentation 4-5
Configured System Options In EICS 4-6
External vs Internal Attributes 4-7
Dates In Content 4-7
Links In Content 4-8
Hypertext Transfer Protocol Status Codes 4-8
Success Codes 4-8
Client Failure Codes 4-8
System Failure Codes 4-9
JSON Error Element and Error Codes 4-9
Schema — ErrorList 4-9
Example Value 4-10
Error Attribute Definitions 4-10
Integration Error Codes 4-11
Error Code Data Elements 4-11
REST Service: Activity Lock 4-12
Service Base URL 4-12

API Definitions 4-12
API: Find Lock 4-12
API: Create Lock 4-15
API: Delete Lock 4-16
API: Read Lock 4-17
REST Notification 4-18
Service Base URL 4-18
APIs 4-18
API: findNotifications 4-18
API: createNotifications 4-19
Index 4-20
REST Service: Address 4-21
Service Base URL 4-21
API Definitions 4-22
API: Import Address 4-22
API: Delete Address 4-23
API: Read Address 4-24
REST Service Batch 4-25
Service Base URL 4-25
API Definitions 4-26
API: Execute Batch 4-26
API: Find Batch Jobs 4-26
REST Service: Differentiator 4-28
Service Base URL 4-28
API Definitions 4-28
API: Import Differentiator Types 4-28
API: Import Differentiators 4-29
API: Delete Differentiator Type 4-30
REST Service: Finisher 4-30
Service Base URL 4-30
API Definitions 4-30
API: Import Finishers 4-31
API: Delete Finisher 4-32
API: Import Items 4-32
API: Remove Items 4-33
REST Service: Inventory Adjustment 4-34
Service Base URL 4-34
APIs 4-34
API: Find Adjustments 4-35
API: Read Adjustment 4-36
API: Update Adjustment 4-37
API: Confirm Adjustment 4-39

API: Cancel Adjustment 4-39
API: Create Adjustment 4-39
REST Service: Item 4-41
Service Base URL 4-41
API Definitions 4-42
API: Import Items 4-42
API: Remove Items 4-44
API: Import Hierarchies 4-45
API: Remove Hierarchies 4-46
API: Import Related Items 4-47
API: Delete Related Items 4-48
API: Import Image Urls 4-48
API: Delete Image Urls 4-49
REST Service: Item Inquiry 4-51
Service Base URL 4-51
API Definitions 4-51
API: Find Item by Search Scan 4-52
API: Find Item by Source 4-53
API: Find Items 4-54
API: Find Items by Scan 4-58
API: Find Items by Inventory 4-60
API: Find Related Items 4-63
API:Find Items by UDA 4-65
REST Service: Item Inventory 4-68
Service Base URL 4-68
API Definitions 4-68
API: Find Available Inventory 4-68
API: Find Inventory 4-71
API: Find Expanded Inventory 4-74
API: Find Future Inventory 4-78
API: Find Inventory in Buddy Stores 4-79
API: Find Inventory in Transfer Zone Stores 4-83
REST Service: Item ISN 4-86
APIs 4-86
API: Create ISN 4-86
API: Update ISN 4-87
API: Delete ISN 4-88
API: Find ISNs 4-89
API: Read ISN Types 4-90
REST Service: Item Price 4-90
Service Base URL 4-90
API Definitions 4-90

API: findPrices 4-90
API: findPriceByIds 4-91
REST Service: Item UDA 4-94
Service Base URL 4-94
API Definitions 4-94
API: Import Udas 4-94
API: Delete Udas 4-95
API: Import Item Udas 4-96
API: Delete Item Udas 4-97
API: Find Item Udas 4-97
REST Service: Item UIN 4-99
Service Base URL 4-99
API Definitions 4-99
API: createUin 4-99
API: readUin 4-100
API: findUins 4-101
API: findUinLabels 4-102
API: findUinHistory 4-102
API: generateUins 4-103
REST Service: Manifest 4-105
Service Base URL 4-105
API Definitions 4-105
API: Close Manifest 4-105
REST Service: POS Transaction 4-106
Service Base URL 4-106
API Definitions 4-107
API: Import POS Transactions 4-107
REST Product Group 4-110
Service Base URL 4-110
APIs 4-110
API: findProductGroup 4-111
API: readProductGroup 4-112
API: createProductGroup 4-115
API: updateProductGroup 4-125
API Basics 4-125
Input Data Definition 4-126
API: cancelProductGroup 4-130
Index 4-130
Rest Service: Security 4-132
Service Base URL 4-132
API Definitions 4-132
API: Import Users 4-132

API: Delete Users 4-134
REST Service: Reason Code 4-135
Service Base URL 4-135
API Definitions 4-136
API: Find Adjustment Reason Codes 4-136
API: Find Shipment Reason Codes 4-137
Rest Shipping 4-139
Service Base URL 4-139
APIs 4-139
API: findCarriers 4-140
API: findCarrierServices 4-140
API: findCartonSizes 4-141
API: findWeightUoms 4-141
API: findMotives 4-142
Index 4-142
REST Service: Stock Count 4-143
Service Base URL 4-143
API Definitions 4-143
API: Snapshot Stock Count 4-143
REST Service: Store 4-143
Service Base URL 4-144
API Definitions 4-144
API: Import Stores 4-144
API: Delete Store 4-146
API: Read Store 4-146
API: Find Stores 4-148
API: Find Associated Stores 4-150
API: Find Auto Receive Stores 4-152
API: Find Transfer Zone Stores 4-153
REST Service: Store Item 4-155
Service Base URL 4-155
API Definitions 4-155
API: Import Store Items 4-155
API: Remove Store Items 4-157
API: Import Replenishment Items 4-158
API: Remove Replenishment Items 4-159
REST Service: Store Order 4-160
APIs 4-160
API: Read Order 4-161
API Find Orders 4-162
API: Approve Order 4-163
API: Cancel Order 4-164

Additional Data Definitions 4-164
REST Service: Store Sequencing 4-164
APIs 4-165
API: Find Sequence Areas 4-165
API: Read Sequence Area 4-166
API: Create Sequence Area 4-168
API: Update Sequence Area 4-169
API: Delete Sequence Area 4-171
API: Create Sequence Item 4-171
API: Update Sequence Item 4-172
API: Delete Sequence Item 4-174
Additional Data Definitions 4-174
REST Service: Supplier 4-175
Service Base URL 4-175
API Definitions 4-175
API: Import Suppliers 4-175
API: Delete Supplier 4-177
API: Import Items 4-178
API: Delete Items 4-179
API: Import Item UOMs 4-180
API: Delete Item UOMs 4-182
API: Import Item Countries 4-183
API: Delete Item Countries 4-184
API: Import Item Dimensions 4-185
API: Delete Item Dimensions 4-187
API: Import Item Manufacturers 4-188
API: Delete Item Manufacturers 4-189
REST Service: Ticket 4-190
Service Base URL 4-190
API Definitions 4-190
API: readTicket 4-191
API: findTickets 4-193
API: readArchivedTicket 4-194
API: findArchivedTickets 4-195
API: createTickets 4-196
API: Update Tickets 4-198
API: printTickets 4-199
API: findTicketFormats 4-200
API: findTicketPrinters 4-201
REST Service: Transfer 4-202
Service Base URL 4-202
API Definitions 4-203

API: Read Transfer 4-203
API: Find Transfer 4-205
API: Request Transfer 4-206
API: Release Request 4-207
API: Approve Request 4-207
API: Reject Request 4-207
API: Cancel Request 4-208
API: Approve Transfer 4-208
API: Cancel Transfer 4-209
API: Close Transfer 4-209
API: Find Transfer Contexts 4-210
Additional Data Definitions 4-210
REST Service: Transfer Shipment 4-211
Service Base URL 4-211
API Definitions 4-211
API: Read Shipment 4-212
API: Find Transfer Shipment 4-216
API: Submit Shipment 4-217
API: Cancel Submit Shipment 4-218
API: Dispatch Shipment 4-218
API: Cancel Shipment 4-218
API: Confirm Carton 4-219
API: Cancel Carton 4-219
API: Open Carton 4-220
API: createShipment 4-220
API: updateShipment 4-223
API: createCarton 4-224
API: updateCarton 4-226
Index 4-227
REST Service: Translations 4-228
Service Base URL 4-228
API Definitions 4-228
API: Find Locales 4-228
API: Find Translations 4-230
API: Find Items Descriptions 4-231
REST Service: Vendor Delivery 4-232
Service Base URL 4-232
API Definitions 4-232
API: Read Delivery 4-233
API: Find Vendor Delivery 4-236
API: Receive Delivery 4-237
API: Confirm Delivery 4-238

API: Reject Delivery 4-238
API: Cancel Delivery 4-239
API: Submit Carton 4-239
API: Cancel Submit Carton 4-240
API: Confirm Carton 4-240
API: Cancel Carton 4-240
API: Open Carton 4-241
Additional Data Definitions 4-241
Rest Service: Vendor Return 4-243
Service Base URL 4-243
API Definitions 4-243
API: Approve Return 4-243
API: Close Return 4-244
API: Find Return 4-245
API: Import Return 4-247
API: Import Return Delete 4-249
API: Read Return 4-250
API: Update Return 4-252
Vendor Shipment 4-253
Service Base URL 4-253
Find Shipments — GET 4-254
Method 4-254
URL 4-254
Request 4-254
Responses 4-255
Create Shipment — POST 4-256
Method 4-256
URL 4-256
Request 4-256
Responses 4-260
Read Shipment — GET 4-261
Method 4-261
URL 4-261
Request 4-261
Responses 4-262
Update Shipment — POST 4-268
Method 4-268
URL 4-268
Request 4-268
Responses 4-270
Submit Shipment — POST 4-270
Method 4-271

URL 4-271
Request 4-271
Responses 4-272
Cancel Submit Shipment — POST 4-272
Method 4-272
URL 4-272
Request 4-272
Responses 4-273
Dispatch Shipment — POST 4-273
Method 4-273
URL 4-273
Request 4-273
Responses 4-274
Cancel Shipment — POST 4-274
Method 4-274
URL 4-274
Request 4-274
Responses 4-274
Create Carton — POST 4-275
Method 4-275
URL 4-275
Request 4-275
Responses 4-277
Update Carton — POST 4-278
Method 4-278
URL 4-278
Request 4-278
Responses 4-280
Confirm Carton — POST 4-281
Method 4-281
URL 4-281
Request 4-281
Responses 4-281
Cancel Carton — POST 4-281
Method 4-282
URL 4-282
Request 4-282
Responses 4-282
Open Carton — POST 4-282
Method 4-282
URL 4-283
Request 4-283

Responses 4-283
REST Service: Warehouse 4-283
Service Base URL 4-283
API Definitions 4-283
API: Import Warehouses 4-284
API: Delete Warehouse 4-285
API: Import Items 4-286
API: Delete Items 4-287
API: Import Adjustments 4-288
API: Import Inventory 4-289

5 Sales Integration

POS and Sales Audit Process Flow 5-2
Sales and Return Processing 5-2
Customer Order Processing 5-3
Item Disposition 5-3
Drop Ship 5-3
Item Types 5-3
RFID 5-4

6 Outbound Integration

Integration with Customer Order System 6-1
Integration with Manifesting Systems 6-2
Integration for Notifications 6-3
Integration for Sales Forecast 6-3
Integration for Store Order 6-3
Integration for Ticket Printing 6-4
Rest Ticket Printing 6-4
API Publish Tickets 6-4

7 File Transfer Services

Overview 7-1
How to Call FTS APIs 7-2
Service Base URL 7-3
FTS Bucket Name 7-3
FTS Endpoints 7-3
Preparing for Authorization 7-4
Retrieving Access Client Token 7-6
FTS API Call Common Headers 7-6

How to Use FTS API to find Object Storage Prefixes 7-7
How to Use FTS APIs to Upload Files to Object Storage 7-7
Step1: Request upload PAR 7-7
Step2: Use PAR to upload data files to Object Storage 7-7
How to Use FTS API to List Files in Object Storage 7-8
How to Use FTS APIs to Download Files from Object Storage 7-8
Step1: Find what files are available for downloads 7-8
Step2: Request Download PAR for downloading data files from Object Storage 7-8
Step3: Download the file using the par returned from step2 7-9
Handling Import Data Files 7-9
Supported Import Data Files 7-9
Upload Import Data Files to Object Storage 7-10
Handling Export Data Files 7-10
Supported Export Data Files 7-11
Steps to Download Export Data Files from Object Storage 7-11
File Transfer Service UI 7-11
FTS API Specifications 7-14
Ping 7-14
List Prefixes 7-14
List Files 7-15
Move Files 7-17
Delete Files 7-18
Request Upload PAR 7-19
Request Download PAR 7-20
File Transfer Service Troubleshooting 7-21
Test FTS API using Postman 7-22
Step 1: Get Client Access Token 7-22
Step 2: Call FTS Endpoints 7-22

8 SOAP Web Services

Security Considerations 8-1
Functionality 8-2
Available Web Services 8-2
Web Services Basic Design Principles 8-3
Internally Managed vs Externally Managed 8-4
Web Service Operation Basic Design Standards 8-5
Interpreting Validation Errors 8-6
Common Error Codes 8-6
Validation Error (Fault Example) 8-7
Business Error (Fault Example) 8-8
Web Services 8-8

ActivityLock 8-8
FulfillmentOrderDelivery 8-9
FulfillmentOrderPick 8-9
FulfillmentOrderReversePick 8-10
InventoryAdjustment 8-11
ItemBasket 8-11
OrderRequest 8-12
POSTransaction 8-13
ProductGroup 8-13
ProductGroupSchedule 8-13
ReplenishmentGap 8-14
RfidInventory 8-14
ShelfAdjustment 8-15
ShelfReplenishment 8-15
StockCount 8-16
Store 8-17
StoreFulfillmentOrder 8-17
StoreInventory 8-18
StoreInventoryISN 8-18
StoreInventoryUIN 8-19
StoreItem 8-19
StoreItemPrice 8-20
StoreNotification 8-20
StoreShipmentManifest 8-20
StoreShipmentReason 8-21
StoreTicket 8-21
StoreTransfer 8-21
TransferDelivery 8-22
TransferShipment 8-24
VendorDelivery 8-25
VendorReturn 8-26
VendorShipment 8-26
Enterprise Documentation 8-28

Send Us Your Comments

Oracle Retail Enterprise Inventory Cloud Service Inbound and Outbound Implementation Guide, Release 24.1.401.0

Oracle welcomes customers' comments and suggestions on the quality and usefulness of this document.

Your feedback is important, and helps us to best meet your needs as a user of our products. For example:

- Are the implementation steps correct and complete?
- Did you understand the context of the procedures?
- Did you find any errors in the information?
- Does the structure of the information help you with your tasks?
- Do you need different information or graphics? If so, where, and in what format?
- Are the examples correct? Do you need more examples?

If you find any errors or have any other suggestions for improvement, then please tell us your name, the name of the company who has licensed our products, the title and part number of the documentation and the chapter, section, and page number (if available).

> **Note:**
>
> Before sending us your comments, you might like to check that you have the latest version of the document and if any concerns are already addressed. To do this, access the Online Documentation available on the Oracle Help Center (OHC) website. It contains the most current Documentation Library plus all documents revised or released recently.

Send your comments to us using the electronic mail address: retail-doc_us@oracle.com

Please give your name, address, electronic mail address, and telephone number (optional).

If you need assistance with Oracle software, then please contact your support representative or Oracle Support Services.

If you require training or instruction in using Oracle software, then please contact your Oracle local office and inquire about our Oracle University offerings. A list of Oracle offices is available on our Web site at http://www.oracle.com.

Preface

This document describes the administration tasks for Oracle Retail Enterprise Inventory Cloud Service.

Audience

This document is intended for administrators.

Documentation Accessibility

For information about Oracle's commitment to accessibility, visit the Oracle Accessibility Program website at .

https://www.oracle.com/corporate/accessibility/

Access to Oracle Support

Oracle customers that have purchased support have access to electronic support through My Oracle Support. For information, visit

My Oracle Support or visit

https://www.oracle.com/corporate/accessibility/learning-support/#support-tab if you are hearing impaired.

Related Documents

For more information, see the following documents in the Oracle Retail Store Inventory Operations Cloud Services Release 24.1.401.0 documentation set:

- *Oracle Retail Store Inventory Operations Cloud Services Release Notes*
- *Oracle Retail Store Inventory Operations Cloud Services Implementation Guide*
- *Oracle Retail Store Inventory Operations Cloud Services Data Model*
- *Oracle Retail Enterprise Inventory Cloud Service Administration Guide*
- *Oracle Retail Enterprise Inventory Cloud Service Security Guide*
- *Oracle Retail Enterprise Inventory Cloud Service User Guide*
- *Oracle Retail Store Operations Cloud Service User Guide*
- *Oracle Retail Store Operations Cloud Service Mobile Guide*

Improved Process for Oracle Retail Documentation Corrections

To more quickly address critical corrections to Oracle Retail documentation content, Oracle Retail documentation may be republished whenever a critical correction is needed. For critical

corrections, the republication of an Oracle Retail document may at times not be attached to a numbered software release; instead, the Oracle Retail document will simply be replaced at the Oracle Help Center (OHC) website, or, in the case of Data Models, to the applicable My Oracle Support Documentation container where they reside.

This process will prevent delays in making critical corrections available to customers. For the customer, it means that before you begin installation, you must verify that you have the most recent version of the Oracle Retail documentation set. Oracle Retail documentation is available at the Oracle Help Center at the following URL:

https://docs.oracle.com/en/industries/retail/index.html

An updated version of the applicable Oracle Retail document is indicated by Oracle part number, as well as print date (month and year). An updated version uses the same part number, with a higher-numbered suffix. For example, part number F123456-02 is an updated version of a document with part number F123456-01.

If a more recent version of a document is available, that version supersedes all previous versions.

Oracle Retail Documentation at the Oracle Help Center

Oracle Retail product documentation is available on the following website:

https://docs.oracle.com/en/industries/retail/index.html

(Data Model documents are not available through Oracle Help Center. You can obtain them through My Oracle Support.)

Conventions

The following text conventions are used in this document:

Convention	Meaning
boldface	Boldface type indicates graphical user interface elements associated with an action, or terms defined in text or the glossary.
italic	Italic type indicates book titles, emphasis, or placeholder variables for which you supply particular values.
`monospace`	Monospace type indicates commands within a paragraph, URLs, code in examples, text that appears on the screen, or text that you enter.

1

Introduction

This guide describes integration through RIB, batches, web services and file transfer services.

- Retail Home Integration
- Retail Integration Cloud Service (RICS) - based Integration
- REST Web Services
- Sales Integration
- Outbound Integration — Including:
  - Integration with Customer Order System
  - Integration with Manifesting Systems
  - Integration for Notifications
  - Integration for Sales Forecast
  - Integration for Ticket Printing
- File Transfer Services
- SOAP Web Services

2

Retail Home Integration

EICS now supports following integration scenarios with Retail Home:

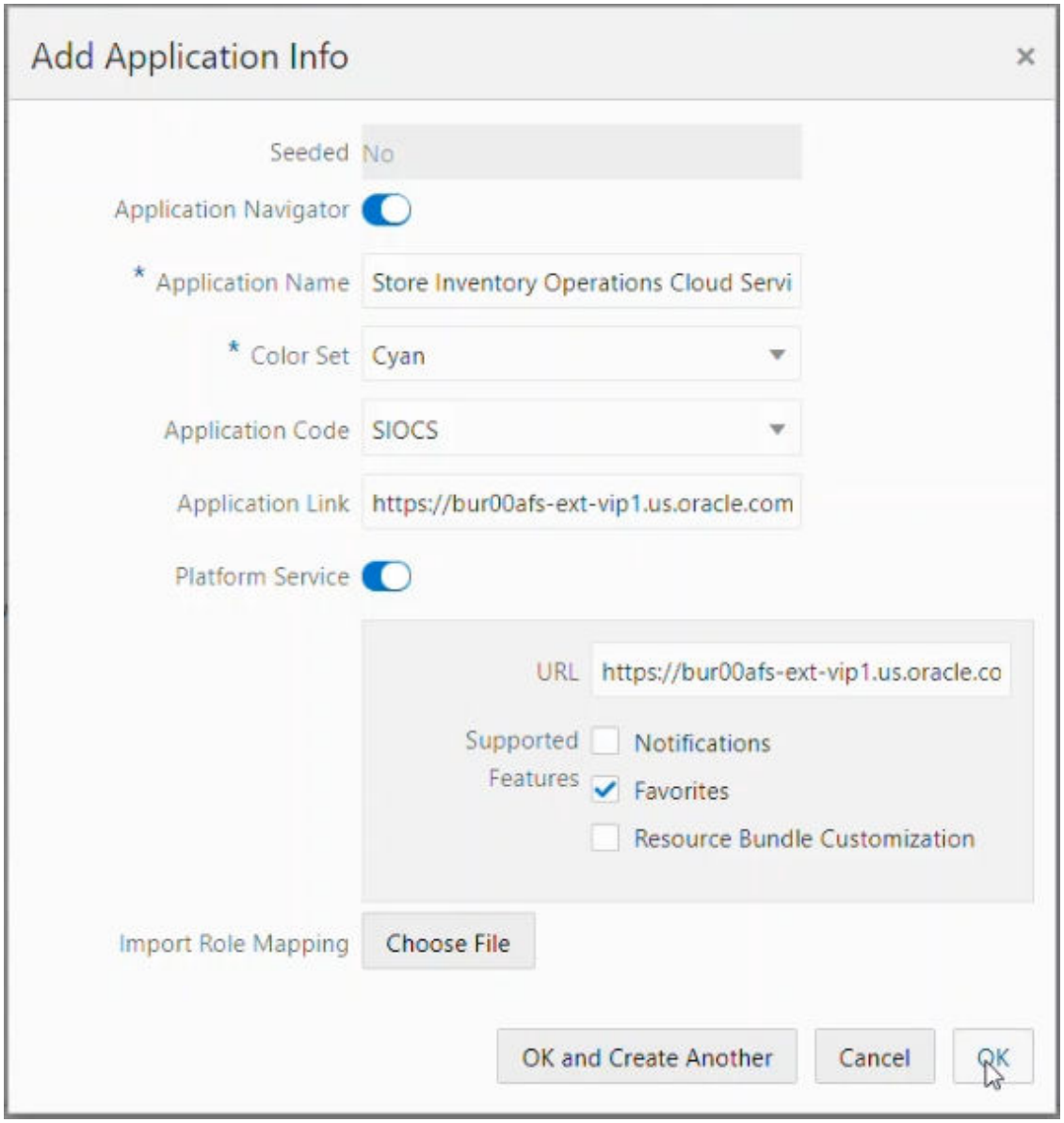
- Launch SIOCS web client from Retail Home
- Launch SIOCS favorites from Retail Home
- Display a tile report for items that are out of stock on shop floor
- Display a tile report for stock counts that are pending authorization
- Launch detailed operational views in SIOCS web client from related tile reports in Retail Home

Launch SIOCS from Retail Home

Launching SIOCS client requires an entry to be made under the application navigator section of Retail Home. It enables the user to launch SIOCS web client in a new browser tab from within Retail Home. Please refer to *Oracle Retail Home Administration Guide* for information on how to work with application navigator in Retail Home.

The SIOCS application configuration should look like this:

Figure 2-1 Add Application Info



- **Seeded**: Disabled and set to No.
- **Application Navigator**: Enable it to launch SIOCS client from Retail Home.
- **Application Name**: The name of the application that is, Store Inventory Operations Cloud.
- **Color Set**: Any color that you want to allocate to SIOCS.
- **Application Code**: Select SIOCS from the drop down.
- **Application Link**: The URL of SIOCS web client.
- **Platform Service**: Enable it to use Favorites feature.
  - **URL**: The base URL of the platform services. The URL would be of the form https://<SIOCS-HOST>/RetailAppsPlatformServices
    <SIOCS-HOST> is the same host in Application Link.

- **Supported Features**: Check only the favorites feature.

The user needs to be part of RETAIL_HOME_ADMIN security group in order to access Application Navigator in Retail Home.

Tile Reports

EICS supports following two types of two metric reports:

- Shop Floor Out of Stock Items
- Stock Counts - Ready to Authorize

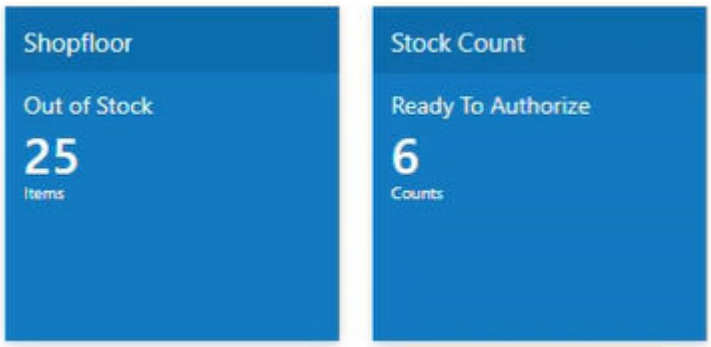
Adding an application navigator entry for SIOCS will automatically configure EICS tiles on Retail Home.

The data seed features do the following:

1. Creates a custom report for EICS tiles on Retail Home.
2. Creates two tiles from the custom report and maps them to retail_home_users IDCS or OCI IAM application role.
3. The data seed features will also configure tile states for the two tiles and hook them up with EICS end points.

After all the configuration, you should be able to see EICS tiles on the dashboard. They should look like the ones below:

Figure 2-2 Example EICS Tiles



EICS Endpoints

EICS exposes following two endpoints:

Shop Floor Out of Stock Items

This endpoint can be used as a data source for **Shop floor Out of Stock** tile state.

The response contains information on number of items that are out of stock across all the stores that are accessible to the user.

If the percentage of out of stock items to total items is greater than the **Shopfloor Out of Stock Items Critical Percentage** system configuration, EICS marks the response as important which displays a '!' mark next to the number on the tile report.

Table 2-1 Shop Floor Out of Stock

Endpoint	Operational View
https://<eics_external_load_balancer_address>/<CUST_ENV>/siocs-client-services/internal/rhreports/outofstock/shopfloor/tile	Shopfloor Out of Stock

Stock Counts - Ready to Authorize

This endpoint can be used as a data source for **Stock Count - Ready to Authorize** tile state.

The response contains information on number of stock counts that are pending authorization across all stores that are accessible to the user.

Table 2-2 Stock Counts - Ready to Authorize

Endpoint	Operational View
https://<eics_external_load_balancer_address>/<CUST_ENV>/siocs-client-services/internal/rhreports/readytoauthorize/tile	Stock Count - Ready To Authorize

The response payloads of both these endpoints confirm to the two metric payload specifications of Retail Home.

User should be a part of retail_home_users IDCS or OCI IAM application role to access these endpoints.

For convenience, EICS also provides a RETAIL HOME security role that captures security permissions required to access these operational views. The user still needs appropriate functional area permissions to navigate to transaction detail screens.

SIOCS Operational Views

EICS has added following operational views that can be hooked with related tiles:

- **Shopfloor Out of Stock Items**

  This view gives a store and item level breakdown of the information that is displayed on the tile. The user can look at item level records for each store and navigate to the item detail screen for any store/item combination provided he or she has the required permissions.

  This view is available under Operations / Operational Views / Shopfloor Out of Stock menu.

- **Stock Count - Ready to Authorize**

  This view gives a store and stock count level breakdown of the information that is displayed on the tile. The user can look at stock count level records for each store and navigate to the stock count detail for any store/count combination provided he or she has the required permissions.

  This view is available under Operations / Operational Views / Stock Count / Ready to Authorize menu.

Launch SIOCS Operational Views from Tile Report

Launching SIOCS operational views from related tile report requires the tile report to be configured with the URL of the related operational view. Once that is done, clicking on tile report header should open the related EICS operational view in a new browser tab.

Subscription Usage Batch

EICS has added a new batch to extract subscription usage for EICS and SOCS respectively during the subscription period. These extracted metrics are pushed to platform services from where Retail Home displays these on the Application Dashboard screen.

This is a restricted batch which by default is scheduled to run every month. The schedule can only be updated by Oracle.

It can also be run as an Adhoc batch from EICS / Admin / Technical Maintenance / Job Admin / Adhoc Job.

3

Retail Integration Cloud Service (RICS) - based Integration

- Security Considerations
- Customer Orders
- Picking
- Deliveries
- Reverse Picking
- Multi Leg
- RIB Payloads
- Purchase Orders and Vendor Deliveries
- Inventory Adjustments
- Items
- Stock Counts
- Transfers
- Transfer Creation
- Transfer Messages
- Transfer Shipment Creation
- Transfer Receiving
- Transfer Doc
- Transfer Shipment
- Transfer Receiving
- Vendor Return

Security Considerations

Customer Administration User must create an IDCS user with the required RIB admin group to access the publisher endpoints.

- ribAdminGroup – For Production environment
- ribAdminGroup_preprod – For Dev/Stage/UAT/test environments

The same user credentials must then be configured on the Credential Administration screen. Refer to Chapter 6 - Technical Maintenance Screens / Credential Administration section for more details.

Customer Orders

- Customer Order Create is used for Customer Orders that are a type of Web Order integrated through a message (FulfilOrdDesc). These integrations are used for the customer order from the Order Management System (OMS).
- The Customer Order Create failure message (FulfilOrdCfmDesc) is a message that will be sent out to external system when we get a Customer Order that comes into the system through the RIB and fails due to validation issues such as an invalid item. The purpose of the create failure is so other systems will know it has failed when it came in and that it is not being processed.
- The Stock Order Status message (SOStatusDesc) will be sent out with an SI upon reserving inventory for the customer order.

Picking

- A Stock Order Status message (SOStatusDesc) is sent out with a type of SI upon reserving inventory. This happens when more is picked than what was on the order due to tolerances. This could also occur when a substitute item is added during the picking process.
- The Stock Order Status message (SOStatusDesc) with a type of SD will be published to un-reserve the original items inventory when a substitute item has been added during picking.
- A Stock Order status message (SOStatusDesc) is sent out with a type of PP when picking is completed.
- Item Substitutes are sent to EICS from the merchandising system through the item message (ItemDesc).

Deliveries

- An ASN Out message (ASNOutDesc) is sent out upon dispatching of the Delivery. This will be done for pick-ups and for shipments.
- The Stock Order Status message (SOStatusDesc) with a type of PP will be published for the pick quantity in the scenario that more was delivered than what was picked.
- The Stock Order Status message (SOStatusDesc) with a type of SI will be published for the reserved quantity. This will occur when more was delivered than what was reserved. This can happen when picking was not required, the reservation occurs upon receipt of a delivery, and the full amount had not been received, therefore not reserved.

Reverse Picking

- Customer Order Cancellations (FulfilOrdRef) will come into EICS from external system such as an OMS through the RIB. This service will perform all the validations to determine if it should create a reverse pick and whether or not that reverse pick should be auto completed.
- Customer Order Cancellation Confirmation (FulfilOrdRef) is a message to send to OMS upon completing of the system-generated reverse pick.

- Stock Order Status message (SOStatusDesc) with a type of SD will be published for the reserved quantity to un-reserve the inventory for the reverse pick for system-generated picks.
- Stock Order Status message (SOStatusDesc) with a type of PU will be published for the reverse picked quantity to un-pick the inventory for system-generated picks.

Multi Leg

The following integrations are in addition to the standard integrations that already exist such as receipt message, and so on:

- The Stock Order Status message (SOStatusDesc) with a type of SI will be published for the reserved quantity.
- The Stock Order Status message (SOStatusDesc) with type of PP will be published for the picked quantity.

RIB Payloads

RIB payloads are used to communicate information to external systems through RIB Integration.

RIB Payload	Description
FulfilOrdDesc	RIB payload that contains information about a new web order type of fulfillment order to be created in.
FulfilOrdCfmDesc	RIB payload sent from EICS that contains fulfillment order information when that order creating in EICS failed
FulfilOrdRef	RIB payload that contains information about a fulfillment order cancelation. It is sent to EICS to convey a cancelation request and sent from EICS to convey actual cancellations.
SOStatusDesc	Sent from EICS to convey changes in item status for a specific fulfillment order. Such changes of status include (un)reservation and (un)picking.
ASNOutDesc	Sent from EICS to convey a delivery for specified fulfillment order.

Purchase Orders and Vendor Deliveries

MERCHANDISING publishes the Purchase Orders created for the direct store deliveries using RIB messages. EICS subscribes to these messages and stores them in the EICS database to enable receipt against Purchase Orders.

MERCHANDISING publishes the unit cost of the item at the item/supplier/country level for EICS to use in the receiving process.

EICS publishes the receipts done against the Purchase Order to the merchandising system (Receiving message).

EICS publishes the DSD receipts created in EICS without a Purchase Order to the merchandising system (DSDReceipts and DSD Deals messages).

EICS publishes the receiver unit adjustment done for the deliveries that are already confirmed (receiving message).

EICS is also capable of subscribing to the vendor EDI ASNs through RIB using the ASN In message format.

RIB payloads are used to communicate information from EICS to external systems and from external system to EICS through RIB Integration.

RIB Payload (Subscriber)	Description
PORef	RIB payload that contains reference level information of a purchase order.This payload is used for removal of purchase orders.
PODesc	RIB payload that contains detailed information of a purchase order.This payload is used for creation and modification of purchase orders.
ASNInRef	RIB payload that contains reference level information of an ASN. This payload is used for removal of an ASN.
ASNInDesc	RIB payload that contains detailed information about the ASN. This payload is used for creation of a direct delivery (document type= 'P') or a warehouse delivery (document type= 'D'). EICS consumes this payload from warehouse when source and/or destination for ASN is a warehouse system.

RIB Payload	Description
ReceiptDesc	RIB payload that contains detailed information of the direct delivery receipt. This is published when the purchase order is not null. EICS also consumes this payload for warehouse receiving.
DSDReceiptDesc	RIB payload that contains detailed information of the direct delivery receipt. This is published when the purchase order is null.
SOStatusDesc	RIB payload sent from EICS to convey changes in item status for a specific fulfillment order. EICS also consumes this payload from warehouse for stock movements originating at the warehouse.
InvAdjustDesc	RIB payload that contains information about destination of the adjustment and an InvAdjustDtl.

Inventory Adjustments

Inventory adjustments integrate to MERCHANDISING at the item level using the RIB. EICS creates the adjustments and groups them together by a header with multiple items, but for integration purposes they are published out at an item level.

Inventory adjustments are published for all manual and external system generated adjustments where the Publish indicator for the reason code is checked. Adjustments are also published for other types of transactions in EICS where the merchandise system is expecting an adjustment for stock on hand updates, for example, receiving a DSD with damaged goods. An adjustment is created behind the scenes only for publishing purposes to notify the merchandising system to move the goods into the unavailable bucket. These system type adjustments are not considered an adjustment within EICS; however, they are published as such for integration purposes.

EICS subscribes to inventory adjustment messages from warehouse systems and updates the warehouse inventory buckets in EICS.

RIB payloads are used to communicate to external systems through RIB Integration.

The following table shows the list of RIB Payloads available for inventory adjustments.

RIB Payload	Description
InvAdjustDesc	RIB payload that contains information about destination of the adjustment and an InvAdjustDtl.
InvAdjustDtl	Contains detailed information about the item adjustment.

Items

Items come to EICS from a merchandising system through the RIB (items, item loc messages). EICS also gets information about items associated to a supplier through the RIB. Extended attributes are not received or sent on RIB payloads.

RIB Payload	Description
ItemDesc	This payload contains information about an item. It contains a wide variety of information about the item including suppliers, UPCs, ticketing information, image information, UDAs, and related items
ItemLocDesc	This payload contains information about an item at a specific location.
ItemSupDesc	This payload contains information about an item for a specific supplier.
ItemSupCtyDesc	This payload contains information about an item for a specific supplier within a specific country.

Stock Counts

Stock counts generate inventory adjustment when completed.

RIB payloads are used to communicate to external systems through RIB.

RIB Payload	Description
InvAdjustDesc	RIB payload that contains information about destination of the adjustment and an InvAdjustDtl.
InvAdjustDtl	Contains detailed information about the item adjustment.

EICS does not integrate using a web service to any other Oracle Retail products for stock counts.

Transfers

The Transfer Shipping allows for creating shipment, dispatching shipment, canceling shipment, creating container, approving container, adjusting container, and canceling the container.

The Transfer Receiving dialog allows for confirming receipt, copying misdirected container, receiving container and detailed receiving.

This section covers creating transfer documents which are then included in a transfer shipment and dispatched to another store, warehouse, or finisher.

Transfer Creation

Transfer documents can be created in the following ways:

- Requesting store can create a transfer request.
- Sending store can initiate a transfer by creating a transfer.
- Merchandising can create a transfer request.

Each transfer document will have one or more items.

Transfer Messages

EICS will publish messages to Merchandising when the following happen:

- Transfer is rejected.
- Transfer is approved.
- Transfer quantity is updated from the shipment.

Transfer Shipment Creation

Transfer Shipment describes the containers and the items for the shipment taking place. The shipment may be for one or more transfer documents if the transfer is going to the same destination. Dispatching a shipment will update the transfer document.

The user can create a shipment without referencing existing transfers or can create a new transfer on fly (Ad hoc transfer) based on the shipment information.

Transfer Receiving

This transaction captures a delivery that took place from a warehouse, store, or finisher to the store receiving the delivery. It describes the containers and the items of the delivery that should be received by the store. Receiving a container of the delivery will update the transfer document.

Figure 3-1 Transfer Request Flow

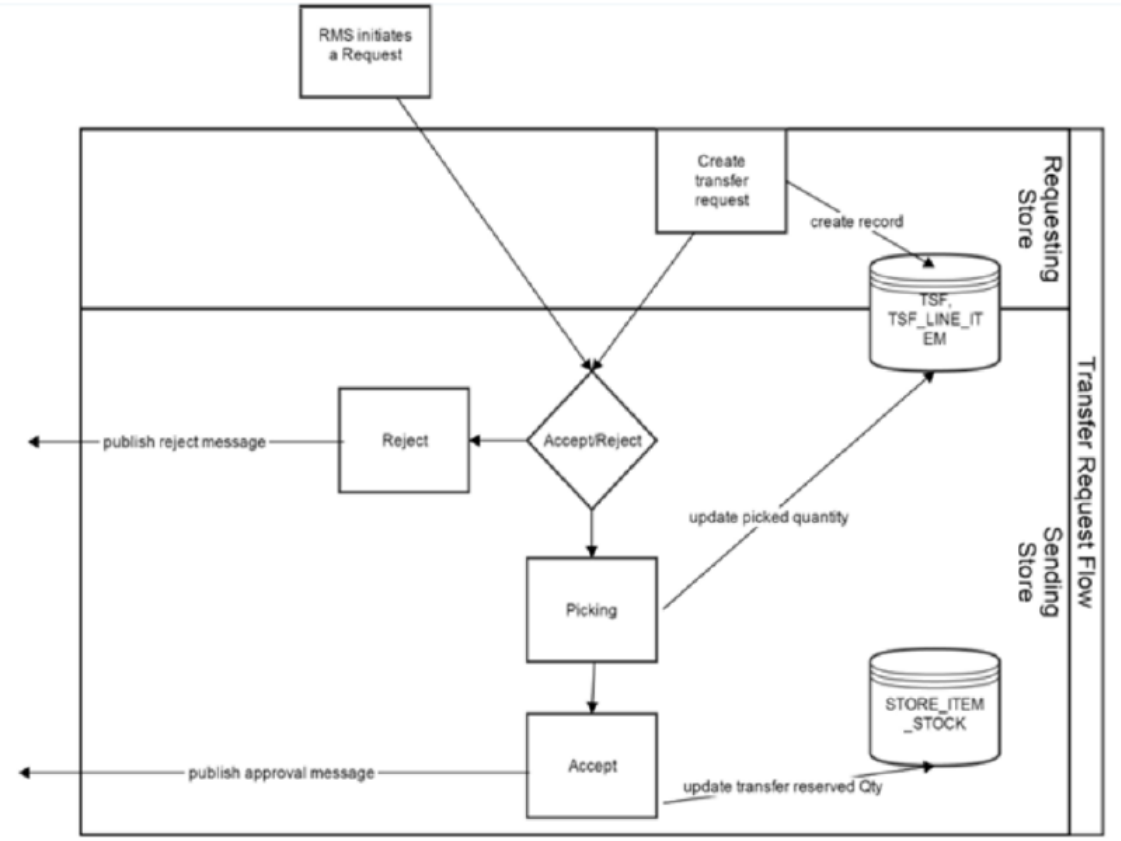


Figure 3-2 Transfer Create Flow

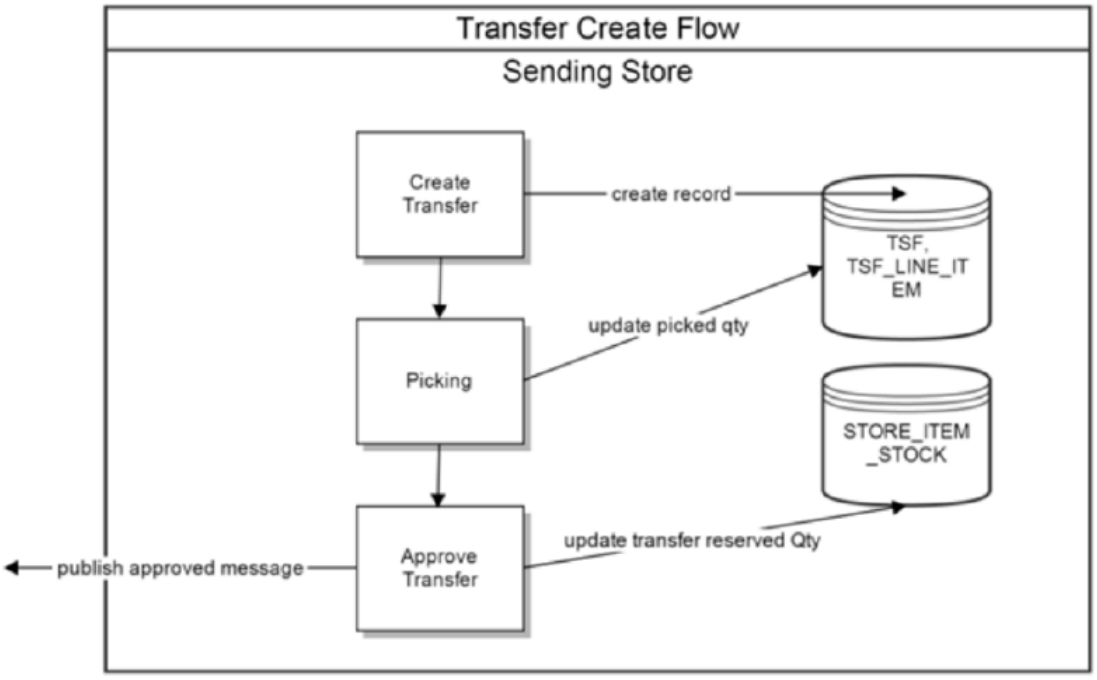


Figure 3-3 Transfer Shipment Creation Flow

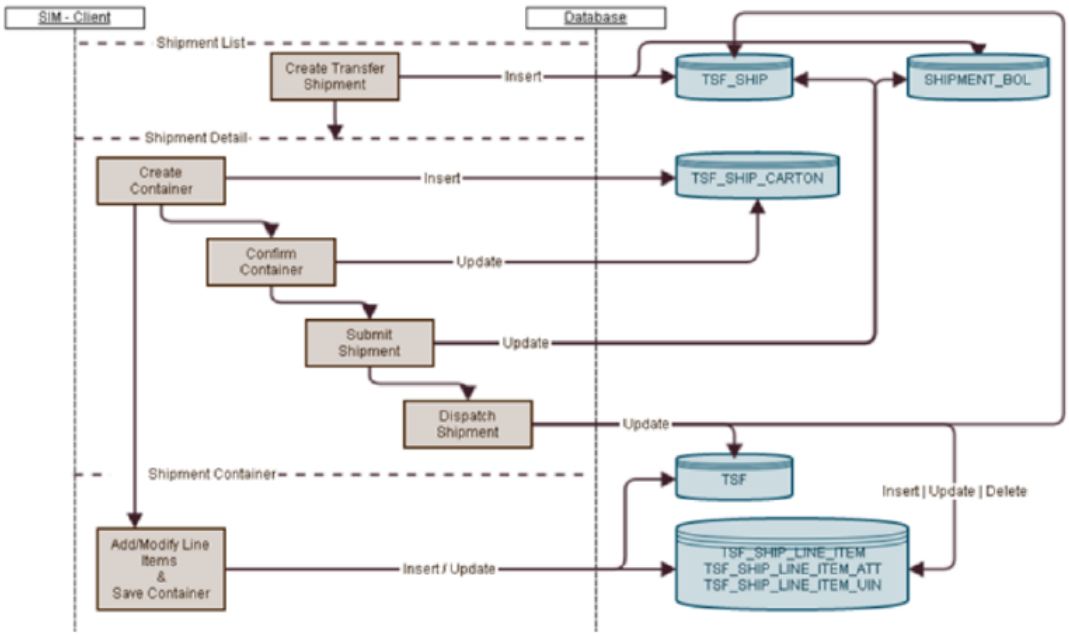
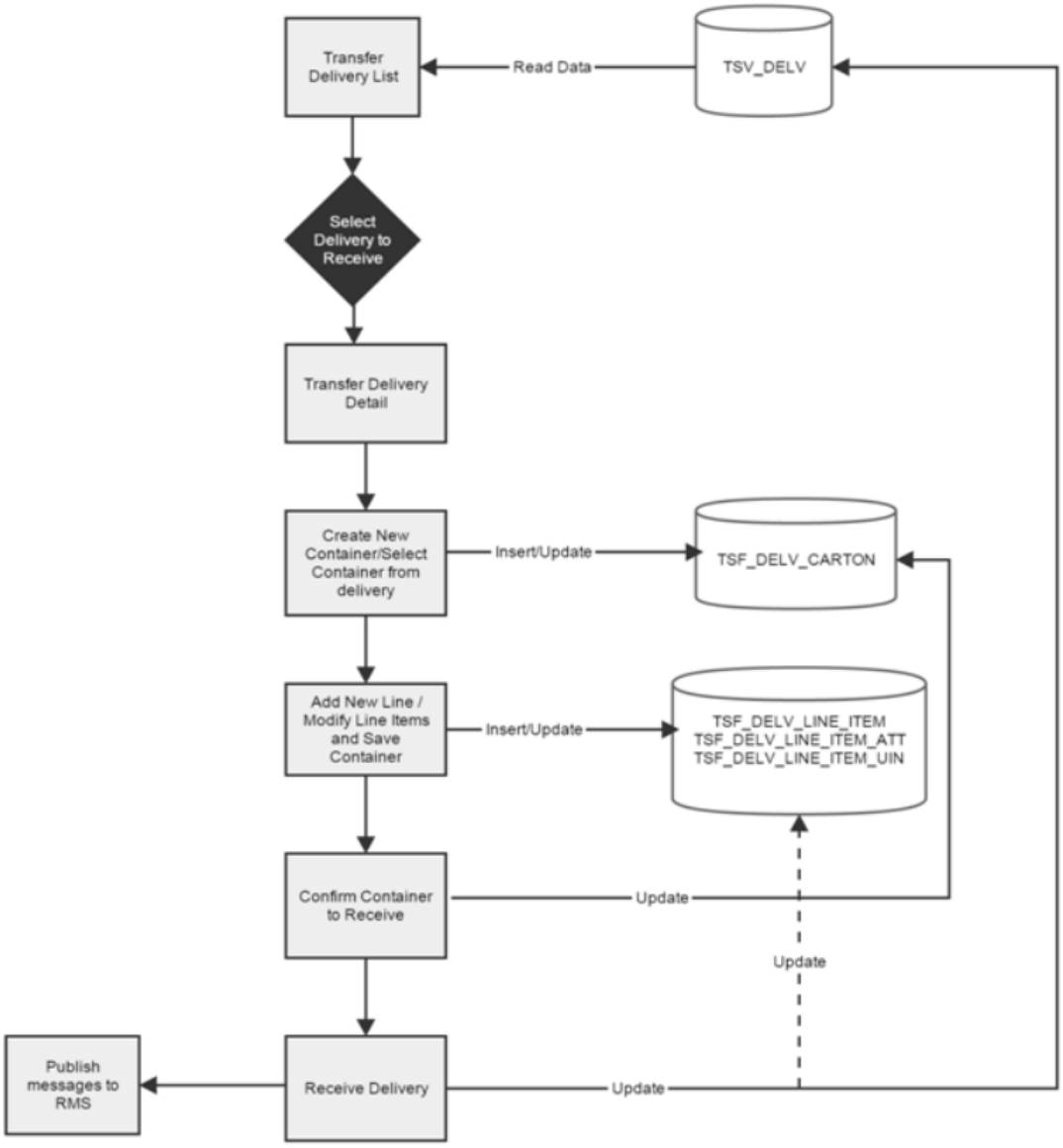


Figure 3-4 Transfer Receiving Process Flow



Transfer Doc

RIB Payload	Description
SODesc	This message is received from external systems when a stock order/ transfer has been created
SOStatusDesc	This message is received from external systems when a stock order/ transfer has been modified.
SORef	This message is received from external systems when a stock order/ transfer has been deleted.

Transfer Shipment

RIB Payload	Description
ASNOutDesc	This message is sent to external systems when the transfer shipment is dispatched.
ManifestCloseVo	This message is received from an external system to indicate physical shipment has been accepted. This will attempt to auto-close the transfer shipment if all items are shipped.
ManifestDesc	This message is sent to an external system when manifesting is activated, and a transfer shipping container is confirmed.
ShipInfoDesc	This message is sent to an external system when pre-shipment notifications are active, and a transfer shipment is either submitted or dispatched (without previously being submitted).
SOStatusDesc	This message is sent to an external system when a transfer shipment container is saved with shipping quantities. It is also sent when a transfer shipment container is canceled but had shipping quantities. Increase and decrease of quantities is indicated by the SI or SD codes.

Transfer Receiving

RIB Payload	Description
ASNInDesc	Sent from external system to indicate a delivery is tracking place. It creates a transfer delivery record within EICS when a store location is involved.
ReceiptDesc	Sent to external system when a transfer delivery is confirmed. Sent from external warehouse system when a transfer delivery is received at the warehouse.

Vendor Return

RTV Creation

RTVs can only be created by a request from MERCHANDISING:

Each vendor return will have one or more items.

RTV Shipment

Each RTV shipment will tie back to a single vendor return document.

RTV shipment can be created in two ways:

- From an externally initiated approved vendor return document.
- Creation of ad hoc vendor return shipment which will create an approved vendor return on the fly.

Each vendor return shipment will have one or more containers; each container in turn will have one or more items.

EICS may publish messages when the following happens:

- RTV shipment container is updated, and saved (Return To Vendor Publish)
- RTV shipment is cancelled or rejected (Return To Vendor Publish)
- RTV shipment is dispatched (Return to Vendor Publish and Ship Info Desc Publish, if dispatched without submitting)
- RTV shipment is submitted (Ship Info Desc Publish)
- RTV shipment container is confirmed (RTV manifesting, if configured)
- RTV shipment is submitted (Pre-shipment notification, if configured)

Figure 3-5 RTV Creation Flow

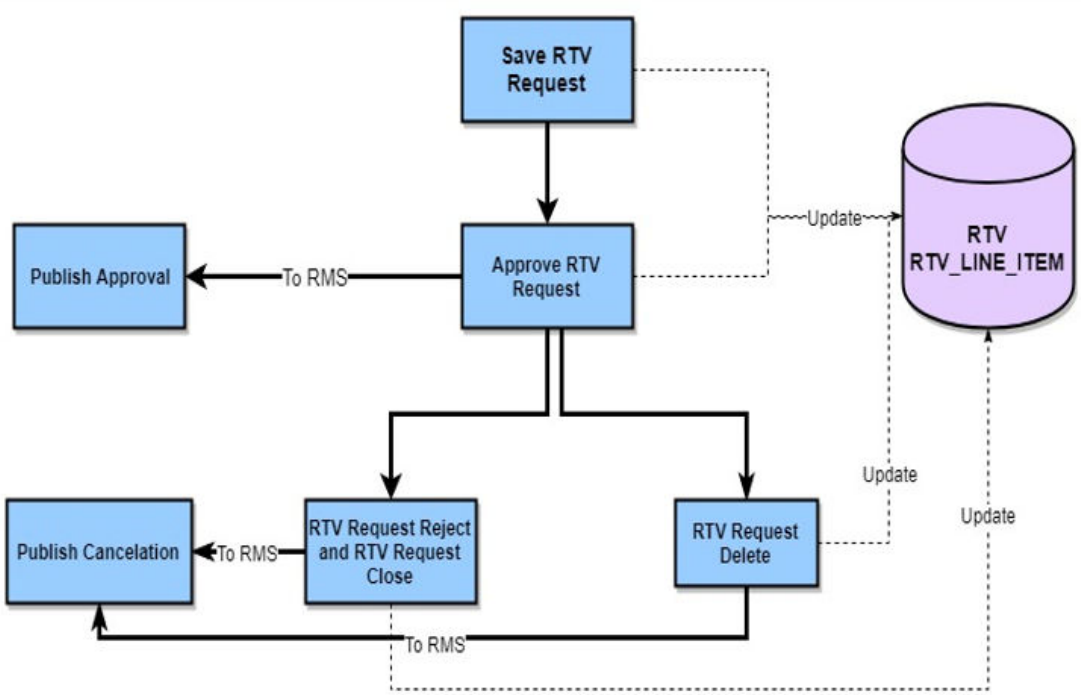


Figure 3-6 RTV Shipment Flow

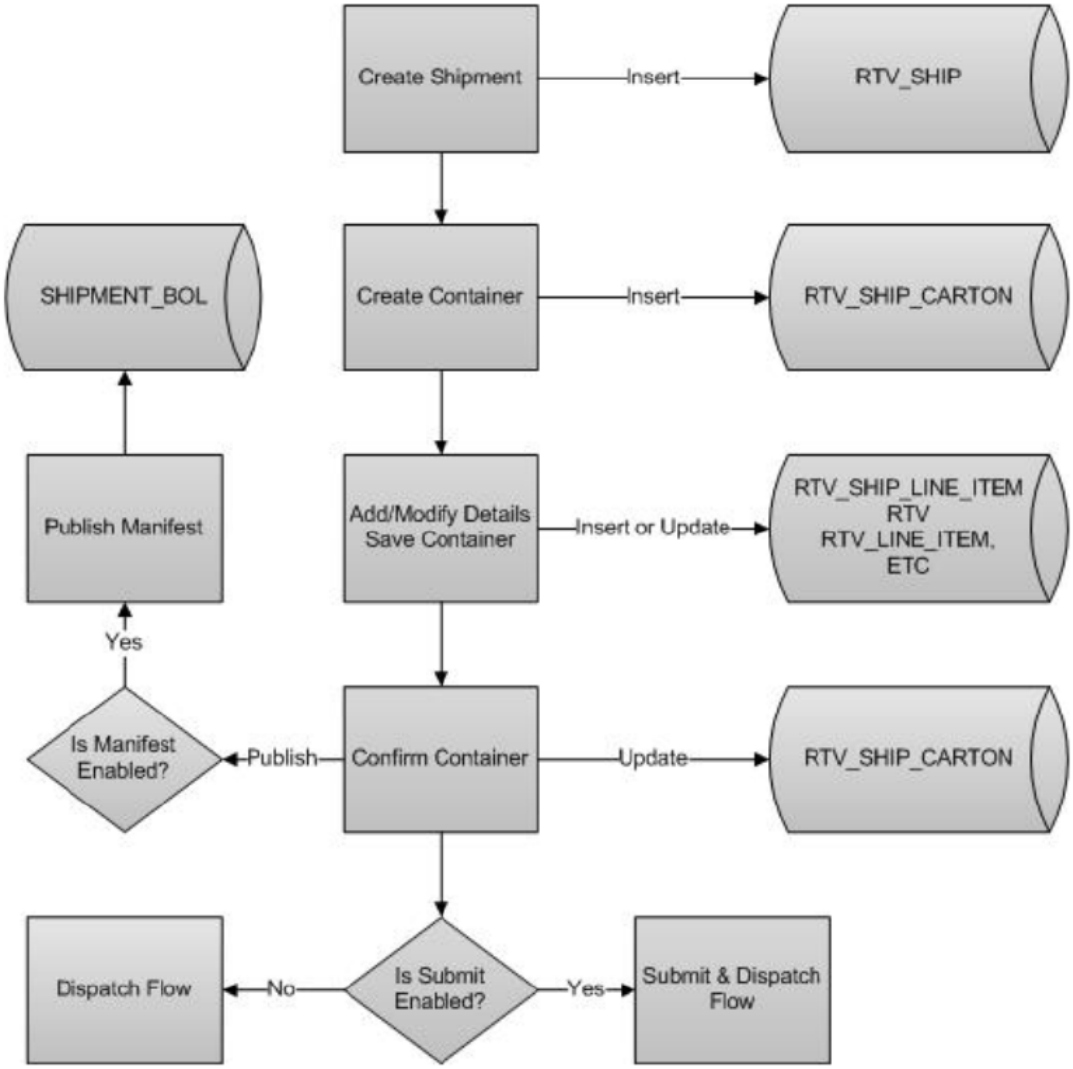


Figure 3-7 RTV Shipment Submit and Dispatch Flow

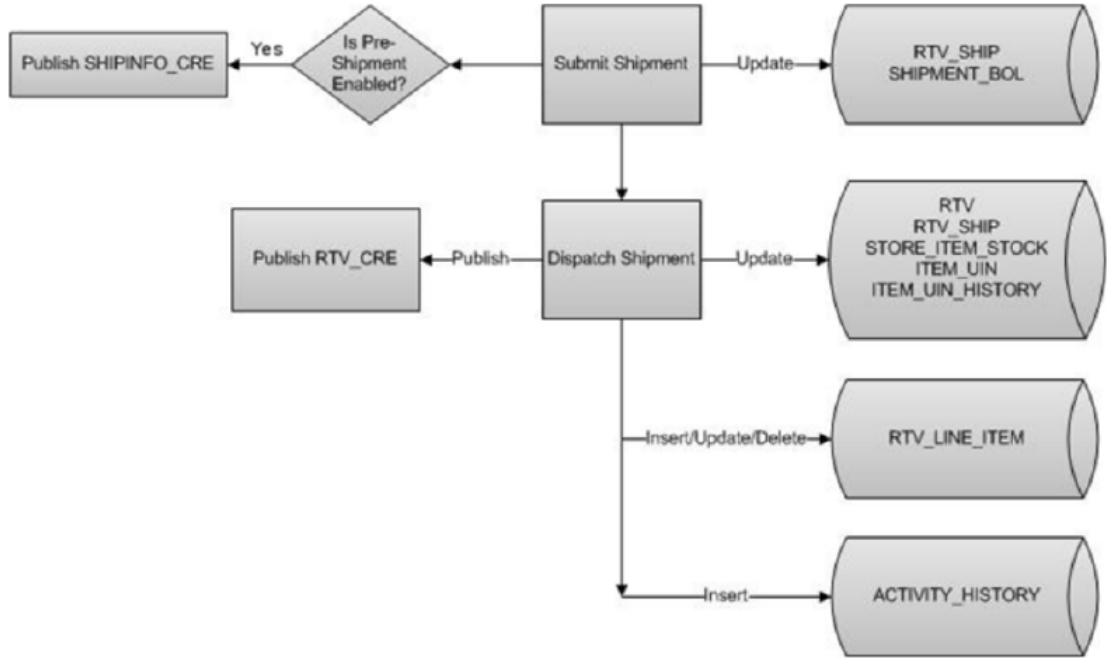
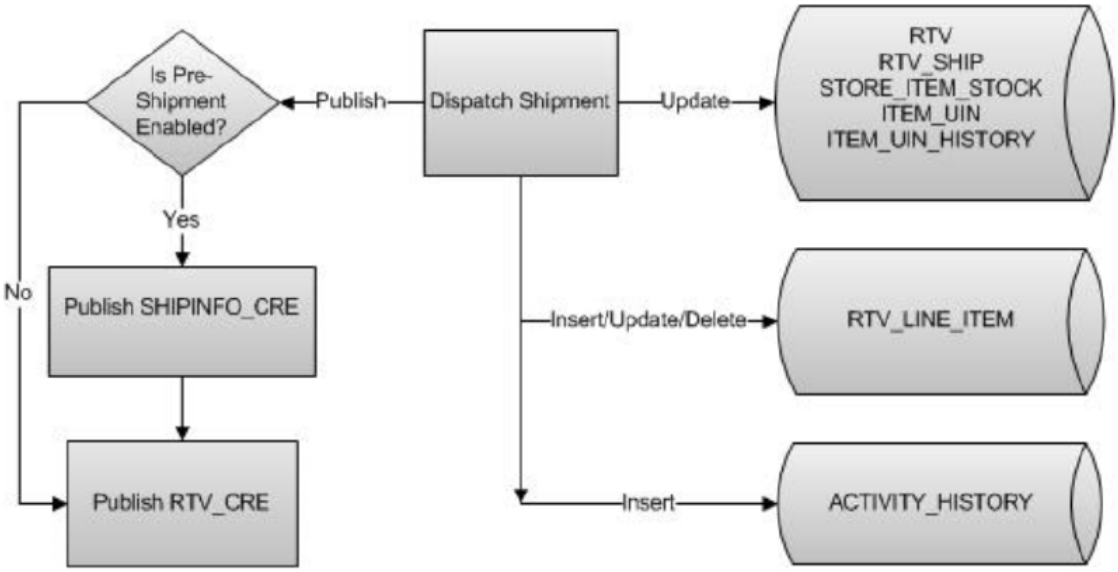


Figure 3-8 RTV Shipment Dispatch Flow



The following payloads are used in RTV operations.

RIB Payload	Description
RTVReqDesc	This payload is sent from an external system to indicate a request for a vendor return. It creates or updates a vendor return document within EICS. It contains a series of RTVReqDtl.

RIB Payload	Description
RTVReqDtl	This payload contains the detailed information about the items on the vendor return.
RTVReqRef	This payload contains reference information about a vendor return when an external system wishes to attempt to cancel the return.
RTVDesc	This payload is sent from EICS to external systems when an RTV shipment is dispatched. This payload is sent from external warehouse system for vendor returns originating at warehouse.

4

REST Web Services

Web services are intended for integration to allow a system using those services to control the flow and processing of data within EICS. There are multiple types of data involved in this integration. Data that is managed by other systems and needs to get into our system, but that EICS does not manage. This includes such concepts as item, stores, and point-of-sale transaction. Data that is managed by EICS includes such ideas as inventory adjustments, transfers, deliveries, and stock counts. Some services will provide ability for external data to get into EICS, some are intended to be used real time such as approving, picking, and dispatching shipments.

- REST WEB Services Security Considerations
- REST WEB Services Basic Design Principles
- Hypertext Transfer Protocol Status Codes
- JSON Error Element and Error Codes
- Integration Error Codes
- Error Code Data Elements

REST WEB Services Security Considerations

The REST web services provided by EICS are secured using OAuth2 tokens and require SSL.

The supported OAuth2 security requires a token requested for the *client_credentials* grant with the EICS integration scope (for example, *rgbu:siocs:integration*).

Note that the scope name differs for each environment.

REST Web Service OAuth2 Requests

This section will describe how to call an EICS web service using the OAuth2 protocol. The target audience is developers who are looking to write code that calls the web service.

Using the OAuth Protocol

The OAuth protocol is relatively straightforward:

- Get an access token from the authentication provider
- Pass the access token along with the web service request

In this case, the authentication provider is Oracle Identity Cloud Service (IDCS). Every customer who purchases a subscription to EICS gets a subscription to IDCS as part of their purchase.

Obtaining a Token

REST APIs use OAuth2.0 for authorization.

To generate a token from IDCS, an IDCS application client will need to be created for you to use.

The Customer Administration users must create their own client credential IDCS application using the Oracle Retail Home Cloud Service. For additional details, refer to Oracle® Retail Home Administration Guide- Chapter: Oauth Application Configuration chapter – Section: Creating OAuth Client Applications.

The App name and scope that should be used for IDCS application creation should be environment specific using the format :

App Name- RGBU_SIOCS_<ENV>_EICS_INT

Scope- rgbu:siocs:integration-<ENV>

Example:

App Name- RGBU_SIOCS_STG1_ EICS_INT

Scope- rgbu:siocs:integration-STG1

You will need the following information about the IDCS application client to request a token:

- IDCS URL
- Client Id
- Client Secret
- Scope Name

> **Note:**
>
> the application client must be assigned the scope from the EICS IDCS cloud service in order to request the token. This assignment is performed when the application client is created.

The scope name will differ for each environment. Please ensure the correct value is used for your environment.

To generate a token, you will need to invoke the appropriate IDCS REST API. The curl command in Linux that describes the POST that will return a token is as follows:

```
curl -H 'Authorization: Basic <base64(clientId:clientSecret)>' -H 'Content-Type:
application/x-www-form-urlencoded;charset=UTF-8' --request POST <IDCS URL>/
oauth2/v1/token -d 'grant_type=client_credentials&scope=<EICS Scope>'
```

In Windows, use double-quotes, as follows:

```
curl -H "Authorization: Basic <base64(clientId:clientSecret)>" -H "Content-Type:
application/x-www-form-urlencoded;charset=UTF-8" --request POST <IDCS URL>/
oauth2/v1/token -d "grant_type=client_credentials&scope=<EICS Scope>"
```

This is a standard REST POST, with the following details:

- <IDCS URL> is the IDCS URL the retailer provided
- Include the Client Id and Client Secret as a Basic Authentication header
- Specify the Content Type as application/x-www-form-urlencoded;charset=UTF-8
- Specify the body as grant_type=client_credentials&scope=<EICS Scope>

The service will respond with the following JSON message:

{

"access_token": "<TOKEN>",

"token_type": "Bearer",

"expires_in": 3600

}

Note that the response will return how long the token is valid for. You should reuse the same token until it expires in order to minimize calls to IDCS to get a token.

If the token request fails, you will receive the following JSON response:

{

"error":"<error>",

"error_description":"<error description>",

"ecid":"u….."

}

The most common errors are:

- **Invalid Client**. This means that the client information you send in is not correct. The error description will expand on the reason:
  - **Client Authentication Failed** means that the client is valid, but the client secret is incorrect.
  - **Invalid OAuth Client <CLIENT>** means that the client id is not valid, and the invalid client will be listed in the error message.
- **Invalid Request**. Some part of the inbound request is not valid. The error description is usually descriptive about what the actual error condition is

Calling the EICS Web Service

To invoke the web service with an OAuth2 token, you must add an **Authorization** header to the request. The value of the Authorization header must be **Bearer <token>**, that is:

- The word **Bearer**
- A space
- A valid token

For a REST service call, the request might look something like this:

```
curl -X POST -H 'Content-Type: application/json' -H 'Authorization: Bearer
<TOKEN>' -i https://CloudServiceURL --data '{PAYLOAD}'
```

Remember that the token will expire after a specific amount time, and to be more efficient you should always use a token so long as it's valid. It is your responsibility to make sure that you are keeping track of whether the token is still valid. Your pattern should be:

- Check to see if you have a valid token that has not expired.
- If not, call to IDCS and get a new token. Store it and its expiration time.
- Send the request into the web service with the token in the **Authorization** header as a **Bearer** token.

REST WEB Services Basic Design Principles

- Requests and Responses
- API Versioning
- Content-Type
- JSON Validation
- Synchronous vs Asynchronous
- Configured System Options In EICS
- External vs Internal Attributes
- Dates In Content
- Links In Content

Requests and Responses

When making requests and processing responses from REST web services it is important for the client to handle headers correctly.

The client should always use *Accept* for the appropriate content type when making requests.

The client should always check the response status code and *Content-Type* header before processing a response body.

When reading a payload from the response body, the *Content-Length* header must be used safely and securely along with the *Content-Type*.

This is important even for error responses. It is possible for errors to occur outside of the REST API layer, which may produce different content for the error. In these cases, it is common to get text or HTML content for the response body.

API Versioning

Accept-Version

The REST end points have an optional API versioning feature allowing the client to specify an API version to be accepted.

This may be used by the client to ensure that no calls may be made to a web service that uses an incorrect version number.

For example: *Accept-Version: 22.1.301*

If the web service does not support this API version, then the server will produce a *400 Bad Request* error response.

Content-Type

application/json

The content type of both REST input and returned output is *application/json*.

In the case that no content is included, a content type may not be assigned.

When handling REST service responses, the client must always check the returned *Content-Type* and *Content-Length* before processing the payload.

JSON Validation

When consuming a REST service end point that requires a request payload as JSON, the client is responsible for verifying that the JSON is valid. If invalid data is sent in a request, there may be a server error processing the JSON or it may ignore some fields if the JSON is valid but does not map correctly to the API payload definition.

Always make sure that the client sends valid JSON that is designed to satisfy the API payload definition.

Synchronous vs Asynchronous

Each service API will be defined as synchronous or asynchronous. Both perform JSON validation as described above. If the API is synchronous, the remaining data validation and live updating of the data will take place immediately and the call will be rejected if any business errors occur. If the API is asynchronous, the data is set aside to be processed later and the REST service is successfully returned noting that the data has been accepted. In the case of asynchronous processing, business error and failed data recovery is monitored and the dealt with outside of the REST web service.

Online Documentation

EICS has exposed Swagger JSON files for customers to download and use with the Swagger application as online documentation. The table below contains the files name for access. The file can be accessed using the following URL format from the server:

```
https://<external_load_balancer>/<cust_env>/siocs-int-services/public/api/file.json
```

Filename
activitylock
address
batch
differentiator
finisher
fts
inventoryadjustment
item
iteminquiry
iteminventory
itemisn
itemprice
itemuda
itemuin
manifest

Filename
notification
postransaction
productgroup
reasoncode
security
shipping
stockcount
store
storeitem
storeorder
storesequence
supplier
ticket
transfer
transfershipment
translation
vendordelivery
vendorshipment
warehouse

For external APIs that must be implemented by the customer for EICS to access, the files can be accessed on the server at:

```
https://<external_load_balancer>/<cust_env>/siocs-int-services/public/api-external/file.json
```

Filename
ticketprint-external

Configured System Options In EICS

Web services apply system configuration to the request that are coming in through a web service but assumes that all in-put validation that requires user interaction to confirm has been completed by the third party system prior to accessing the service. It operates as if the user confirmed any activity. However, if a system option is a fixed restriction that does not require user interaction, and the input fails the restriction, this is always considered an error.

Examples of configurations being applied include:

Shipping inventory when inventory is less than 0 can be allowed by a user of EICS. The web services assumes that the application accessing the service did prompt the user or that their business always allows the user to this activity.

Adding a non-ranged item requires both a system configuration option to be enabled and the user to confirm the addition of the item. If the system configuration does not allow it, the web service will block the transaction and return an error (un-less processing asynchronously). If

the system configuration does allow adding non-ranged items, it will automatically assume that a user confirmed this addition and processing will allow the addition of the item.

Allowing Receiver Unit Adjustments is dependent on a period of time. If a receiver unit adjustment were to come into EICS after that period of time, it would automatically be rejected, and the web service would return an error regardless of presentation or confirmation of user done by the external system.

External vs Internal Attributes

EICS web services are EICS centric and track information from an internal application point-of-view. This has ramifications on three types of data: identifiers, dates, and users.

Almost all paths and information will contain an identifier. In almost all cases, this will be an EICS internal identifier generated within our system. If external identifiers also exist for the date, they will be defined as such in the information. For example, you might encounter transferId and externalTransferId as attributes. In some cases, an API only takes an internal identifier, and you may need to use lookup APIs to retrieve an internal identifier using an external identifier as search criteria.

Timestamps are captured at the time an event occurs within EICS as part of EICS's internal tracking and state management. For example, we capture the timestamp when a shipment is created, last updated, and when it is dispatched. These timestamps occur at the time this occurred within EICS. When a REST service is called to create a transaction, such as a shipment, the create timestamp of the shipment will be the moment that service is called. If the shipment is dispatched using the web service, the dispatch date will be the moment that service is called. So if an external system dispatched a shipment two days earlier, and is just now calling the web service, it will not capture the external dispatch time. In some places, you will encounter a date that can be entered as part of the input information (for example, an externalDispatchDate, or simple a transactionTimestamp). If it is part of the input information, then it will be captured as that attribute defined in the API.

The user responsible for actions is often captured as part of transaction information with EICS. Some examples might be the user that created the data, the user the last updated it, or perhaps the user that approved it. In these cases, the user is considered an internal user as is assigned the current session user at the time the activity takes place. When accessing the REST service, the user will be fiwed as an “External User” to indicate it came from an external system, and not the user that manipulated the information in an external system. If external users are to be captured by the data, there will be independent attribute fields such as externalCreateUser that capture the identity of a user in an external system.

Dates In Content

Dates includes in JSON must be in the following format: 2022-04-19T23:59:59-05:00

Dates included as a query parameter must be in the following format: 220227152543-0700

> **Note:**
>
> After the format permissible length, any additional trailing characters will be ignored and the date will still be processed.

Links In Content

JSON information for a data object may include links. These are self-referential APIs that defined other APIs that are available with the information. In the example below, when reading an activity lock, the following links were included that define a path to accomplish other calls. HRef lists the basic reference and the “rel” the remainder of the path. So, when you read activity lock (1), you get a delete reference that maps to */activitylock/1/delete*, defining the REST path to delete that lock.

[{

"links" : [{

"href" : "/activitylocks/1",

"rel" : "self"

}, {

"href" : "/activitylocks/1",

"rel" : "delete"

}],

Hypertext Transfer Protocol Status Codes

The following information documents the HTTP status codes that Oracle returns via web services calls.

Success Codes

Successful codes are returned whenever the accessing client call was made without any error in the form or content.

Code	Description
200 OK	The information supplied by the customer was in a correct form. This code is returned when reading a resource or querying information about an existing resource or schema. This response code is used when the access is synchronous.
202 Accepted	The information supplied by the customer as in a correct form. This code is returned when access is asynchronous.
204 No Content	The information supplied by the customer was in a correct form. The request was successful but the API itself never provides information as a response.

Client Failure Codes

Client failure codes indicate the client made an error in their service access and must correct their code or its content to fix the failure.

Code	Description
400 Bad Request	If this code is returned, it indicates the customer made a call with invalid syntax or violated the defined properties of the input information. Detailed information may be returned that further identifies the error.
401 Unauthorized 403 Forbidden	If either of these codes are returned, it indicates the access to service was denied. This may occur if no OAuth2 token was provided, or the token had expired, or the identity the token was generated for did not have sufficient access.
404 Not Found	If this code is returned, it indicates the customer made an erroneous access call against a resource or schema that is not defined.
405 Method Not Allowed	If this code is returned, it indicates that the wrong HTTP method was used to make the call. Please check the API.
406 Not Acceptable	If this code is returned, it indicates the wrong Accept header value was used to make the call. Please check the API.
409 Too Many Requests	If this code is returned, it indicates that the web service has received too many service requests recently. This may indicate a cloud issue requiring support to address or the client is making too many calls too frequently and a solution may be required to avoid the issue.

System Failure Codes

System failure codes are returned whenever the processing server encounters an unexpected or severe failure.

Code	Description
500 Internal Server Code	A server error occurred that did not allow the operation to complete.
502 Bad Gateway 503 Service Unavailable 504 Gateway Timeout	If any of these codes are returned, it indicates an issue with the network or cloud services, which may occur due to either client or cloud networking or infrastructure issues, such as outages.

JSON Error Element and Error Codes

If an error occurs in the form of the content, or during processing of the content, an HTTP response code (400 Bad Request) will be returned along with JSON structure defined by the ErrorList schema.

Schema — ErrorList

This section describes the ErrorList schema.

Table 4-1 ErrorList — Object

Element Name	Required	Data Type	Description
errors	NA	object	Information about the errors.

The table below describes the information about the error. See the *Oracle Retail Enterprise Inventory Cloud Service Administration Guide* for further information.

Table 4-2 Error — Object

Element Name	Required	Data Type	Description
code	NA	number example: 1	The code error type.
description	NA	string example: AN_ERROR_TOOK_PLACE	A brief description of the error type (an error key).
dataElement	NA	string example: item	The attribute or element that caused the failure.
dataValue	NA	string example: 1234	The value of the attribute or element that failed, or a piece of information about that element.
referenceElement	NA	string example: inventory adjustment	An attribute or element that may help further identify the error. Most often this is a containing element one level above the failed element.
referenceValue	NA	string example: 123	The value of the reference attribute or element, often something to identify the specific reference element.

Example Value

```
[
  {
    "errors": [
      {
        "code": 1,
        "description": "AN_ERROR_TOOK_PLACE",
        "dataElement": "item",
        "dataValue": "1234",
        "referenceElement": "inventory adjustment",
        "referenceValue": "123"
      }
    ]
  }
]
```

Error Attribute Definitions

Attribute	Definition
Code	A numeric code indicates the issue. See Integration Error Codes table.
Description	The name of the error or issue.
DataElement	The name of an attribute or element of the JSON structure that failed.
DataValue	The value of the attribute or element that failed, or a piece of information about the element that failed (such as a maximum value).
ReferenceElement	The name of an attribute or element that will help further identify the data element. Most often the containing element one level above the failed elements (such as a transaction header).

Attribute	Definition
ReferenceValue	The value of the attribute or element that will help further identify the data element.

Integration Error Codes

The following table contains a listing of the error codes that can be found within returned error information.

Code	Name	Issue
1	Business Error	A business processing error prevent the service from completing.
2	Date Range Error	The date range has a problem (usually indicates end date is earlier than start date in a date range.
3	Duplicate Error	Indicates duplicate element within the data is not permitted.
4	Forbidden	Access is not allowed to the service.
5	Internal Server Error	A severe error occurred with the service attempting to process the request.
6	Invalid Input	Most often this indicates that input was included that is not allowed or not needed, however it also doubles a kind of catch-all category.
7	Invalid Format	An input was in an invalid format (most often a date string in a query parameter not being in a valid date format).
8	Invalid Status	A transaction or entity is not in a valid status to proceed with the request.
9	Missing Path Element	A path element defining the path of the resource URL was not present.
10	Missing Attribute	A required attribute was missing on the input to the service.
11	Not Found	A data element in the input could not be found in the system (most often an invalid identifier).
12	No Data Input	No input exists for a service that requires input.
13	No Query Input	No query input exists at all for a query that requires at least one input.
14	Element Too Large	An input was too large (exceeded maximum count or maximum size).
15	Results Too Large	The results of the service were too large to return.

Error Code Data Elements

The following table contains a listing of likely or possible data elements that would be matched with a code. Data element and value may not be returned in all cases.

Code	Name	Data Element	Data Value
1	Business Error	Business exception name/key	Data Value
2	Date Range Error	Date element name	Value of date
3	Duplicate Error	Duplicate element name	Duplicated Value
4	Forbidden	N/A	-

Code	Name	Data Element	Data Value
5	Internal Server Error	N/A	-
6	Invalid Input	Element name	Value of element
7	Invalid Format	Element name	Value of element
8	Invalid Status	Element name	Status of element
9	Missing Path Element	Missing element	-
10	Missing Attribute	Required element name	-
11	Not Found	Element not found	Value of element not found
12	No Data Input	Missing element	-
13	No Query Input	N/A	-
14	Element Too Large	Element name	Allowed size limit
15	Results Too Large	N/A	-

REST Service: Activity Lock

This service allows the creation, removal, and finding of activity locks.

An activity lock is a record indicating the user, time, and a piece of information (a transaction) that should be considered "locked". All server processing validates that the accessing user has a lock on the information before updating, notifying the current user if someone else has modified the information while they were locked and preventing the stale update.

Developers should create locks on transactional information prior to performing update calls and delete locks when the update if finished. For example, create a lock on inventory adjustment with ID 123 with the ActivityLock service, then use `saveInventoryAdjustment`in the Inventory Adjustment service with Adjustment 123, and then delete the activity lock using the ActivityLock service. If you do not gain the lock, you will receive an error when attempting to save an inventory adjustment.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/activitylocks

API Definitions

API	Description
Find Lock	Search for activity lock information based on a set of criteria.
Create Lock	Create a user activity lock.
Delete Lock	Remove a user activity lock.
Read Lock	Retrieve complete information about an activity lock.

API: Find Lock

Searches for locks based on input criteria. At least one input criteria should be provided.

If the number of activity locks found exceeds 10,000, a maximum limit error will be returned. Additional or more limiting search criteria will be required.

API Basics

Endpoint URL	{base URL}/find
Method	POST
Successful Response	200 OK
Processing Type	Synchronous
Input	Criteria
Output	List of activity locks
Max Response Limit	10,000

Input Data Definition

Attribute	Data Type	Required	Description
activityType	String (40)	No	A type of activity that is locked (see Additional Data Definition).
sessionId	String (128)	No	The unique identifier of the session that owns the lock.
deviceType	Integer (4)	No	The device type (see Additional Data Definition).
userName	String (128)	No	The unique identifier of the user that owns the lock.
lockDateFrom	Date	No	Start date of a range during which the activity was locked.
lockDateTo	Date	No	End date of a range during which the activity was locked.

Example Input

{

"activityType": "3",

"sessionId": "sessionTest",

"deviceType":3

}

Output Data Definition

Attribute	Data Type	Description
lockId	Long	The identifier of an activity lock.
sessionId	String	The identifier of the session that owns the lock.
activityId	String	The identifier of the activity that is locked.
activityType	Integer	The type of activity that is locked.
deviceType	Integer	The device type.

Attribute	Data Type	Description
userName	String	The identifier of the user that owns the lock.
lockDate	Date	The date the activity was locked.

Example Output

[{
"links" : [{
"href" : "/activitylocks/1",
"rel" : "self"
}, {
"href" : "/activitylocks/1",
"rel" : "delete"
}],
"lockId" : 1,
"sessionId" : "sessionTest",
"activityId" : "2",
"activityType" : 3,
"deviceType" : 3,
"userName" : "admin",
"lockDate" : "2023-01-04T08:59:41-06:00"
}]

Additional Data Definitions

Location Type

Value	Definition
1	Bill Of Lading
2	Direct Delivery Invoice
3	Fulfilment Order
4	Fulfilment Order Delivery
5	Fulfilment Order Pick
6	Fulfilment Order Reverse Pick
7	Inventory Adjustment
8	Inventory Adjustment Reason
9	Item Basket
10	Item Request
11	POS Transaction Resolution
12	Price Change

Value	Definition
13	Product Basket
14	Product Group
15	Product Group Schedule
16	Replenishment Gap
17	Shelf Adjustment
18	Shelf Replenishment
19	Shipment Reason
20	Stock Count Child
21	Store Order
22	Ticket
23	Ticket Format Basket
24	Transaction Event
25	Transfer
26	Transfer Delivery Carton
27	Transfer Shipment Carton
28	Vendor Delivery Carton
29	Vendor Shipment Carton
30	Vendor Return

Device Type

Value	Definition
1	Client
2	Server
3	Integration Service

API: Create Lock

Used to create a new user transaction activity lock. This prevents two users from simultaneously changing the same data.

API Basics

Endpoint URL	{base URL}
Method	POST
Successful Response	200 OK
Processing Type	Synchronous
Input	Activity Lock object
Output	Activity lock identifier
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
sessionId	String(128)	Yes	The unique identifier of the session that owns the lock.
activityId	String(128)	Yes	The unique identifier of the activity that is locked. This is often a primary identifier of a transaction.
activityType	Integer	Yes	The type of activity that is locked. This is often a transaction type (see Additional Data Definition).

Example Input

{

"activityType": 5,

"sessionId": "session01",

"activityId":"35"

}

Output Data Definition

Attribute	Data Type	Required	Description
lockId	Long	Yes	The unique identifier if a new activity lock is created, or null if the lock already exists.

Example Output

{

"lockId" : 2

}

API: Delete Lock

Used to remove a lock and indicates the activity should no longer be restricted and another user can now begin activity on that data.

It will remove the lock if it exists and perform no action if the lock does not currently exist. In either case, it returns 204 No Content.

API Basics

Endpoint URL	{base URL}/{activityLockId}/delete
Method	DELETE
Successful Response	204 No Content
Processing Type	Synchronous
Input	None

Output	None

Attribute	Description
activityLockId	The activity lock identifier to be removed.

API: Read Lock

Used to retrieve full information about a lock.

API Basics

Endpoint URL	{base URL}/{activityLockId}
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	None
Output	Activity Lock record
Max Response Limit	N/A

Attribute	Description
activityLockId	The activity lock identifier to be read.

Output Data Definition

Attribute	Data Type	Description
lockId	Long	The identifier of an activity lock.
sessionId	String	The identifier of the session that owns the lock.
activityId	String	The identifier of the activity that is locked.
activityType	Integer	The type of activity that is locked.
deviceType	Integer	The device type.
userName	String	The identifier of the user that owns the lock.
lockDate	Date	The date the activity was locked.

Example Output

{

"lockId": 4,

"sessionId": "session01",

"activityId": "37",

"activityType": 5,

"deviceType": 3,

"userName": "dev",

"lockDate": "2023-01-04T23:52:08-06:00"

REST Notification

This page captures the service APIs related to accepting notification from an external system.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/notifications

APIs

API	Description
findNotifications	Find notifications
createNotifications	Create a series of notifications

API: findNotifications

API is used to search for notifications.

Table 4-3 API Basics

Endpoint URL	{base URL}
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	Query parameters
Output	Collection of notifications
Max Response Limit	20,000

Table 4-4 Query Parameters

Attribute	Type	Definition
storeId	Long(10)	Include only records with this store identifier.
notificationType	Integer(4)	Include only records with this notification type.
transactionType	Integer(4)	Include only records with this transaction type.
status	Integer(4)	Include only records with this status.
createDateFrom	Date	Include only records with a create date on or after this date.
createDateTo	Date	Include only records with a create date on or before this date.

Attribute	Type	Definition
notificationId	Long(15)	The unique identifier of the notification.
notificationType	Integer(4)	The type of notification. See Index.
storeId	Long(10)	The store identifier.
subject	String(400)	The title or subject of the notification.
message	String(2000)	The message text of the notification.
status	Integer(4)	The current status of the notification.
transactionType	Integer(4)	The type of a transaction associated to the notification. See Index.
transactionId	Long(15)	The identifier of a transaction associated to the notification.
createDate	Date	The date the notification was originally created.
createUser	String(128)	The user that created the notification.

API: createNotifications

Create one or more notifications within the system.

Payload	Type	Req	Definition
notifications	Collection	X	The list of notifications to create.

Table 4-5 Notification

Attribute	Type	Req	Definition
storeId	Long(10)	X	The store identifier.
notificationType	Integer(4)	X	The type of notification. See Index.
subject	String(400)	X	The title or subject of the notification.
message	String(2000)	X	The message text of the notification.
transactionType	Integer(4)	X	The type of a transaction associated to the notification. See Index.
transactionId	Long(15)	X	The identifier of a transaction associated to the notification.
createDate	Date		The date the notification was originally created. This will default to current date if not supplied.
createUser	String(128)		The user that created the notification. This will default to integration user if not supplied.

Example

```
{
    "notifications": [
        {
            "storeId": 5000,
            "subject": "New
Subject 2",
            "message": "New
Message 2",
            "notificationType":
8,
            "transactionType":
6,
            "transactionId":
5555,
            "createDate":
"2024-02-16T12:00:00-06:00"
        }
    ]
}
```


Index

Table 4-6 Notification Type

ID	Description
1	Customer Order (New)
2	Customer Order Reauthorization
3	Customer Order Pick Reminder
4	Customer Order Pickup
5	Customer Order Receipt
6	Customer Order Reminder
7	Fulfillment Order Reverse Pick (New)
8	UIN Items on Incoming ASN Failed
9	Return To Vendor Expiration Approaching
10	Transfer Delivery Damaged
11	Transfer Delivery Receiving Discrepancy
12	Transfer Delivery Carton Misdirected
13	Transfer Delivery Overdue
14	Transfer Delivery UIN Discrepancy (From Finisher)
15	Transfer Delivery UIN Discrepancy
16	Transfer Delivery Auto-Receive Failure (From Finisher)
17	Transfer Delivery Auto-Receive Failure (From Warehouse)
18	Transfer Delivery Auto-Receive Failure (From Store)
19	Transfer Request Approved
20	Transfer Request Quantity Unavailable
21	Transfer Request Expiration Approaching

Table 4-6 (Cont.) Notification Type

22	Transfer Request Submitted
23	Transfer Request Rejected
24	UIN Store Unexpectedly Altered
25	Vendor Delivery Over Received Quantity
26	Vendor Return Quantity Unavailable

Table 4-7 Transaction Type

ID	Description
1	Customer Order
2	Customer Order Pick
3	Customer Order Delivery
4	Customer Order Reverse Pick
5	Direct Store Delivery Receiving
6	Inventory Adjustment
7	Item Basket
8	Return To Vendor
9	Return To Vendor Shipment
10	Shelf Adjustment
11	Shelf Replenishment
12	Shelf Replenishment Gap
13	Stock Count
14	Store Order
15	Transfer
16	Transfer Shipment
17	Transfer Receiving

Table 4-8 Notification Status

ID	Description
1	Unread
2	Read

REST Service: Address

This service integrates address foundation data. Asynchronous address integration is processed through staged messages and is controlled by the MPS Work Type: DcsStore.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/addresses

API Definitions

API	Description
readAddress	Read the information about a single address.
importAddress	Create or update the information about a single address.
deleteAddress	Deletes a single address.

API: Import Address

Import a series of addresses. This allows up to 1,000 addresses before an input too large error is returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/import
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	Addresses list
Output	None
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
addresses	List of details	Yes	A list of addresses to import.

Address Detail Data Definition

Attribute	Data Type	Required	Description
addressId	String(25)	Yes	The external identifier of the address.
entityType	Integer	Yes	Donates the type of location of the entity (see Additional Data Definition).
entityId	Long(10)	Yes	The external identifier of the entity (this will also match internal identifiers).
addressType	Integer	Yes	The type of address: (see Additional Data Definition).
primary	Boolean		True if this is the primary address of the entity, false otherwise
addressLine1	String(240)	Yes	The first line of the address

addressLine2	String(240		The second line of the address
addressLine3	String(240)		The third line of the address
city	String(120)	Yes	The city of the address
state	String(3)		The state of the address
countryCode	String(3)	Yes	The country code of the address (used by supplier)
postalCode	String(30)		The postal code of the address
county	String(250)		The county of the address
companyName	String(120)		A company name associated with that address
contactName	String(120)		Contact name for that address
contactPhone	String(20)		Contact phone number for that address
contactFax	String(20)		Contact fax number for that address
contactEmail	String(100)		Contact email for that address
firstName	String(120)		A first name of a contact at that address
lastName	String(120)		A last name of the contact at that address
phoneticFirstName	String(120)		A phonetic spelling of a first name of a contact at that address
phoneticLastName	String(120)		A phonetic spelling of a last name of a contact at that address
supplierLocation	String(120)		Supplier location information

API: Delete Address

Deletes a single address.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/{addressId}/delete
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	None
Output	None
Max Response Limit	N/A

Path Parameter Definitions

Attribute	Description
addressId	The address identifier to be removed

API: Read Address

Used to read a single address.

API Basics

Endpoint URL	{base URL}/{addressId}
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	None
Output	Address record
Max Response Limit	N/A

Path Parameter Definitions

Attribute	Description
addressId	The address identifier to be removed

Output Data Definition

Attribute	Data Type	Description
addressId	String	The external identifier of the address.
entityType	Integer	Donates the type of location of the entity (see Additional Data Definition).
entityId	Long	The external identifier of the entity (this will also match internal identifiers).
addressType	Integer	The type of address: (see Additional Data Definition)
primary	Boolean	True if this is the primary address of the entity, false otherwise
addressLine1	String	The first line of the address
addressLine2	String	The second line of the address
addressLine3	String	The third line of the address
city	String	The city of the address
state	String	The state of the address
countryCode	String	The country code of the address (used by supplier)
postalCode	String	The postal code of the address
county	String	The county of the address

companyName	String	A company name associated with that address
contactName	String	Contact name for that address
contactPhone	String	Contact phone number for that address
contactFax	String	Contact fax number for that address
contactEmail	String	Contact email for that address
firstName	String	A first name of a contact at that address
lastName	String	A last name of the contact at that address
phoneticFirstName	String	A phonetic spelling of a first name of a contact at that address
phoneticLastName	String	A phonetic spelling of a last name of a contact at that address
supplierLocation	String	Supplier location information

Additional Data Definitions

Entity Type

Value	Definition
1	Store
2	Supplier
3	Warehouse
4	Finisher

Address Type

Value	Definition
1	Business
2	Postal
3	Return
4	Order
5	Invoice
6	Remittance
7	Billing
8	Delivery
9	External

REST Service Batch

This service allows an external system to schedule an *adhoc* batch job for execution.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/batches

API Definitions

API	Description
executeBatch	Schedules the batch for immediate execution.
findBatchJobs	Finds all the batch jobs available to schedule.

API: Execute Batch

Schedules the specified batch job for immediate execution.

If parameter date and/or parameter identifier are entered, they are passed as parameters to the batch job identified by the batch name.

See Batch guide for definition of data set identifiers for various batches.

API Basics

Endpoint URL	{base URL}
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	Batch information
Output	None
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
batchName	String (256)	Yes	The name of the batch job to execute.
storeId	Long(10)		A store identifier to run the batch for. If included, it will run the batch processing for a single store. If not included, the batch will run for all stores based on functional description of the batch processing.
parameterDate	Date		A parameter date passed to the batch job (see SIOCS adhoc batch documentation).
parameterId	String		A parameter identifier passed to the batch job (see SIOCS adhoc batch documentation).

API: Find Batch Jobs

Finds all the batch jobs available to schedule.

API Basics

Endpoint URL	{base URL}
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	None
Output	List of batch jobs
Max Response Limit	N/A

Output Data Definition

Attribute	Data Type	Description
jobName	String	The job name used to execute the batch.
shortDescription	String	A short description of the batch job.
longDescription	String	A long description of the batch job.
jobParamHint	String	Some hint text for what parameter values might be.
jobInterval	Integer	The execution interval of the batch job (see Additional Data Definition).
batchType	Integer	The type of batch job (see Additional Data Definition).
storeRelated	Boolean	Y indicates the batch job requires store level processing.
enabled	Boolean	Y indicates the batch job is currently enabled and scheduled. This will not prevent batch execution via this service.
lastExecutionTime	String	The timestamp of the last execution of the batch job.
updateDate	String	The last time this record was updated by SIOCS.

Additional Data Definitions

Batch Type

Value	Definition
1	Cleanup
2	Operation
3	System
4	System Cleanup
5	Data Seed
6	Archive

Batch Interval Type

Value	Definition
1	30 Minutes
2	1 Hour
3	2 Hours
4	3 Hours
5	4 Hours
6	6 Hours
7	8 Hours
8	12 Hours
9	24 Hours
10	Monthly

REST Service: Differentiator

This service integrates differentiator foundation data. Asynchronous differentiator integration is processed through staged messages and is controlled by the MPS Work Type: DcsDiff.

This service replaces the RIB flow for differentiators.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/differentiators

API Definitions

API	Description
importDifferentiatorTypes	Create or update differentiator types.
importDifferentiators	Create or update differentiators.
deleteDifferentiatorType	Delete a differentiator type and all associated differentiators.
deleteDifferentiator	Delete a differentiator.

API: Import Differentiator Types

Imports a differentiator type by writing a staged message and processing through DCS consumer.

If the number of records exceed 1000, an input too large error is returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/types/import
Method	POST

Successful Response	202 Accepted
Processing Type	Asynchronous
Input	Differentiator Type information
Output	None
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
differenatiorTypes	List of details	Yes	The differentiator types to import.

Detail Data Definition

differenatiorTypeId	String (10)	Yes	The differentiator type identifier.
description	String (255)	Yes	The differentiator type description (not translated).

API: Import Differentiators

Imports a differentiator.

If the number of records exceed 1000, an input too large error is returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/import
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	Differentiator information
Output	None
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
differenatiors	List of details	Yes	The differentiators to import.

Detail Data Definition

Attribute	Data Type	Required	Description

differenatiorId	String (10)	X	The differentiator type identifier.
differenatiorTypeId	String (10)	X	The differentiator type identifier.
description	String (255)	X	The differentiator type description (not translated).

API: Delete Differentiator Type

Deletes a differentiator type and all associated differentiators.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/types/{differenatiorTypeId}/delete
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	None
Output	None
Max Response Limit	N/A

Path Parameter Definitions

Attribute	Description
differentiatorId	The differentiator Id to be removed

REST Service: Finisher

This service integrates finisher and finisher item foundation data. Asynchronous finisher integration is processed through staged messages and is controlled by the MPS Work Type: DcsPartner. Asynchronous finisher item integration is processed through staged messages and is controlled by the MPS Work Type: DcsItemLocation.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/finishers

API Definitions

API	Description
importFinishers	Imports a collection of finishers.
deleteFinisher	Deletes a finisher.

importItems	Imports a collection of finisher items.
removeItems	Marks finisher items for deletion.

API: Import Finishers

Imports finishers. This allows 500 finishers per service call.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/import
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	List of finisher import
Output	None
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
finishers	List of details	Yes	A list of finishers to import

Detail Data Definition

Attribute	Data Type	Required	Description
finisherId	Long (10)	Yes	The finisher identifier.
name	String (240)		The finisher name.
status	Integer		Finisher Import Status (see Additional Data Definition).
currencyCode	String (3)		ISO currency code used by the finisher
countryCode	String (3)		The ISO country code assigned to the finisher.
languageCode	String (6)		The ISO language code of the finisher
contactName	String (120)		Name of the finisher's representative contact.
contactPhone	String (20)		Phone number of the finisher's representative contact.
contactFax	String (20)		Fax number of the finisher's representative contact.

contactTelex	String (20)	Telex number of the finisher's representative contact.
contactEmail	String (100)	Email address of the finisher's representative contact.
manufacturerId	String (18)	Manufacturer's identification number
taxId	String (18)	Tax identifier number of the finisher.
transferEntityId	String (20)	Identifier of the transfer entity that the finisher belongs to.
paymentTerms	String (20)	Payment terms for the partner
importCountryCode	String (3)	The ISO country code of the import authority.
importPrimary	Boolean	True Indicates the code is the primary import authority of the import country.

API: Delete Finisher

Deletes a finisher.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/{finisherId}/delete
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	None
Output	None
Max Response Limit	N/A

Path Parameter Definitions

Attribute	Description
finisherId	The finisher identifier to be removed.

API: Import Items

Imports finisher items. This allows 5000 items per service call.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/{finisherId}/items/import
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	Finisher Item list
Output	None
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
items	List of details	Yes	A list of items to import.

Detail Data Definition

Attribute	Data Type	Required	Description
itemId	String (25)	Yes	The item identifier.
status	Integer	Yes	Finisher Item Import Status (see Additional Data Definition).

API: Remove Items

Marks finisher items for later deletion. This allows 5000 items per service call.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/{finisherId}/items/remove
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	Items list
Output	None
Max Response Limit	N/A

Path Parameter Definitions

Attribute	Description
finisherId	The finisher identifier to be removed.

Input Data Definition

Attribute	Data Type	Required	Description
itemIds	List<String>	Yes	A list of items to remove.

Additional Data Definitions

Finisher Import Status

Value	Definition
1	Active
2	Inactive

Finisher Item Import Status

Value	Definition
1	Active
2	Discontinued
3	Inactive

REST Service: Inventory Adjustment

This service defines operations to manage Inventory Adjustment information.

Service Base URL

The Cloud service base URL follows the format:

https://external_load_balancer/cust_env/siocs-int-services/api/invadjustments

APIs

API	Description
Find Adjustments	Search for transactional store inventory adjustments summary headers based on input criteria.
Read Adjustment	Reads a transaction store inventory adjustment.
Update Adjustment	This API updates the details of an inventory adjustment that is currently "In Progress."
Confirm Adjustment	This API is used to confirm an inventory adjustment, finalizing it and processing the inventory movement.
Cancel Adjustment	This API is used to cancel an inventory adjustment.
Create Adjustment	This API is used to create a new manual inventory adjustment whose status is In Progress.

API: Find Adjustments

API is used to search for transaction headers for inventory adjustments. No items or detailed information is returned via this service.

API Basics

Endpoint URL	{base URL}
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	Query parameters
Output	Collection of adjustments
Max Response Limit	10,000

Query Params

Attribute	Type	Req	Definition
referenceId	Long(12)	No	Return adjustments containing this reference identifier.
status	Integer(2)	No	Returns adjustments containing this status: See Index.
adjustmentDateFrom	Date	No	Returns adjustments where the adjustment date is on or after this date.
adjustmentDateTo	Date	No	Returns adjustments where the adjustment date is on or before this date.
updateDateFrom	Date	No	Returns adjustments where the adjustment date is on or after this date.
updateDateTo	Date	No	Returns adjustments where the adjustment date is on or before this date.
itemId	String(25)	No	Returns adjustments containing this item.
reasonId	Long(12	No	Returns adjustments containing this reason code.

Output Data Definition

Attribute	Type	Definition
adjustmentId	Long(12)	Adjustment identifier
storeId	Long(10)	Store Identifier
referenceId	Long(12)	Identifier of original adjustment it was copied from
externalId	String(128)	A unique identifier of the adjustment in an external system

adjustmentDate	Date	The timestamp the inventory is official considered to have been adjusted
status	Integer(2)	The adjustment status: See Index.
createDate	Date	The date the adjustment was created in EICS
updateDate	Date	The date the adjustment was last updated by EICS
approveDate	Date	The date the adjustment was approved in EICS

API: Read Adjustment

This API is used to read an inventory adjustment.

API Basics

Endpoint URL	{base URL}/{adjustmentId}
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	N/A
Output	Inventory Adjustment
Max Response Limit	N/A

Path Parameter Definition

Payload	Type	Definition
adjustmentId	Long(12)	The unique inventory adjustment identifier

Output Data Definition

Attribute	Type	Definition
adjustmentId	Long(12)	The unique inventory adjustment identifier
storeId	Long(10	The unique identifier of the store
referenceId	Long(12)	Identifier of original adjustment it was copied from
externalId	String(128)	Identifier to the adjustment in an external system
adjustmentDate	Date	The timestamp the inventory is official considered to have been adjusted
status	Integer(2)	The adjustment status: See Index
comments	String(2000)	Comments associated to the adjustment
createDate	Date	The date the adjustment was created by EICS
createUser	String (128)	The user that created the adjustment in EICS
updateDate	Date	The date the adjustment was last updated by EICS
updateUser	String (128)	The user that last updated the adjustment in EICS

approveDate	Date	The date the adjustment was approved in EICS
approveUser	String (128)	The user that approved the adjustment in EICS
externalCreateUser	String (128)	The non-EICS user that created the inventory adjustment in the external system
externalUpdateUser	String (128)	The non-EICS user that updated the inventory adjustment in the external system
lineItems	Collection	A collection of Inventory Adjustment Line Items

Inventory Adjustment Line Item

Payload	Type	Definition
lineId	Long(12)	The unique SIOCS identifier of the line item
itemId	Long(25)	The sku level item identifier
reasonId	Long(12)	The unique internal identifier of the reason code
caseSize	BigDecimal(10,2)	Case size associated to the line item
quantity	BigDecimal(20,4)	Quantity associated to the line item
uins	Collection{String}	The UINs associated to the item quantities

API: Update Adjustment

This API updates the details of an inventory adjustment that is currently "In Progress"

API Basics

Endpoint URL	{base URL}/{adjustmentId}
Method	POST
Successful Response	204 OK
Processing Type	
Input	Inventory Adjustment
Output	N/A
Max Response Limit	

Path Parameter Definition

Payload	Type	Definition
adjustmentId	int12	The unique inventory adjustment identifier.

Input Data Definition

Attribute	Type	Req	Definition
referenceId	Big Decimal(12)		Identifier of original adjustment from which it was copied.
externalUser	String(128)		A non-EICS user that created the inventory adjustment in the external system.

adjustmentDate	Date		The timestamp the inventory is officially considered to have been adjusted. If left blank, it will default to the creation date of the record.
comments	String(2000)		Comments associated to the adjustment. These comments will replace any previous comments. Comments will be trimmed down to 2000 characters if the entry is larger.
lineItems	Array	X	A collection of line items attached to the adjustment.

Inventory Adjustment Line Item

Payload	Type	Req	Definition
itemId	String(25)	X	The SKU level item identifier. If SKU is already present, its line item will be updated. Otherwise, a new line item will be added.
reasonId	BigDecimal(12)	X	The unique identifier of the reason code associated to the adjustment. The reason code is not modifiable on an update, but it will be used if a new item is added.
quantity	BigDecimal(20,4)	X	The quantity of this line item. For UINS, the quantity must match the number of UINs. If the quantity is set to 0, the system will attempt to remove the line item from the adjustment
caseSize	BigDecimal(10,2)		Case size associated to the line item
uins	array{String}(128)		The UINs to attach to this line item.

Responses

Code	Description
204	No Content

400	Bad Request
	Example:
	
	[
	{
	"errors": [
	{
	"code": 1,
	"description": "AN_ERROR_TOOK_PLACE",
	"dataElement": "item",
	"dataValue": "1234",
	"referenceElement": "inventory adjustment",
	"referenceValue": "123"
	}
	]
	}
	]

API: Confirm Adjustment

This API is used to confirm an inventory adjustment, finalizing it and processing the inventory movement.

API Basics

Endpoint URL	{base URL}/{adjustmentId}/confirm
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	N/A
Output	N/A
Max Response Limit	N/A

Path Parameter Definition

Payload	Type	Definition
adjustmentId	Long(12)	The unique inventory adjustment identifier

API: Cancel Adjustment

This API is used to cancel an inventory adjustment.

API Basics

Endpoint URL	{base URL}/{adjustmentId}/cancel
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	N/A
Output	N/A
Max Response Limit	N/A

Path Parameter Definition

Payload	Type	Definition
adjustmentId	Long(12)	The unique inventory adjustment identifier

API: Create Adjustment

This API is used to create a new manual inventory adjustment whose status is In Progress.

API Basics

Endpoint URL	{base URL}
Method	POST
Successful Response	200 OK
Processing Type	Synchronous
Input	Inventory Adjustment
Output	Inventory Adjustment Reference
Max Response Limit	5,000 Items

Input Data Definition

Attribute	Type	Req	Definition
storeId	Long(10)	X	Store Identifier
referenceId	Long(12)		Identifier of original adjustment it was copied from
externalId	String(128)		A unique identifier to the adjustment in an external system
adjustmentDate	Date		The timestamp the inventory is official considered to have been adjusted
externalUser	String(128)		A non-EICS user that created the inventory adjustment in the external system
comments	String(2000)		Comments associated to the inventory adjustment
lineItems	Collection	X	collection of inventory adjustment line items

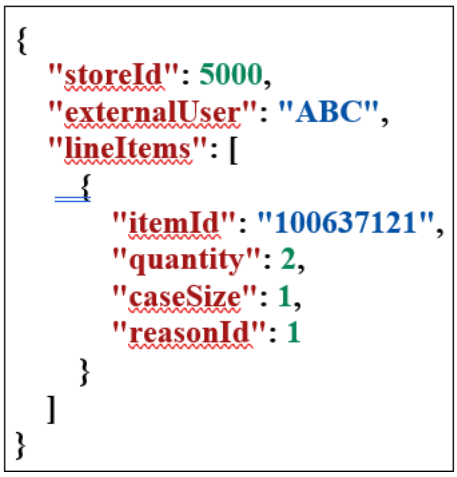
Inventory Adjustment Line Item

Payload	Type	Req	Definition
itemId	String(25)	X	The sku level item identifier. It must be an inventoriable item.
reasonId	Long(12)	X	The unique internal identifier of the reason code.
quantity	BigDecimal(20,4)	X	Quantity associated to the line item
caseSize	BigDecimal(10,,2)		Case size associated to the line item
uins	Collection{String}		The UINs associated to the item quantities

Output Data Definition

Attribute	Type	Definition
adjustmentId	Long(12)	The newly created adjustment identifier

Example Input



```
{
   "storeId": 5000,
   "externalUser": "ABC",
   "lineItems": [
      {
         "itemId": "100637121",
         "quantity": 2,
         "caseSize": 1,
         "reasonId": 1
      }
   ]
}
```

Business Data Errors

Error	Definition
CASE_SIZE_NOT_WHOLE	The adjustment requires a non-decimal case size.
EXISTING_UIN_CANNOT_BE_ADDED	An already existing UIN cannot be added to the store.
NON_INVENTORIABLE_ITEM	The item cannot be adjusted because it cannot have inventory.
NON_MANAGED_STORE	The store does not manage its inventory.
QUANTITY_NOT_WHOLE	The adjustment requires a non-decimal quantity.

Additional Date Definition

Inventory Adjustment Status

ID	Status
1	In Progress
2	Completed
3	Canceled

REST Service: Item

This service integrates the item foundation data with an external application. Asynchronous item integration is processed through staged messages and is controlled by the MPS Work Types.

Note that this is item level foundational data. To lookup or access item information, use the item inquiry REST service.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/items

API Definitions

API	Description
importItems	Import items into the system.
removeItems	Mark items for deletion at some point in the future when no records are using them.
importHierarchies	Imports item hierarchies into the system.
removeHierarchies	Marks hierarchies for deletion at some point in the future when no records are using them.
importRelatedItems	Imports the associations of related items.
deleteRelatedItems	Deletes an association of related items.
importImageUrls	Imports image URLs associated to the item.
deleteImageUrls	Delete image URLs associated to the item.

API: Import Items

Imports items.

This flow is managed in MPS system with the following family: DcsItem.

If the input exceeds more than 100 records, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/import
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	List of Items to import
Output	None
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
finishers	List of details	Yes	A list of items to import

Detail Data Definition

Attribute	Data Type	Required	Description
itemId	String (25)	Yes	The unique item identifier (sku number).

transactionLevel	Long	Yes	Number indicating which of the three levels transactions occur for the item. Items may only be used on transactions with inventory tracked if the transaction level and item level match.
itemLevel	Long	Yes	Number indicating which of the three levels an item resides at. Items may only be used on transactions with inventory tracked if the transaction level and item level match.
departmentId	Long (12)	Yes	The merchandise hierarchy department identifier.
classId	Long (12)		The merchandise hierarchy class identifier.
subclassId	Long (12)		The merchandise hierarchy subclass identifier.
shortDescription	String (255)		A short description of the item.
longDescription	String (400)		A long description of the item.
differentiator1	String (10)		The first differentiator identifier.
differentiator2	String (10)		The second differentiator identifier.
differentiator3	String (10)		The third differentiator identifier.
differentiator4	String (10)		The fourth differentiator identifier.
status	Integer		Item Import Status (see Additional Data Definition).
parentId	String (25)		The unique identifier of the item at the next level above this item.
pack	Boolean		True if the item is pack, false otherwise.
simplePack	Boolean		True if the item is a simple pack, false otherwise.
sellable	Boolean		True if the item is sellable, false otherwise.
orderable	Boolean		True if the item can be ordered from a supplier, false otherwise.
shipAlone	Boolean		True if the item must be shipped in separated packaging, false otherwise.
inventoriable	Boolean		True if the item is inventoried, false otherwise.
notionalPack	Boolean		True indicates the inventory is held at the component level. All notional pack are marked as inventoriable in SIOCS.

estimatePackInventory	Boolean	True if the item allows estimating pack inventory from component positions, false otherwise.
primaryReferenceItem	Boolean	True indicates it the primary sub-translation level item.
orderAsType	Boolean	True indicates a buyer pack is receivable at the pack level. N means at the component level.
standardUom	String (4)	The unit of measure that inventory is tracked in.
packageUom	String (4)	The unit of measure associated with a package size.
packageSize	Double	The size of the product printed (will be printed on the label).
eachToUomFactor	BigDecimal	The multiplication factor to convert 1 EA to the equivalent standard unit of measure.
barcodeFormat	String (4)	The format of a barcode (used for Type 2 barcode items).
barcodePrefix	Long (9)	The barcode prefix used in association with the Type 2 barcode of this item.
wastageType	Integer	Waste Type (see Additional Data Definition).
wastagePercent	BigDecimal	Wastage percent.
wastagePercentDefault	BigDecimal	Default wastage percent.
suggestedRetailCurrency	String (3)	The currency of the manufacturer suggested retail price.
suggestedRetailPrice	BigDecimal	Manufacturer suggested retail price.
brand	String (30)	Brand name of the brand the item belongs to.
brandDescription	String (120)	Brand description of the brand the item belongs to.
components	List of components	A list of components for the item (if the item is a pack)

Component Data Definition

Attribute	Data Type	Required	Description
componentItemId	String (25)	Yes	The item identifier of the component item within the pack.
quantity	BigDecimal	Yes	The quantity of component item within the pack

API: Remove Items

Deactivate items.

This flow is managed in MPS system with the following family: DcsItem.

If the input exceeds more than 500 records, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/remove
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	Items list to remove
Output	None
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
items	List of details	Yes	A list of item reference to update to a non-active status.

Detail Data Definition

Attribute	Data Type	Required	Description
itemId	String (25)	Yes	The unique item identifier
status	Integer	Yes	Item Remove Status (see Additional Data Definition).

API: Import Hierarchies

Imports item hierarchies.

This flow is managed in MPS system with the following family: DcsHierarchy.

If the input exceeds more than 500 records, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/hierarchies/import
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	Items hierarchy list
Output	None
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
hierarchies	List of details	Yes	The hierarchies to import.

Detail Data Definition

Attribute	Data Type	Required	Description
departmentId	Long (12)	Yes	The hierarchy department identifier.
departmentName	String (360)		The hierarchy department name.
classId	Long (12)		The hierarchy class identifier.
className	String (360)		The hierarchy class name.
subclassId	Long (12)		The hierarchy subclass identifier.
subclassName	String (360)		The hierarchy subclass name.

API: Remove Hierarchies

Deactivates item hierarchies. Once no information is associated to the item hierarchies, a cleanup batch will remove them from the database.

This flow is managed in MPS system with the following family: DcsHierarchy.

If the input exceeds more than 500 records, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/hierarchies/remove
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	Items hierarchy list
Output	None
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
hierarchies	List of details	Yes	The images to remove.

Detail Data Definition

Attribute	Data Type	Required	Description
departmentId	Long (12)	Yes	The hierarchy department identifier.
classId	Long (12)		The hierarchy class identifier.
subclassId	Long (12)		The hierarchy subclass identifier

API: Import Related Items

Imports item relationships that an item may belong to.

This flow is managed in MPS system with the following family: DcsItem.

If the input exceeds more than 500 records, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/related/import
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	Items Relationship list
Output	None
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
relationships	List of details	Yes	The relationships to import.

Detail Data Definition

Attribute	Data Type	Required	Description
relationshipId	Long (20)	Yes	The identifier of the relationship.
relationshipType	Integer	Yes	Relationship Type (see Additional Data Definitions).
name	String (120)		The name of the relationship.
itemId	String (25)	Yes	The item whose related records are being recorded.

mandatory	Boolean	Yes	True if the relationships are mandatory.
relatedItems	List of related items	Yes	The related items.

Related Item Data Definition

Attribute	Data Type	Required	Description
itemId	String (25)	Yes	The item that is related.
effectiveDate	Date		Date at which this relationship becomes active.
endDate	Date		Last date at which this relationship is active.
priorityNumber	Long (4)		Number defining priority in the case of multiple substitute items.

API: Delete Related Items

Deletes relationships between items.

This flow is managed in MPS system with the following family: DcsItem.

If the input exceeds more than 500 records, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/related/delete
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	None
Output	None
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
relationshipIds	List<Long>	Yes	The relationship identifiers to remove.

API: Import Image Urls

Import image URLs associated to the item.

This flow is managed in MPS system with the following family: DcsItem.

If the input exceeds more than 500 records, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/images/import
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	Items Image list
Output	None
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
images	List of details	Yes	The images to import.

Detail Data Definition

Attribute	Data Type	Required	Description
itemId	String (25)	Yes	The item identifier/sku number.
imageType	Integer	Yes	Image Type (see Additional Data Definitions).
storeId	Long (10)		The store identifier. This is required only if the image type is QR_CODE.
imageName	String (120)	Yes	The name of the image.
imageSize	String (6)		The size of the image: (T) thumbnail. Other than (T), any text is accepted and there is no definition to validate against, but the text has no meaning.
url	String (1000)	Yes	The universal resource locator of the image.
displaySequence	Integer (2)		The sequence the item should be displayed in.
startDate	Date		The date the image becomes active.
endDate	Date		The date the image ceases being active.

API: Delete Image Urls

Deletes image URLs.

This flow is managed in MPS system with the following family: DcsItem

If the input exceeds more than 500 records, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/images/delete
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	Items Image list
Output	None
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
images	List of details	Yes	The images to remove.

Detail Data Definition

Attribute	Data Type	Required	Description
itemId	String (25)	Yes	The item identifier/sku number.
imageName	String (120)	Yes	The name of the image.
imageType	Integer	Yes	Image Type (see Additional Data Definitions).

Additional Data Definitions

Item Status

Value	Definition
1	Active
2	Discontinued
3	Inactive
4	Deleted
5	Auto Stockable
6	Non Ranged

Item Import Status

Value	Definition
1	Active
5	Auto Stockable

6	Non Ranged

Item Remove Status

Value	Definition
2	Discontinued
3	Inactive
4	Deleted

Wastage Type

Value	Definition
1	Sales Wastage
2	Spoilage Wastage

Relationship Type

Value	Definition
1	Related
2	Substitute
3	Up-Sell
4	Cross-Sell

Image Type

Value	Definition
1	Image
2	QRCode

REST Service: Item Inquiry

This service allows the customer to retrieve information about items. These services are intended to find item themselves and do not retrieve inventory. For inventory queries, see inventory inquiry services.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/iteminquiries

API Definitions

API	Description
findItemBySearchScan	Searches for summary item information based on basic item scan information such as UPC, barcode, and so on.

findItemBySource	Searches for summary item information based on hierarchy or source location search criteria.
findItems	Searches item information based on multiple items (and an optional store).
findItemsByScan()	Searches for item information by an unknown identifier, most likely a scan. It first searches for the item using the searchScan as a potential item, and if found will return records based on that. It then searches as a UPC, barcode, or UIN, in that order, halting if it finds potential matches. An optional store identifier can be included as well.
findItemsByInventory	Searches for summary information about an item based on search criteria, primarily inventory information.
findRelatedItems()	Search for summary information about an item based on search criteria, primarily parent item.
findItemsByUDA()	Search for summary information about an item based on search criteria, primarily user defined attributes (UDA).

API: Find Item by Search Scan

Search for item information by on unknown identifier, most likely a scan. It first searches for the item using the scan itself as the potential item, and if found will return record(s). It then searches as a UPC, barcode, or UIN in that order halting if it finds potential matches.

API Basics

Endpoint URL	{base URL}/scan/{searchScan}
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	None
Output	List of items scanned
Max Response Limit	N/A

Path Parameter Definitions

Attribute	Description
searchScan	An Item, UPC, Barcode or UIN take from a scan.

Query Parameter Definition

Attribute	Data Type	Required	Description
storeId	Long		A store to verify if found item is ranged to.

Output Data Definition

Attribute	Data Type	Description

itemId	String	The item identifier.
type	Integer	Item Type (see Additional Data Definition).
status	String	Item Status (see Additional Data Definition).
shortDescription	String	A short description of the item.
longDescription	String	A long description of the item.
departmentId	Long	The identifier of the department the item belongs to.
classId	Long	The identifier of the class the item belongs to.
subclassId	Long	The identifier of the subclass the item belongs to.
storeId	Long	The store identifier passed in as the query parameter.
ranged	Boolean	True if the item is ranged to the requested store, false otherwise.

API: Find Item by Source

Searches for summary information about an item based on search criteria, primarily hierarchy and source location.

If the number of items found exceeds 10000, a maximum limit error will be returned. Additional or more limiting search criteria will be required.

At least one query parameter is required. If source type location is entered, then a source id for that type must also be entered.

API Basics

Endpoint URL	{base URL}/source
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	None
Output	List of Items summary
Max Response Limit	N/A

Query Parameter Definition

Attribute	Data Type	Description
description	String	Include only items with description text matching this description.

sourceType	Integer	Include only items available from this source type (see Additional Data Definition).
sourceId	String	Include only items available from this source identifier.
departmentId	Long	Include only items associated to this merchandise hierarchy department.
classId	Long	Include only items associated to this merchandise hierarchy class.
subclassId	Long	Include only items associated to this merchandise hierarchy subclass.
storeId	Long	Include only items ranged to this particular store.

Output Data Definition

Attribute	Data Type	Description
itemId	String	The item identifier.
type	Integer	Item Type (see Additional Data Definition).
status	String	Item Status (see Additional Data Definition).
shortDescription	String	A short description of the item.
longDescription	String	A long description of the item.
departmentId	Long	The identifier of the department the item belongs to.
classId	Long	The identifier of the class the item belongs to.
subclassId	Long	The identifier of the subclass the item belongs to.

API: Find Items

Searches for detailed information about the items using the specified input.

If the number of items found exceeds 10000, a maximum limit error will be returned. Additional or more limiting input criteria will be required.

API Basics

Endpoint URL	{base URL}
Method	POST
Successful Response	200 OK
Processing Type	Synchronous
Input	Criteria
Output	List of Items
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
storeId	Long	Yes	The store to retrieve item information for
itemIds	List<String>	Yes	A list of items to retrieve item information for

Output Data Definition

Attribute	Data Type	Description
itemId	String	The unique item identifier (sku number).
transactionLevel	Long	Number indicating which of the three levels transactions occur for the item. Items may only be used on transactions with inventory tracked if the transaction level and item level match.
itemLevel	Long	Number indicating which of the three levels an item resides at. Items may only be used on transactions with inventory tracked if the transaction level and item level match.
departmentId	Long	The merchandise hierarchy department identifier.
classId	Long	The merchandise hierarchy class identifier.
subclassId	Long	The merchandise hierarchy subclass identifier.
shortDescription	String	A short description of the item.
longDescription	String	A long description of the item.
differentiator1	String	The first differentiator identifier.
differentiator2	String	The second differentiator identifier.
differentiator3	String	The third differentiator identifier.
differentiator4	String	The fourth differentiator identifier.
status	Integer	The status (see Additional Data Definition).
parentId	String	The unique identifier of the item at the next level above this item.
pack	Boolean	True if the item is pack, false otherwise.
simplePack	Boolean	True if the item is a simple pack, false otherwise.
sellable	Boolean	True if the item is sellable, false otherwise.
orderable	Boolean	True if the item can be ordered from a supplier, false otherwise.

shipAlone	Boolean	True if the item must be shipped in separated packaging, false otherwise.
inventoriable	Boolean	True if the item is inventoried, false otherwise.
notionalPack	Boolean	True indicates the inventory for the pack is tracked at the component level.
estimatePackInventory	Boolean	True if the item allows estimating pack inventory from component positions, false otherwise.
primaryReferenceItem	Boolean	True indicates it the primary sub-translation level item.
orderAsType	Boolean	True indicates a buyer pack is receivable at the pack level. N means at the component level.
standardUom	String	The unit of measure that inventory is tracked in.
packageUom	String	The unit of measure associated with a package size.
packageSize	Double	The size of the product printed (will be printed on the label).
eachToUomFactor	BigDecimal	The multiplication factor to convert 1 EA to the equivalent standard unit of measure.
barcodeFormat	String	The format of a barcode (used for Type 2 barcode items).
barcodePrefix	Long	The barcode prefix used in association with the Type 2 barcode of this item.
wastageType	Integer	Type of wastage (see Additional Data Definition).
wastagePercent	BigDecimal	Wastage percent.
wastagePercentDefault	BigDecimal	Default wastage percent.
suggestedRetailCurrency	String	The currency of the manufacturer suggested retail price.
suggestedRetailPrice	BigDecimal	Manufacturer suggested retail price.
brand	String	Brand name of the brand the item belongs to.
brandDescription	String	Brand description of the brand the item belongs to.
createDate	Date	The date the item was created in EICS.
updateDate	Date	The last date the item was updated in EICS.

(RANGED INFO)

storeId	Long	The store identifier.
status	String	The item status. (see Additional Data Definition).

primarySupplierId	Long	The unique identifier of the primary supplier of the item to this store location.
storeControlPricing	Boolean	True indicates the item price can be controlled by the store.
rfid	Boolean	True indicates the item is RFID tagged.
defaultCurrencyCode	String	The default currency of the item's price at this store.
purchaseType	Long	Purchase Type (see Additional Data Definition).
uinType	Integer	UIN Type (see Additional Data Definition).
uinCaptureTime	Integer	UIN Capture Time (see Additional Data Definition).
uinLabelId	Long	The UIN label unique identifier.
uinExternalCreateAllowed	Boolean	True if an external system can create a UIN, false otherwise.
replenishmentMethod	String	The replenishment method: (SO) Store Orders (has meaning), otherwise meaningless text.
rejectStoreOrder	Boolean	True indicates store orders must be on or after the next delivery date or should be rejected.
multipleDeliveryPerDayAllowed	Boolean	True indicates the item allows multiple deliveries per day at the location.
nextDeliveryDate	Date	The next delivery date of the time based on its replenishment type.

Additional Data Definitions

Item Status Type

Value	Definition
1	Active
2	Discontinued
3	Inactive
4	Deleted
5	Auto Stockable
6	Non Ranged

Item Type

Value	Definition
1	Item
2	Simple Pack
3	Complex Pack
4	Simple Breakable Pack

5	Complex Breakable Pack

Source Type

Value	Definition
1	Supplier
2	Warehouse
3	Finisher

Item Purchase Type

Value	Definition
1	Consignment
2	Concession

Item UIN Type

Value	Definition
1	Serial
2	AGSN

Item UIN Capture Time Type

Value	Definition
1	Sale
2	Store Receiving

Item Wastage Type

Value	Definition
1	Sales Wastage
2	Spoilage

API: Find Items by Scan

This API searches for item information by an unknown identifier, most likely a scan. It first searches for the item using the searchScan as a potential item, and if found will return records based on that. It then searches as a UPC, barcode, or UIN, in that order, halting if it finds potential matches. An optional store identifier can be included as well.

API Basics

Endpoint URL	{base URL}/scan
Method	POST
Successful Response	200 OK
Processing Type	

Input	Criteria
Output	List of Items
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
searchScan	String(128)	Yes	A scan to find matching items.
storeId	Long	No	The store to retrieve item information for.

Output Data Definition

Table 4-9 Good Response (Code 200)

Attribute	Data Type	Description
itemId	String(25)	The item identifier
type	Integer(2)	The item type. Valid values are< br/> 1 - Item< br/ 2> - Simple Pack< br/ 3> - Complex Pack< br/ 4> - Simple Breakable Pack < br/ 5> - Complex Breakable Pack Enum: 1 2 3 4 5 example: 1
status	Integer(2)	The status of the item Valid values are< br/> 1 - Active < br/ 2> - Discontinued < br/ 3> - Inactive < br/ 4> - Deleted < br/ 5> - Auto Stockable Non-Ranged Enum: 1 2 3 4 5 example: 1
shortDescription	String(255)	The short description of the item
longDescription	String(400)	The long description of the item
departmentId	BigDecimal(12)	The identifier of the department the item belongs to
classId	BigDecimal(12)	The identifier of the class the item belongs to
subclassId	BigDecimal(12)	The identifier of the subclass the item belongs to
storeId	Integer(100)	The store identifier (if passed in as a query parameter)
ranged	Boolean	True if the item is ranged to the requested store, false otherwise.

Responses

Code	Description

200	Successful
	Example:
	
	`[`
	`  {`
	`  "itemId": "1111111",`
	`  "type": 1,`
	`  "status": 1,`
	`  "shortDescription": "Shoe",`
	`  "longDescription": "Steel-toed Shoe (two-tone)",`
	`  "departmentId": 1000,`
	`  "classId": 2000,`
	`  "subclassId": 3000,`
	`  "storeId": 5000,`
	`  "ranged": true`
	`  }`
	`]`

400	Bad Request
	Example:
	
	`[`
	`  {`
	`  "errors": [`
	`  {`
	`  "code": 1,`
	`  "description": "AN_ERROR_TOOK_PLACE",`
	`  "dataElement": "item",`
	`  "dataValue": "1234",`
	`  "referenceElement": "store",`
	`  "referenceValue": "123"`
	`  }`
	`  ]`
	`  }`
	`]`

API: Find Items by Inventory

Search for summary information about an item based on search criteria, primarily inventory information.

API Basics

Endpoint URL	{base URL}/inventory
Method	GET
Successful Response	200 Successful
Processing Type	
Input	Query Parameters

Output	Inventory information
Max Response Limit	

Query Parameter Definition

Attribute	Data Type	Description
available	boolean	Include only items with positive available inventory.
unavailable	boolean	Include only items with positive unavailable inventory.
nonsellableTypeId	integer(12)	Include only items with this non-sellable quantity type.
departmentId	integer(12)	Include only items associated to this merchandise hierarchy department.
classId	integer(12)	Include only items associated to this merchandise hierarchy class.
subclassId	integer(12)	Include only items associated to this merchandise hierarchy subclass.
storeId	integer(12)	Include only items ranged to this particular store.

Output Data Definition

Attribute	Data Type	Description
itemId	String(25)	The item identifier.
storeId	Integer(10)	The store identifier (if passed in a query parameter)
type	Integer(2)	The item type. Valid values are< br/> 1 - Item< br/ 2> - Simple Pack< br/ 3> - Complex Pack< br/ 4> - Simple Breakable Pack < br/ 5> - Complex Breakable Pack Enum: 1 2 3 4 5 example: 1
status	Integer(2)	The status of the item Valid values are< br/> 1 - Active < br/ 2> - Discontinued < br/ 3> - Inactive < br/ 4> - Deleted < br/ 5> - Auto Stockable Non-Ranged Enum: 1 2 3 4 5 example: 1
shortDescription	String(255)	A short description of the item.
longDescription	String(400)	A long description of the item.
departmentId	BigDecimal(12)	The identifier of the department the item belongs to.
classId	BigDecimal(12)	The identifier of the class the item belongs to.
subclassId	BigDecimal(12)	The identifier of the subclass the item belongs to.

Output Data Definition

Table 4-10 Good Response (Code 200)

Attribute	Data Type	Description
itemId	String(25)	The item identifier
storeId	String(25)	The store identifier (if passed in a query parameter)
type	Integer(2)	The item type. Valid values are< br/> 1 - Item< br/ 2> - Simple Pack< br/ 3> - Complex Pack< br/ 4> - Simple Breakable Pack < br/ 5> - Complex Breakable Pack Enum: 1 2 3 4 5 example: 1
status	Integer(2)	The status of the item Valid values are< br/> 1 - Active < br/ 2> - Discontinued < br/ 3> - Inactive < br/ 4> - Deleted < br/ 5> - Auto Stockable Non-Ranged Enum: 1 2 3 4 5 example: 1
shortDescription	String(255)	The short description of the item
longDescription	String(400)	The long description of the item
departmentId	BigDecimal(12)	The identifier of the department the item belongs to
classId	BigDecimal(12)	The identifier of the class the item belongs to
subclassId	BigDecimal(12)	The identifier of the subclass the item belongs to

Responses

Code	Description
200	Successful Example: `[ { "itemId": "1111111", "storeId": 5000, "type": 1, "status": 1, "shortDescription": "Shoe", "longDescription": "Steel-toed Shoe (two-tone)", "departmentId": 1000, "classId": 2000, "subclassId": 3000 } ]`

400	Bad Request
	Example:
	
	[
	{
	"errors": [
	{
	"code": 1,
	"description": "AN_ERROR_TOOK_PLACE",
	"dataElement": "item",
	"dataValue": "1234",
	"referenceElement": "store",
	"referenceValue": "123"
	}
	]
	}
	]

API: Find Related Items

Search for summary information about an item based on search criteria, primarily parent item.

API Basics

Endpoint URL	{base URL}/related
Method	GET
Successful Response	200 Successful
Processing Type	
Input	Query Parameters
Output	Item information
Max Response Limit	

Query Parameter Definition

Attribute	Data Type	Description
parentItemId	string(25)	Include only items related to this parent item identifier.
storeId	integer(10)	Include only items ranged to this particular store.

Output Data Definition

Attribute	Data Type	Description
itemId	String(25)	The item identifier.
storeId	Integer(10)	The store identifier (if passed in a query parameter)

type	Integer(2)	The item type. Valid values are< br/> 1 - Item< br/ 2> - Simple Pack< br/ 3> - Complex Pack< br/ 4> - Simple Breakable Pack < br/ 5> - Complex Breakable Pack Enum: 1 2 3 4 5 example: 1
status	Integer(2)	The status of the item Valid values are< br/> 1 - Active < br/ 2> - Discontinued < br/ 3> - Inactive < br/ 4> - Deleted < br/ 5> - Auto Stockable Non-Ranged Enum: 1 2 3 4 5 example: 1
shortDescription	String(255)	A short description of the item.
longDescription	String(400)	A long description of the item.
departmentId	BigDecimal(12)	The identifier of the department the item belongs to.
classId	BigDecimal(12)	The identifier of the class the item belongs to.
subclassId	BigDecimal(12)	The identifier of the subclass the item belongs to.

Output Data Definition

Table 4-11 Good Response (Code 200)

Attribute	Data Type	Description
itemId	String(25)	The item identifier
storeId	String(25)	The store identifier (if passed in a query parameter)
type	Integer(2)	The item type. Valid values are< br/> 1 - Item< br/ 2> - Simple Pack< br/ 3> - Complex Pack< br/ 4> - Simple Breakable Pack < br/ 5> - Complex Breakable Pack Enum: 1 2 3 4 5 example: 1
status	Integer(2)	The status of the item Valid values are< br/> 1 - Active < br/ 2> - Discontinued < br/ 3> - Inactive < br/ 4> - Deleted < br/ 5> - Auto Stockable Non-Ranged Enum: 1 2 3 4 5 example: 1
shortDescription	String(255)	The short description of the item
longDescription	String(400)	The long description of the item
departmentId	BigDecimal(12)	The identifier of the department the item belongs to
classId	BigDecimal(12)	The identifier of the class the item belongs to
subclassId	BigDecimal(12)	The identifier of the subclass the item belongs to

Responses

Code	Description

200	Successful
	Example:
	
	[
	  {
	  "itemId": "1111111",
	  "storeId": 5000,
	  "type": 1,
	  "status": 1,
	  "shortDescription": "Shoe",
	  "longDescription": "Steel-toed Shoe (two-tone)",
	  "departmentId": 1000,
	  "classId": 2000,
	  "subclassId": 3000
	  }
	]

400	Bad Request
	Example:
	
	[
	  {
	  "errors": [
	  {
	  "code": 1,
	  "description": "AN_ERROR_TOOK_PLACE",
	  "dataElement": "item",
	  "dataValue": "1234",
	  "referenceElement": "store",
	  "referenceValue": "123"
	  }
	  ]
	  }
	]

API:Find Items by UDA

Search for summary information about an item based on search criteria, primarily user defined attributes.

API Basics

Endpoint URL	{base URL}/uda
Method	GET
Successful Response	200 Successful
Processing Type	
Input	Query Parameters
Output	Inventory information

Max Response Limit

Query Parameter Definition

Attribute	Data Type	Description
udaId	integer(5)	Include only items with this user defined attribute identifier.
udaText	string(250)	Include only items with this UDA text value
udaLovId	string(25)	Include only items with a UDA value matching this list of values identifier.
udaDateFrom	date-time	Include only items with a date UDA value on or after this date.
udaDateTo	date-time	Include only items with a date UDA value on or before this date.
storeId	integer(12)	Include only items ranged to this particular store.

Output Data Definition

Attribute	Data Type	Description
itemId	String(25)	The item identifier.
storeId	Integer(10)	The store identifier (if passed in a query parameter)
type	Integer(2)	The item type. Valid values are< br/> 1 - Item< br/ 2> - Simple Pack< br/ 3> - Complex Pack< br/ 4> - Simple Breakable Pack < br/ 5> - Complex Breakable Pack Enum: 1 2 3 4 5 example: 1
status	Integer(2)	The status of the item Valid values are< br/> 1 - Active < br/ 2> - Discontinued < br/ 3> - Inactive < br/ 4> - Deleted < br/ 5> - Auto Stockable Non-Ranged Enum: 1 2 3 4 5 example: 1
shortDescription	String(255)	A short description of the item.
longDescription	String(400)	A long description of the item.
departmentId	BigDecimal(12)	The identifier of the department the item belongs to.
classId	BigDecimal(12)	The identifier of the class the item belongs to.
subclassId	BigDecimal(12)	The identifier of the subclass the item belongs to.

Output Data Definition

Table 4-12 Good Response (Code 200)

Attribute	Data Type	Description
itemId	String(25)	The item identifier
storeId	String(25)	The store identifier (if passed in a query parameter)
type	Integer(2)	The item type. Valid values are< br/> 1 - Item< br/ 2> - Simple Pack< br/ 3> - Complex Pack< br/ 4> - Simple Breakable Pack < br/ 5> - Complex Breakable Pack Enum: 1 2 3 4 5 example: 1
status	Integer(2)	The status of the item Valid values are< br/> 1 - Active < br/ 2> - Discontinued < br/ 3> - Inactive < br/ 4> - Deleted < br/ 5> - Auto Stockable Non-Ranged Enum: 1 2 3 4 5 example: 1
shortDescription	String(255)	The short description of the item
longDescription	String(400)	The long description of the item
departmentId	BigDecimal(12)	The identifier of the department the item belongs to
classId	BigDecimal(12)	The identifier of the class the item belongs to
subclassId	BigDecimal(12)	The identifier of the subclass the item belongs to

Responses

Code	Description

200	Successful
	Example:
	
	[
	{
	"itemId": "1111111",
	"storeId": 5000,
	"type": 1,
	"status": 1,
	"shortDescription": "Shoe",
	"longDescription": "Steel-toed Shoe (two-tone)",
	"departmentId": 1000,
	"classId": 2000,
	"subclassId": 3000
	}
	]

400 Bad Request

Example:


```
[
  {
    "errors": [
      {
        "code": 1,
        "description": "AN_ERROR_TOOK_PLACE",
        "dataElement": "item",
        "dataValue": "1234",
        "referenceElement": "store",
        "referenceValue": "123"
      }
    ]
  }
]
```


REST Service: Item Inventory

This service retrieves information about item inventory.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/inventory

API Definitions

API	Description
Find Available Inventory	Search for available inventory information by multiple items and multiple locations.
Find Inventory	Searches for standard inventory information by multiple items and multiple locations.
Find Expanded Inventory	Searches for expanded inventory information by multiple items at a single store.
Find Future Inventory	Searches for future inventory delivery information by a single item and a single store.
Find Inventory In Buddy Stores	Searches for inventory information at buddy stores by single input store and multiple items.
Find Inventory In Transfer Stores	Searches for inventory information at transfer zone stores by single input store and multiple items.

API: Find Available Inventory

Searches for available inventory quantity about an item in requested locations. Only transaction-level items are processed, and only current available inventory is returned. The multiplied combination of items and locations within the input criteria cannot exceed 10,000.

Invalid items or locations will not cause this API to fail. Inventory is returned for any item and locations found and is not returned invalid or not found items or locations.

API Basics

Endpoint URL	{base URL}/available
Method	POST
Successful Response	200 OK
Processing Type	Synchronous
Input	Criteria
Output	List of items
Max Response Limit	10,000

Input Data Definition

Attribute	Data Type	Required	Description
itemIds	List of Strings	Yes	A list of items to retrieve available inventory for.
locationIds	List of Longs	Yes	A list of location identifiers to retrieve available inventory for.
locationType	Integer	Yes	A location type: See Location Type

Example Input

```
{
"itemIds": [
"100637156",
"100637172",
"100653105"
],
"locationIds": [
5000,
5001,
5005
],
"locationType": 1
}
```

Output Data Definition

Attribute	Data Type	Required	Description
itemId	String	Yes	The item identifier.
locationId	Long	Yes	The location identifier.

Attribute	Data Type	Required	Description
locationType	Integer	Yes	The location type: See Location Type.
availableQuantity	BigDecimal	Yes	The amount of available inventory.
unitOfMeasure	String	Yes	The unit of measure of the available inventory.
estimatedPack	Boolean	Yes	True if this is an estimated pack quantity, false otherwise.

Example Output

[

{

"itemId": "100637113",

"locationId": 5000,

"locationType": 1,

"availableQuantity": 200.0000,

"unitOfMeasure": "EA",

"estimatedPack": false

},

{

"itemId": "100637113",

"locationId": 5001,

"locationType": 1,

"availableQuantity": 200.0000,

"unitOfMeasure": "EA",

"estimatedPack": false

},

}

Additional Data Definitions

Location Type

Value	Definition
1	Store
2	Warehouse

API: Find Inventory

Query lookup of detailed inventory information about a multiple item in multiple stores. The multiplied combination of items and locations within the input criteria cannot exceed 10,000. Invalid items or locations will not cause this API to fail. Inventory is returned for any item and locations found and is not returned invalid or not found items or locations.

API Basics

Endpoint URL	{base URL}/positions
Method	POST
Successful Response	200 OK
Processing Type	Synchronous
Input	Criteria
Output	List of inventory of item at stores
Max Response Limit	10,000

Input Data Definition

Attribute	Data Type	Required	Description
itemIds	List of Strings	Yes	A list of items to retrieve inventory for.
storeIds	List of Longs	Yes	A list of store identifiers to retrieve inventory for.
sellingUnitOfMeasure	Boolean	-	True indicates an attempt to use the selling unit of measure of the item, false indicates to use the standard unit of measure. If conversion cannot take place, it defaults back to standard unit of measure.

Example Input

{
"itemIds": [
"100637156",
"100637172",
"100668091"
],
"storeIds": [
5000,
5001,
5002
],
"sellingUnitOfMeasure": true

}

Output Data Definition

Attribute	Data Type	Required	Description
itemId	String	Yes	The item identifier.
storeId	Long		The store identifier if the item is ranged to a store.
ranged	Boolean	Yes	True if the item is ranged to the store, false otherwise.
estimated	Boolean	Yes	True if the quantities are estimated, false otherwise.
unitOfMeasure	String	Yes	The unit of measure of the quantities.
caseSize	BigDecimal	Yes	The default case size of the item.
quantityStockOnHand	BigDecimal	Yes	The stock on hand quantity.
quantityBackroom	BigDecimal	Yes	The quantity located in the back room area.
quantityShopfloor	BigDecimal	Yes	The quantity located on the shop floor.
quantityDeliveryBay	BigDecimal	Yes	The quantity located in the delivery bay.
quantityAvailable	BigDecimal	Yes	The available to sell quantity.
quantityUnavailable	BigDecimal	Yes	The unavailable to sell quantity.
quantityNonSellable	BigDecimal	Yes	The total non-sellable quantity.
quantityInTransit	BigDecimal	Yes	The quantity currently in transit.
quantityCustomerReserved	BigDecimal	Yes	The quantity reserved for customer orders.
quantityTransferReserved	BigDecimal	Yes	The quantity reserved for transfers.
quantityVendorReturn	BigDecimal	Yes	The quantity reserved for vendor returns.
nonSellableQuantities	List of Non-Sellable Quantities	-	A collection containing the specific quantity in each non-sellable quantity type bucket.

Non-Sellable Quantity Data Definition

Attribute	Data Type	Required	Description
nonsellableTypeId	Long	Yes	The non-sellable type unique identifier.
quantity	quantity	Yes	The quantity in this particular non-sellable type bucket.

Example Output

[

{

"itemId": "100637156",

"storeId": 5000,

"ranged": true,

"estimated": false,

"unitOfMeasure": "EA",
"caseSize": 100.00,
"quantityStockOnHand": 10.0000,
"quantityBackroom": 10.0000,
"quantityShopfloor": 0.0000,
"quantityDeliveryBay": 0.0000,
"quantityAvailable": 10.0000,
"quantityUnavailable": 0.0000,
"quantityNonSellable": 0.0000,
"quantityInTransit": 0.0000,
"quantityCustomerReserved": 0.0000,
"quantityTransferReserved": 0.0000,
"quantityVendorReturn": 0.0000
},
{
"itemId": "100637172",
"storeId": 5000,
"ranged": true,
"estimated": false,
"unitOfMeasure": "EA",
"caseSize": 100.00,
"quantityStockOnHand": 10.0000,
"quantityBackroom": -10.0000,
"quantityShopfloor": 0.0000,
"quantityDeliveryBay": 0.0000,
"quantityAvailable": -10.0000,
"quantityUnavailable": 20.0000,
"quantityNonSellable": 20.0000,
"quantityInTransit": 0.0000,
"quantityCustomerReserved": 0.0000,
"quantityTransferReserved": 0.0000,
"quantityVendorReturn": 0.0000,
"nonSellableIdos": [
{

"nonsellableTypeId": 1,
"quantity": 15.0000
},
{
"nonsellableTypeId": 2,
"quantity": 5.0000
}
]
}
}

API: Find Expanded Inventory

Searches for expanded inventory information about multiple items within a single store.

API Basics

Endpoint URL	{base URL}/{storeId}/expanded
Method	POST
Successful Response	200 OK
Processing Type	Synchronous
Input	Criteria
Output	List of inventory of items
Max Response Limit	2,500

Path Parameter Definitions

Attribute	Description
storeId	The store identifier of the store to process items for.

Input Data Definition

Attribute	Data Type	Required	Description
itemIds	List of Strings	Yes	A list of items to retrieve expanded inventory for.

Example Input

{
"itemIds": [
"100637156",
"100637172",
"100695081"

]

}

Output Data Definition

Attribute	Data Type	Required	Description
itemId	String	Yes	The item identifier.
storeId	Long		The store identifier if the item is ranged to a store.
ranged	Boolean	Yes	True if the item is ranged to the store, false otherwise.
estimated	Boolean	Yes	True if the quantities are estimated, false otherwise.
unitOfMeasure	String	Yes	The unit of measure of the quantities.
caseSize	BigDecimal	Yes	The default case size of the item.
quantityStockOnHand	BigDecimal	Yes	The stock on hand quantity.
quantityBackroom	BigDecimal	Yes	The quantity located in the back room area.
quantityShopfloor	BigDecimal	Yes	The quantity located on the shop floor.
quantityDeliveryBay	BigDecimal	Yes	The quantity located in the delivery bay.
quantityAvailable	BigDecimal	Yes	The available to sell quantity.
quantityUnavailable	BigDecimal	Yes	The unavailable to sell quantity.
quantityNonSellable	BigDecimal	Yes	The total non-sellable quantity.
quantityInTransit	BigDecimal	Yes	The quantity currently in transit.
quantityCustomerReserved	BigDecimal	Yes	The quantity reserved for customer orders.
quantityTransferReserved	BigDecimal	Yes	The quantity reserved for transfers.
quantityVendorReturn	BigDecimal	Yes	The quantity reserved for vendor returns.
firstReceivedDate	Date	-	The first date the item was received into stock.
lastReceivedDate	Date	-	The date the item last received inventory into stock.
lastReceivedQuantity	BigDecimal	-	Total amount of inventory received on the last date it was received.
openStockCounts	Integer	-	The number of stock counts open for the item at this store.
lastStockCountType	Integer	-	The type of stock count (see Additional Data Definition).
lastStockCountApprovedDate	Date	-	The date this item was last approved on a stock count at this store.
lastStockCountTimeframe	Integer	-	The stock count timeframe (see Additional Data Definition).
uinProblemLine	Boolean	Yes	True indicates it is UIN problem line item, false otherwise.
lastRequestedQuantity	BigDecimal	-	The quantity last requested for this item.
lastUpdateDate	Date	Yes	The timestamp of the last time this record was updated.

Attribute	Data Type	Required	Description
nonSellableQuantities	Collection of Non-Sellable Quantities	-	The specific quantities in each non-sellable quantity type bucket.

Non-Sellable Quantity Data Definition

Attribute	Data Type	Required	Description
nonsellableTypeId	Long	Yes	The non-sellable type unique identifier.
quantity	quantity	Yes	The quantity in this particular non-sellable type bucket.

Example Output

[

{

"itemId": "100637113",

"storeId": 5000,

"ranged": true,

"estimated": false,

"unitOfMeasure": "EA",

"caseSize": 100.00,

"quantityStockOnHand": 200.0000,

"quantityBackroom": 200.0000,

"quantityShopfloor": 0.0000,

"quantityDeliveryBay": 0.0000,

"quantityAvailable": 200.0000,

"quantityUnavailable": 0.0000,

"quantityNonSellable": 0.0000,

"quantityInTransit": 0.0000,

"quantityCustomerReserved": 0.0000,

"quantityTransferReserved": 0.0000,

"quantityVendorReturn": 0.0000,

"quantityLastReceived": 0.0000,

"quantityLastRequested": 0.0000,

"openStockCounts": 0,

"lastStockCountTimeframe": 3,

"uinProblemLine": false,

"lastUpdateDate": "2022-07-15T06:23:27-05:00"
},
{
"itemId": "100637121",
"storeId": 5000,
"ranged": true,
"estimated": false,
"unitOfMeasure": "EA",
"caseSize": 100.00,
"quantityStockOnHand": 200.0000,
"quantityBackroom": 180.0000,
"quantityShopfloor": 0.0000,
"quantityDeliveryBay": 0.0000,
"quantityAvailable": 180.0000,
"quantityUnavailable": 20.0000,
"quantityNonSellable": 20.0000,
"quantityInTransit": 0.0000,
"quantityCustomerReserved": 0.0000,
"quantityTransferReserved": 0.0000,
"quantityVendorReturn": 0.0000,
"quantityLastReceived": 0.0000,
"quantityLastRequested": 0.0000,
"openStockCounts": 0,
"lastStockCountTimeframe": 3,
"uinProblemLine": false,
"lastUpdateDate": "2022-07-15T06:23:27-05:00",
"nonSellableIdos": [
{
"nonsellableTypeId": 1,
"quantity": 15.0000
},
{
"nonsellableTypeId": 2,
"quantity": 5.0000

```
}
]
}
}
```

API: Find Future Inventory

Searches for future delivery records for a single store and single item.

API Basics

Endpoint URL	{base URL}/{storeId}/{itemId}/future
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	None
Output	List of delivery records
Max Response Limit	N/A

Path Parameter Definitions

Attribute	Description
storeId	The store identifier to retrieve future inventory for.
itemId	The item identifier to retrieve future inventory for.

Output Data Definition

Attribute	Data Type	Required	Description
itemId	String (25)	Yes	The item identifier.
storeId	Long	Yes	The store identifier.
deliveries	List of deliveryIds	-	A list of delivery information if it exists.

Delivery Data Definition

Attribute	Data Type	Required	Description
sourceLocationType	Integer	Yes	Item Location Type (see Additional Data Definition).
sourceLocationId	Long	Yes	The unique identifier of the source location of the delivery.
deliveryType	Integer	Yes	Item Delivery Type (see Additional Data Definition).
expectedDate	Date	Yes	The date the inventory is expected to arrive.
quantityInbound	BigDecimal	Yes	Amount of inventory inbound on the delivery.
quantityOrdered	BigDecimal	Yes	Amount of inventory on order.

Example Output

{
"itemId": "100637121",
"storeId": 5000,
"deliveryIdos": [
{
"sourceLocationType": 1,
"sourceLocationId": 5001,
"deliveryType": 3,
"quantityInbound": 30.0000,
"quantityOrdered": 0.0000
}
]
}

Additional Data Definitions

Item Delivery Type

Value	Definition
1	Allocation
2	Purchase Order
3	Transfer
-	-

Item Location Type

Value	Definition
1	Store
2	Supplier
3	Warehouse
4	Finisher

API: Find Inventory in Buddy Stores

Searches for inventory information at buddy stores by single input store and multiple items.

API Basics

Endpoint URL	{baseUrl}/{storeId}/associated
Method	POST

Successful Response	200 OK
Processing Type	Synchronous
Input	List of items
Output	List of inventory records
Max Response Limit	N/A

Path Parameter Definitions

Attribute	Description
storeId	The store identifier to find buddy stores for.

Input Data Definition

Attribute	Data Type	Required	Description
itemIds	List of Strings	Yes	A list of items to retrieve inventory for.
sellingUnitOfMeasure	Boolean	-	True indicates an attempt to use the selling unit of measure of the item, false indicates to use the standard unit of measure. If conversion cannot take place, it defaults back to standard unit of measure.

Example Input

{

"itemIds": [

"100637156",

"100637172",

"100668091"

],

"sellingUnitOfMeasure": true

}

Output Data Definition

Attribute	Data Type	Required	Description
itemId	String	Yes	The item identifier.
storeId	Long		The store identifier if the item is ranged to a store.
ranged	Boolean	Yes	True if the item is ranged to the store, false otherwise.
estimated	Boolean	Yes	True if the quantities are estimated, false otherwise.
unitOfMeasure	String	Yes	The unit of measure of the quantities.

Attribute	Data Type	Required	Description
caseSize	BigDecimal	Yes	The default case size of the item.
quantityStockOnHand	BigDecimal	Yes	The stock on hand quantity.
quantityBackroom	BigDecimal	Yes	The quantity located in the back room area.
quantityShopfloor	BigDecimal	Yes	The quantity located on the shop floor.
quantityDeliveryBay	BigDecimal	Yes	The quantity located in the delivery bay.
quantityAvailable	BigDecimal	Yes	The available to sell quantity.
quantityUnavailable	BigDecimal	Yes	The unavailable to sell quantity.
quantityNonSellable	BigDecimal	Yes	The total non-sellable quantity.
quantityInTransit	BigDecimal	Yes	The quantity currently in transit.
quantityCustomerReserved	BigDecimal	Yes	The quantity reserved for customer orders.
quantityTransferReserved	BigDecimal	Yes	The quantity reserved for transfers.
quantityVendorReturn	BigDecimal	Yes	The quantity reserved for vendor returns.
nonSellableQuantities	List of Non-Sellable Quantities	-	A collection containing the specific quantity in each non-sellable quantity type bucket.

Non-Sellable Quantity Data Definition

Attribute	Data Type	Required	Description
nonsellableTypeId	Long	Yes	The non-sellable type unique identifier.
quantity	quantity	Yes	The quantity in this particular non-sellable type bucket.

Example Output

[

{

"itemId": "100637156",

"storeId": 5000,

"ranged": true,

"estimated": false,

"unitOfMeasure": "EA",

"caseSize": 100.00,

"quantityStockOnHand": 10.0000,

"quantityBackroom": 10.0000,

"quantityShopfloor": 0.0000,

"quantityDeliveryBay": 0.0000,

"quantityAvailable": 10.0000,

"quantityUnavailable": 0.0000,
"quantityNonSellable": 0.0000,
"quantityInTransit": 0.0000,
"quantityCustomerReserved": 0.0000,
"quantityTransferReserved": 0.0000,
"quantityVendorReturn": 0.0000
},
{
"itemId": "100637172",
"storeId": 5000,
"ranged": true,
"estimated": false,
"unitOfMeasure": "EA",
"caseSize": 100.00,
"quantityStockOnHand": 10.0000,
"quantityBackroom": -10.0000,
"quantityShopfloor": 0.0000,
"quantityDeliveryBay": 0.0000,
"quantityAvailable": -10.0000,
"quantityUnavailable": 20.0000,
"quantityNonSellable": 20.0000,
"quantityInTransit": 0.0000,
"quantityCustomerReserved": 0.0000,
"quantityTransferReserved": 0.0000,
"quantityVendorReturn": 0.0000,
"nonSellableIdos": [
{
"nonsellableTypeId": 1,
"quantity": 15.0000
},
{
"nonsellableTypeId": 2,
"quantity": 5.0000
}

]

}

}

API: Find Inventory in Transfer Zone Stores

Searches for inventory at transfer zone stores by single input store and multiple items.

API Basics

Endpoint URL	{baseUrl}/{storeId}/transferzone
Method	POST
Successful Response	200 OK
Processing Type	Synchronous
Input	List of items
Output	List of inventory records
Max Response Limit	N/A

Path Parameter Definitions

Attribute	Description
storeId	The store identifier to find transfer zone stores for.

Input Data Definition

Attribute	Data Type	Required	Description
itemIds	List of Strings	Yes	A list of items to retrieve inventory for.
sellingUnitOfMeasure	Boolean	-	True indicates an attempt to use the selling unit of measure of the item, false indicates to use the standard unit of measure. If conversion cannot take place, it defaults back to standard unit of measure.

Example Input

{

"itemIds": [

"100637156",

"100637172",

"100668091"

],

"sellingUnitOfMeasure": true

}

Output Data Definition

Attribute	Data Type	Required	Description
itemId	String	Yes	The item identifier.
storeId	Long		The store identifier if the item is ranged to a store.
ranged	Boolean	Yes	True if the item is ranged to the store, false otherwise.
estimated	Boolean	Yes	True if the quantities are estimated, false otherwise.
unitOfMeasure	String	Yes	The unit of measure of the quantities.
caseSize	BigDecimal	Yes	The default case size of the item.
quantityStockOnHand	BigDecimal	Yes	The stock on hand quantity.
quantityBackroom	BigDecimal	Yes	The quantity located in the back room area.
quantityShopfloor	BigDecimal	Yes	The quantity located on the shop floor.
quantityDeliveryBay	BigDecimal	Yes	The quantity located in the delivery bay.
quantityAvailable	BigDecimal	Yes	The available to sell quantity.
quantityUnavailable	BigDecimal	Yes	The unavailable to sell quantity.
quantityNonSellable	BigDecimal	Yes	The total non-sellable quantity.
quantityInTransit	BigDecimal	Yes	The quantity currently in transit.
quantityCustomerReserved	BigDecimal	Yes	The quantity reserved for customer orders.
quantityTransferReserved	BigDecimal	Yes	The quantity reserved for transfers.
quantityVendorReturn	BigDecimal	Yes	The quantity reserved for vendor returns.
nonSellableQuantities	List of Non-Sellable Quantities	-	A collection containing the specific quantity in each non-sellable quantity type bucket.

Non-Sellable Quantity Data Definition

Attribute	Data Type	Required	Description
nonsellableTypeId	Long	Yes	The non-sellable type unique identifier.
quantity	quantity	Yes	The quantity in this particular non-sellable type bucket.

Example Output

[

{

"itemId": "100637156",

"storeId": 5000,

"ranged": true,

"estimated": false,
"unitOfMeasure": "EA",
"caseSize": 100.00,
"quantityStockOnHand": 10.0000,
"quantityBackroom": 10.0000,
"quantityShopfloor": 0.0000,
"quantityDeliveryBay": 0.0000,
"quantityAvailable": 10.0000,
"quantityUnavailable": 0.0000,
"quantityNonSellable": 0.0000,
"quantityInTransit": 0.0000,
"quantityCustomerReserved": 0.0000,
"quantityTransferReserved": 0.0000,
"quantityVendorReturn": 0.0000
},
{
"itemId": "100637172",
"storeId": 5000,
"ranged": true,
"estimated": false,
"unitOfMeasure": "EA",
"caseSize": 100.00,
"quantityStockOnHand": 10.0000,
"quantityBackroom": -10.0000,
"quantityShopfloor": 0.0000,
"quantityDeliveryBay": 0.0000,
"quantityAvailable": -10.0000,
"quantityUnavailable": 20.0000,
"quantityNonSellable": 20.0000,
"quantityInTransit": 0.0000,
"quantityCustomerReserved": 0.0000,
"quantityTransferReserved": 0.0000,
"quantityVendorReturn": 0.0000,
"nonSellableIdos": [

{
"nonsellableTypeId": 1,
"quantity": 15.0000
},
{
"nonsellableTypeId": 2,
"quantity": 5.0000
}
]
}
}

REST Service: Item ISN

This rest service defines operations to manage Item ISN information.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env/siocs-int-services>/api/isns

APIs

API	Description
Create ISN	Create a new ISN
Update ISN	Updates an existing ISN
Delete ISN	Delete an existing ISN
Find ISNs	Search for ISNs based on a set of criteria
Read ISN Types	Read all Item ISN types

API: Create ISN

This API will create ISN information.

API Basics

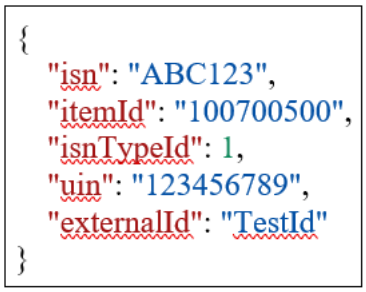
Endpoint URL	{base URL}
Method	POST
Successful Response	200 OK
Processing Type	Synchronous

Input	The ISN
Output	The ISN including ID
Max Response Limit	N/A

Input Data Definition

Attribute	Type	Req	Definition
isn	String(128)	X	A scan number used to scan the item
isnTypeId	Long(12)	X	The unique identifier of the item ISN type
itemId	String(25)	X	The unique item identifier (sku number)
uin	String(128)		A universal identification number
externalId	String(128)		An identifier from an external system

Example



Output Data Definition

Attribute	Type	Definition
isnId	Long(12)	The unique identifier of the ISN record
isn	String(128)	This is a scan number used to scan the item
externalId	String(128)	An identifier from an external system

API: Update ISN

This API allows the ISN information to be modified. An optional value left blank will update the ISN information to blank for that value.

API Basics

Endpoint URL	{base URL}/isnId
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	ISN Update Information

Output	N/A
Max Response Limit	N/A

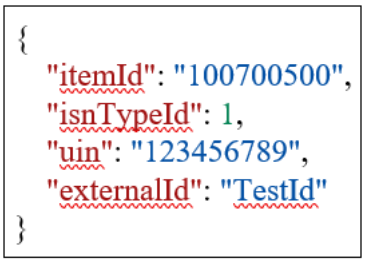
Path Data Definiton

Attribute	Type	Definition
isnId	Long(12)	The unique identifier of the item ISN.

Input Data Definition

Attribute	Type	Req	Definition
isnTypeId	Long(12)	X	The unique identifier of the item ISN type
itemId	String(25)	X	The unique item identifier (sku number)
uin	String(128)		A universal identification number
externalId	String(128)		An identifier from an external system

Example



API: Delete ISN

This API deletes an item ISN.

API Basics

Endpoint URL	{base URL}/{isnId}/delete
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	N/A
Output	N/A
Max Response Limit	N/A

Path Data Definition

Attribute	Type	Definition

isinId	Long(12)	The unique identifier of the item ISN.

API: Find ISNs

This API is used to lookup or find ISNs.

API Basics

Endpoint URL	{base URL}
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	N/A
Output	List of ISNs
Max Response Limit	10,000

Query Parameters

Attribute	Type	Definition
isn	String(128)	Retrieves records containing this ISN, which could be for multiple items.
ItemId	String(25)	Retrieves records associated to this unique item identifier (sku number).
uin	String(128)	Retrieves records associated to this universal identification number.
updateDateFrom	String	Retrieves records on or after this date.
updateDateTo	String	Retrieves records on or before this date.

Output Data Definition

Attribute	Type	Definition
itemIsnId	Long(12)	The unique identifier of the record
isn	String(128)	Retrieves records containing this ISN, which could be for multiple items.
ItemId	String(25)	Retrieves records associated to this unique item identifier (sku number).
uin	String(128)	Retrieves records associated to this universal identification number.
itemIsnTypeId	Long(12)	The unique identifier of the item ISN type
externalId	String(128)	An identifier of this ISN from an external system
createDate	Date	The date this ISN record was first created
createUser	String(128)	The user that created this ISN record
updateDate	Date	The last date this ISN record was updated
updateUser	String(128)	The user that last updated this ISN record.

API: Read ISN Types

This API is used to lookup all the ISN types available for an ISN.

API Basics

Endpoint URL	{base URL}/types
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	N/A
Output	List of ISN Types
Max Response Limit	N/A

Output Data Definition

Attribute	Type	Definition
isnTypeId	Long(12)	The unique ISN type identifier
labelKey	String(128)	A label for the ISN type (not translated)
restricted	Boolean	Y if this represents secure data, N otherwise.

REST Service: Item Price

This service allows for the search and retrieval of item pricing information stored within EICS.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer></cust_env>/siocs-int-services/api/prices

API Definitions

API Definitions

API	Description
findPrices	This API can be used to search for price summary information matching filter criteria.
FindPriceByIds	Find extended price information based on a list of potential unique price identifiers.

API: findPrices

This API can be used to search for price summary information matching filter criteria.

At least one input criteria is required or a bad request error will be returned.

API Basics

Endpoint URL	{base URL}
Method	GET
Success Response	200 OK
Processing Type	Synchronous
Input	Query Parameters
Output	List of Prices
Maximum Results Allowed	10,000

Input Data Definition

Attribute	Data Type	Definition
storeId	Long(10)	Only retrieve item price information for this store.
effectiveDateFrom	String	Only retrieve item price information on or after this date.
effectiveDateTo	String	Only retrieve item price information on or before this date.
itemId	String(25)	Only retrieve item price information for this item.

Output Data Definition

Attribute	Data Type	Definition
priceId	Long(12)	The unique identifier of the price.
itemId	String(25)	The unique identifier of the item.
storeId	Long(10)	The unique identifier of the store.
status	Integer	The status of the price.
priceType	Integer	The type of the price.
effectiveDate	Date	The effective date of the price.
endDate	Date	The end date of the price.
priceValue	BigDecimal(20,4)	The price amount.
priceCurrency	String(3)	The price currency.
unitOfMeasure	String(4)	The item unit of measure associated with the price.

API: findPriceByIds

Find extended price information based on a list of potential unique price identifiers.

It will return information only for price identifiers that are found. It will not return errors or fail to process if invalid identifiers occur. It is up to the accessing information to determined prices not found using this API.

API Basics

Endpoint URL	{base URL}/find
Method	POST
Success Response	200 OK
Processing Type	Synchronous
Input	ID List

Output	List of Prices
Maximum Input Allowed	5,000

Input Data Definition

Attribute	Type	Definition
priceIds	List<Long(12)>	A list of price identifiers.

Output Data Definition

Attribute	Type	Definition
priceId	Long(12)	The unique identifier of the price.
itemId	String(25)	The unique identifier of the item.
storeId	Long(10)	The unique identifier of the store.
Status	Integer	The status of the price.
priceType	Integer	The type of the price
effectiveDate	Date	The effective date of the price.
endDate	Date	The end date of the price.
priceValue	BigDecimal(20, 4)	The price amount.
priceCurrency	String(3)	The price currency.
unitOfMeasure	String(4)	The item unit of measure associated with the price.
externalPriceId	Long(12)	The unique identifier of the external price or price event.
clearanceId	Long(15)	The unique identifier of the clearance price change from the pricing engine.
promotionId	Long(10)	The unique identifier of the promotion.
regularPriceChangeId	Long(15)	The unique identifier of the regular price change from the pricing engine.
resetClearanceId	Long(15)	The identifier of the clearance reset.
storeRequested	Boolean	True indicates it is store requested, false indicates it is not store requested.
sellingUnitPriceChange	Boolean	True indicates the selling unit retail price has changed, false indicates it has not.
multiUnitPriceChange	Boolean	True indicates the multi-unit pricing has changed, false indicates it has not.
multiUnitRetail	BigDecimal(20, 4)	The multi-unit retail price.
multiUnitRetailCurrency	String(3)	The currency type of the multi-unit retail price in the multi-selling unit of measure.
multiUnits	BigDecimal(12, 4)	The number of multi-units.
multiUnitUom	String(4)	The unit of measure of the multi-unit retail price.
promotionName	String(160	The promotion name.
promotionCompDtlId	Long(15)	The unique identifier of the promotion component detail from the pricing engine.
promotionType	Integer	Promotion Component Type (See Index)
promotionDurationType	Integer	Promotion Duration Type (See Index)

promotionCompId	Long(10)	The unique identifier of the promotion component.
promotionCompName	String(160)	The promotion component name.
promotionDescription	String(640)	The promotion description.
updateDate	Date	The date that the update took place.

Example Input:

```
{
"priceIds": [
123,
 456,
 789,
 012
  ]
}
```

Additional Data Definitions

Price Type

ID	Status
1	Permanent
2	Promotional
3	Clearance
4	Clearance Reset

Price Status

ID	Status
1	New
2	Pending
3	Approved
4	Completed
5	Rejected
6	Ticket List
7	Extract Failed
8	Deleted
99	Default

Promotion Component Type

ID	Status
1	Complex Promotion
2	Simple Promotion
3	Threshold Promotion
4	Credit (Finance) Promotion
5	Transaction Promotion

Promotion Duration Type

ID	Status
1	All Day Promotion
2	Partial Day Promotion
3	Multiple Day Promotion

REST Service: Item UDA

This service integrates user defined attribute foundation data. Asynchronous item UDA integration is processed through staged messages and is controlled by the MPS Work Types.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/udas

API Definitions

API	Description
importUdas	Imports user defined attributes.
deleteUdas	Deletes user defined attributes.
importItemUdas	Imports an association between items and user defined attributes.
deleteItemUdas	Deletes the association between items and user defined attributes.
readItemUdas	Retrieves all the user defined attributes for a particular item.

API: Import Udas

Imports user defined attributes. It is managed by DcsUda work type. If the input exceeds 500 UDAs an input too large exception will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/import
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	UDA import List
Output	None
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
udas	List of details	Yes	A list of user defined attributes.

Detail Data Definition

Attribute	Data Type	Required	Description
udaId	Integer (5)	Yes	The user defined attribute identifier.
type	Integer	Yes	See Index: UdaType
description	String (120)	Yes	The description of the user defined attribute.
printTicket	Boolean		True indicates tickets are printed for this user defined attribute.
printLabel	Boolean		True indicates labels are printed for this user defined attribute.
lovId	String (25)		The unique identifier of a list of values UDA.
lovDescription	String (250)		The description of the list of values UDA.

API: Delete Udas

Deletes user defined attributes. It is managed by DcsUda work type. If the input exceeds 500 UDAs an input too large exception will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/delete
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	UDA delete list
Output	None
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
udaIds	List<Long>	Yes	A list of user defined attribute.

API: Import Item Udas

Imports associations between items and user defined attributes. This is controlled by the work type: DcsItem.

If the input exceeds 500 Item UDAs an input too large exception will be returned.

A "Forbidden" response will occur if application is integrated with MFCS.

API Basics

Endpoint URL	{base URL}/items/import
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	Item UDA list
Output	None
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
itemUdas	List of details	x	A list of associations between an item and user defined attributes.

Detail Data Definition

Attribute	Data Type	Required	Description
itemId	String	X	The item identifier.
udaId	Integer	X	The user defined attribute identifier.
udaDate	Date		Holds the value of the user defined attribute if it is a date.
udaText	String (250)		Holds the value of the user defined attribute if it is text.
udaLovId	String (25)		Holds the unique numeric identifier of the user defined attribute if it is a list of values selection.
print	Boolean		Y indicates printing is done for this item and user defined attribute (which is also controlled by the UDA).

API: Delete Item Udas

Deletes an association between item and user defined attributes. This is controlled by the work type: DcsItem.

If the input exceeds 500 Item UDAs an input too large exception will be returned.

A "Forbidden" response will occur if application is integrated with MFCS.

API Basics

Endpoint URL	{base URL}/items/delete
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	UDA item delete list
Output	None
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
itemUdas	List of details	Yes	A list of associations between item and user defined attribute identifiers.

Detail Data Definition

Attribute	Data Type	Required	Description
itemId	String (25)	Yes	The item identifier.
udaId	Integer (5)	Yes	The user defined attribute identifier.
udaDate	Date		Holds the value of the user defined attribute if it is a date.
udaText	String (250)		Holds the value of the user defined attribute if it is text.
udaLovId	String (25)		Holds the unique numeric identifier of the user defined attribute if it is a list of values selection.

API: Find Item Udas

It will retrieve UDAs for the inputted items.

API Basics

Endpoint URL	{base URL}/items/find

Method	POST
Successful Response	200 OK
Processing Type	Synchronous
Input	Item list
Output	UDA item list
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
itemIds	List<String>	Yes	A list of items to find UDAs for.

Output Data Definition

Attribute	Data Type	Description
itemId	String	The item identifier.
udaId	Integer	The user defined attribute identifier.
type	String	The user defined attribute type: (LV) List of Value, (FF) Free Form Text, DT (Date)
description	String	The description of the user defined attribute.
udaDate	Date	Holds the value of the user defined attribute if it is a date.
udaText	String	Holds the value of the user defined attribute if it is text.
udaLovId	Long	Holds the unique numeric identifier of the user defined attribute if it is a list of values selection.
udaLoveDescription	String	Holds the value description of the UDA List of Values selection.
printTicket	Boolean	True indicates tickets are printed for this user defined attribute.
printLabel	Boolean	True indicates labels are printed for this user defined attribute.

Additional Data Definitions

UDA Type

Value	Definition
1	Date
2	Free Form Text
3	List of Value

REST Service: Item UIN

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/uins

API Definitions

API	Description
createUin	Create a new unique identification number.
readUin	Reads information about a Universal Identification Number.
findUins	Find unique identification number summary information based on search criteria.
findUinLabels	This API is used to find all the UIN labels available for a uin items.
findUinHistory	This API is used to find UIN historical information based on search criteria.
generateUins	This API generates new Type 2 (Auto Generated Serial Numbers) Universal Identification Numbers without changing store inventory positions.

API: createUin

Create a new unique identification number. Note that the combination of store and item determines the administrative information about a UIN (such as UIN Type).

The newly created UIN will be in "Unconfirmed" status and its transaction type will be "UIN Web Service." To move it into inventory, use inventory adjustment or another transaction.

API Basics

Endpoint URL	{base URL}
Method	POST
Success Response	200 OK
Processing Type	Synchronous
Input	UIN information
Output	UIN confirmation information

Input Data Definition

Payload	Type	Req	Definition
storeId	Long	X	The identifier of the store.
itemId	String	X	The identifier of the item.

uin	String	X	The universal identification number.

Output Data Definition

Payload	Type	Definition
itemUinId	Long	The unique identifier to the record.
storeId	Long	The identifier of the store.
itemId	String	The identifier of the item.
uin	String	The universal identification number.
status	Integer	The current status of the UIN. Valid values are in index.

Example

```
{
  "storeId": 5000,,
   "itemId": "100700500",
   "uin": "1234"
}
```

API: readUin

Reads information about a Universal Identification Number.

API Basics

Endpoint URL	{base URL}/items/{itemId}/{uin}
Method	GET
Success Response	200 OK
Processing Type	Synchronous
Input	Path Parameters Item identifier and UIN
Output	UIN information

Output Data Definition

Payload	Type	Definition
itemUinId	Long	A unique identifier representing the record in the database.
itemId	String	The identifier of the item.
uin	String	The universal identification number.
type	Integer	The type of UIN. Valid values are: (1) Serial Number, (2) Auto Generated Serial Number
status	Integer	The current status of the UIN. See Index for valid values.
storeId	Long	The store identifier

transactionType	Integer	The business area that last contained the UIN,
transactionId	String	The transaction id of the transaction containing the UIN.
cartonId	String	The identifier of the carton containing the UIN.
nonsellableTypeId	Long	A non-sellable inventory bucket the UIN was within.
previousStatus	Integer	The previous status of the UIN. Valid values are in index.
previousStoreId	Long	The previous store identifier associated with the previous status.
previousTransactionType	Integer	The previous business area that contained the UIN for that previous status.
previousTransactionId	String	The transaction id of the transaction that previously contained the UIN for that previous status.
previousCartonId	String	The identifier of the carton that previously container the UIN for that previous status.
previousNonsellableTypeId	Long	A non-sellable inventory bucket the UIN was last within for that previous status.
damaged	Boolean	True if the UIN is damaged, N otherwise.
createDate	Date	The date the UIN was first inserted into the system.
updateDate	Date	The last date the UIN was updated.
createUser	String	The user that first inserted the UIN into the system.
updateUser	String	The user that last updated the UIN in the system.

API: findUins

Find unique identification number summary information based on search criteria.

API Basics

Endpoint URL	{base URL}
Method	GET
Success Response	200 OK
Processing Type	Synchronous
Input	Query Parameters
Output	List of UINs (see ReadUIN API for Data Output)
Maximum Results Allowed	10,000

Input Data Definition

Attribute	Type	Definition
storeId	Long	Include only UINs for this store identifier.
itemId	String	Include only UINS for this item
status	Integer	Include only UINs with this current status.
updateDateFrom	Date/String	Include only UINs updated on or after this date.
updateDateTo	Date/String	Include only UINs updated on or before this date.

API: findUinLabels

This API is used to find all the UIN labels available for a uin items.

API Basics

Endpoint URL	{base URL}/labels
Method	GET
Success Response	200 OK
Processing Type	Synchronous
Input	N/A
Output	List of labels

Output Data Definition

Attribute	Type	Definition
labelId	Long	The unique identifier of the record.
labelCode	String	A unique code that defines the label.
description	String	The description or label associated to the code (not translated).

API: findUinHistory

This API is used to find UIN historical information based on search criteria.

API Basics

Endpoint URL	{base URL}/histories
Method	GET
Success Response	200 OK
Processing Type	Synchronous
Input	Query parameters
Output	List of UIN history records
Maximum Results Allowed	10,000

Input Data Definition

Attribute	Type	Definition
itemId	String	Include only UIN history for this item.
uin	Integer	Include only UIN history for this UIN.
createDateFrom	Date/String	Include only UIN history created on or after this date.
createDateTo	Date/String	Include only UIN history created on or before this date.

Output Data Definition

Payload	Type	Definition
itemId	String	The identifier of the item.
uin	String	The universal identification number.
type	Integer	The type of UIN. Valid values are: (1) Serial Number, (2) Auto Generated Serial Number
status	Integer	The current status of the UIN. Valid values are in index.
storeId	Long	The store identifier
transactionType	Integer	The business area that last contained the UIN,
transactionId	String	The transaction id of the transaction containing the UIN.
cartonId	String	The identifier of the carton containing the UIN.
nonsellableTypeId	Integer	A non-sellable inventory bucket the UIN was within.
createDate	Date	The date the UIN was first inserted into the system.
updateDate	Date	The last date the UIN was updated.
createUser	String	The user that first inserted the UIN into the system.
updateUser	String	The user that last updated the UIN in the system.

API: generateUins

This API generates new Type 2 (Auto Generated Serial Numbers) Universal Identification Numbers without changing store inventory positions.

If the UIN administrative data for the item and store used do not indicate Type 2, an error will be returned.

The new UINs generated will have a status of “Unconfirmed”.

API Basics

Endpoint URL	{base URL}/generate
Method	POST
Success Response	200 OK
Processing Type	Synchronous
Input	UIN generation information
Output	N/A

Input Data Definition

Payload	Type	Req	Definition
itemId	String	X	The identifier of the item.
storeId	Long	X	The identifier of the store

quantity	Integer	X	The amount of universal identification numbers to generate.
transactionType	Integer	X	See Index: UIN Functional Area
transactionId	String		A transaction reference identifier.

Example

```
{
  "storeId": 5000,
  "itemId": "100663071",
  "uin": "1234",
 "quantity": 5,
   "transactionType": 13
}
```

Additional Data Definitions

UIN Type

ID	Description
1	Serial Number
2	Auto-Generated Serial Number

UIN Status

ID	Description
1	In Stock
2	Sold
3	Shipped To Warehouse
4	Shipped To Store
5	Reserved For Shipping
6	Shipped To Vendor
7	Removed From Inventory
8	Unavailable
9	Missing
10	In Receiving
11	Customer Order Reserved
12	Customer Order Fulfilled
13	Shipped To Finisher
99	Unconfirmed

UIN Functional Area

ID	Description

1	Warehouse Delivery Receipt
2	Direct Delivery Receipt
3	Create Transfer
4	Dispatch Transfer
5	Receive Transfer
6	Receipt Adjustment
7	Create Return
8	Dispatch Return
9	Inventory Adjustment
10	Stock Count
11	Stock Recount
12	Stock Count Authorized
13	Manual
14	POS Sale
15	POS Return
16	POS Sales Void
17	POS Return Void
18	UIN Web Service
19	Customer Order
20	Direct Delivery ASN Inbound
21	Transfer ASN Inbound
22	Transfer Shipment

REST Service: Manifest

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/manifests

API Definitions

API	Description
Close Manifest	Call this method to close all manifested shipments matching the input criteria.

API: Close Manifest

Call this method to close all manifested shipments for the carrier code and carrier services. A processing message is sent to the internal message processing system and the services

returns an “Accepted” response. The closing of the manifest will occur later when the message is processed.

API Basics

Endpoint URL	{base URL}/close
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	Criteria
Output	None
Max Response Limit	-

Input Data Definition

Attribute	Data Type	Required	Description
carrierCode	String (4)	Yes	A carrier code.
carrierServiceCode	String (6)	Yes	A carrier service code.
trackingNumber	List of Strings (120)	Yes	A list of tracking numbers associated to the contents of the manifest.
shipDate	Date	-	Indicates all items manifested prior to this date for the carrier have been shipped.

Example Input

```
{
    "carrierCode": "O",
    "carrierServiceCode": "O",
    "trackingIds": ["7861","45722"],
    "shipDate":"2022-04-19T23:59:59-05:00"
}
```

REST Service: POS Transaction

This service retrieves information about item inventory.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/postransactions

API Definitions

API	Description
Import POS Transactions	Imports point-of-sale transactions.

API: Import POS Transactions

POS may integration its transaction to EICS using this web service. The service imports and process point-of-sale transactions through an asynchronous process. The service has a default limit of 1000 total items, though they may be distributed across any number of transactions. Only one store is allowed across all the transaction sent in a single requires.

The web service is optimized for speed at greater than 400 items and less than 500 items per service call. The further above or below this optimized point, processing speed will be reduced, and it may take longer for the sales to be recorded in inventory.

Since this import is asynchronous, only the form is validated prior to the data being captured and processed later. See Sales Integration for additional information about processing.

POS Transaction ID and Sales Audit

In order for the sales audit process to match and audit a POS transaction that was previously recieved, the transactionId attribute of the POS Transaction must use the same data elements and format as the sales audit file. The sales audit file concatenates the THEAD file line id value, plus a dash separator, plus the THEAD transaction number clume value. For example, 111-222. The transactionId attribute of the POS transaction must match these values and format.

API Basics

Endpoint URL	{base URL}
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	List of transactions
Output	None
Max Input Limit	-

Input Data Definition

Attribute	Data Type	Required	Description
Transactions	List of PosTransactions	Yes	A list of transactions to process.

Pos Transaction Data Definition

Attribute	Data Type	Required	Description
storeId	Long	Yes	The unique identifier of the store that is the source of the transaction. This attribute is used to synchronize with sale audit.
transactionId	String (128)	Yes	A unique identifier of the point-of-sale transaction. This attribute is used to synchronize with sale audit.
transactionTimestamp	Date	Yes	The date and time of the transaction.
custOrderId	String (128)	-	An external customer order identifier. This attribute is required for customer order related point-of-sale transactions.
CustomerOrderComm	String (512)	-	A comment associated to the customer order.
externalUser	String (128)	-	User information from the external point-of-sale system.
items	List of Items	Yes	A collection of items belonging to the transaction.

Pos Transaction Item Data Definition

Attribute	Data Type	Required	Description
itemId	String (25)	Yes	The transaction-level SKU number of the item.
quantity	BigDecimal	Yes	The quantity of the item transacted.
unitOfMeasure	String (4)	Yes	Unit of measure of the quantity.
uin	String (128)	-	The unique identifier number (serial number). If not empty, the quantity will be overwritten with a quantity of one.
epc	String (256)	-	A complete SGTIN-96 EPC of the item.
reasonCode	Integer	-	A reason code associated to the line item. This reason code represents the inventory movement reason. If the field is left blank, the appropriate inventory movement reason code will be defaulted based on the type of sale. This field is required when non-sellable sub-level inventory tracking is active in the system.
dropShip	Boolean	-	True if this item is a drop ship, false if it is not. Drop ship sales do not impact stock positions.
fulfillOrderId	String (128)	-	If the transaction is associated to a customer order, this is the external fulfillment order identifier.

Attribute	Data Type	Required	Description
fulfillmentOrderLineNumber	Long	-	If the transaction is associated to a customer order, this is the line number of the order that this item transaction aligns with.
reservationType	Integer	-	If the transaction is a customer order, this is the type of reservation. See Reservation Type.
transactionCode	Integer	Yes	A code that indicates the transaction event that took place on the item. See Transaction Codes.
comments	String (512)	-	Comments associated to the line item.

Example Input

{
"transactions":
[
{
"storeId": 5000,
"transactionId": 1236,
"transactionTimestamp": "2022-04-19T23:59:59-05:00",
"externalUser": "ABC",
"custOrderId": "1111",
"items":
[
{
"itemId": 5678,
"transactionCode": 5,
"reservationType": 1,
"fulfillOrderId": "2222",
"dropShip": false
}
]
}
]
}

Possible Business Exception Codes

In addition to the normal REST error codes, the following business data element may be returned when a business error occurs.

Business Exception Data Definition

Name	Definition
DUPLICATE_TRANSACTION	One or more the transactions or transaction items are not unique. This will cause the entire request to be rejected.

Additional Data Definitions

Reservation Type

Value	Definition
1	Web Order
2	Special Order
3	Pickup or Delivery
4	Layaway
5	On Hold

Transaction Code

Value	Definition
1	Sale
2	Return
3	Void Sale
4	Void Return
5	Order New
6	Order Fulfill
7	Order Cancel

REST Product Group

This service allows for the integration of product group with external systems.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/productgroups

APIs

API	Description
findProductGroups	Searches for product group summary headers based on input criteria.
readProductGroup	Reads the details about a specific product group.
createProductGroup	Creates a new product group including its components.
updateProductGroup	Modifies an existing product group including its components.
cancelProductGroup	Cancels an existing product group.

API: findProductGroup

This API is used to search for a product group summary using input criteria.

Table 4-13 API Basics

Endpoint URL	{base URL}
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	Query parameters
Output	Collection of product groups
Max Response Limit	5,000

Table 4-14 Query Parameters

Query	Type	Definition
storeId	Long(10)	Returns only product groups that match this store (which includes all store product groups).
type	Integer(3)	Returns only product groups that match this product group type. See Index for types.
description	String(100)	Returns only products groups that contain this text in its description.
itemId	String(25)	Returns only product groups that contain this individual sku number in its single items list.
departmentId	Long(12)	Returns only product groups that contain this hierarchy department identifier.
classId	Long(12)	Returns only product groups that contain this hierarchy class identifier.
subclassId	Long(12)	Returns only product groups that contain this hierarchy subclass identifier.
updateDateFrom	String	Returns only product groups updated on or after this date.
updateDateTo	String	Returns only product groups updated on or before this date.

Table 4-15 Output Data Definitions

Payload	Type	Definition

Table 4-15 (Cont.) Output Data Definitions

productGroupId	Long(12)	The unique product group identifier.
description	String(100)	A description of the product group.
storeId	Long(10)	The store identifier or blank if the product group is for all stores.
type	Integer(3)	Indicates the type of product group. See Index.
status	Integer(1)	Indicates the status of the product group. See Index.
updateDate	Date	The date the product group was last updated.

API: readProductGroup

This API is used to read a product group.

Table 4-16 API Basics

Endpoint URL	{base URL}/{productGroupId}
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Path Parameter	The identifier of the product group.
Output	Collection of product groups
Max Response Limit	5,000

Table 4-17 Output Data Definition

Payload	Type	Definition
productGroupId	Long(12)	The unique identifier of the product group.
description	String(100)	A description of the product group.
storeId	Long(10)	The unique identifier of a store associated to the product group. This will be blank/null if the product group is for all stores.
type	Integer(3)	Indicates the type of product group. See Index.
status	Integer(1)	Indicates the status of the product group. See Index.
unitOfMeasureMode	Integer(4)	Unit of measure type for pick list product groups. See Index.
allItemGroup	Boolean	True indicates the product group is for all items. If set to true, then hierarchies and items will not be present.
updateDate	Date	The date the product group was last updated.
stockCountDetail	Object	Details about a stock count product group (used for all three stock count types). This will only be populated if the product group type is for a stock count.
replenishmentDetail	Object	Details about a replenishment product group. This will only be populated if the product group type is for replenishment.

Table 4-17 (Cont.) Output Data Definition

storeOrderDetail	Object	Details about a store order product group. This will only be populated if the product group type is for store orders.
autoAdjustmentDetail	Object	Details about an auto inventory adjustment product group. This will only be populated if the product group type is for auto adjustments.
autoTicketPrintDetail	Object	Details about an auto ticket print product group. This will only be populated if the product group type is for auto ticket print.
hierarchies	List<Object>	A list of merchandise item hierarchies that belong to the group.
itemIds	List<String(25)>	A list of item SKU numbers associated to the product group.

Table 4-18 Stock Count Detail

Payload	Type	Definition
countingMethod	Integer	The method of counting a stock count: See Index (StockCountingMethod)
breakdownType	Integer	The method of breaking a stock count down into child counts: See Index (StockCountBreakdownType)
varianceCount	Integer(8)	The number of units in standard unit of measure considered discrepant on a stock count.
variancePercent	BigDecimal(12,4)	The percentage of units in standard unit of measure considered discrepant on a stock count.
varianceValue	BigDecimal(12,4)	The value of units considered discrepant on a stock count. This will only be populated for unit and amount stock count product group type.
performRecount	Boolean	True indicates the stock count should be recounted if discrepancies are found.
autoAuthorize	Boolean	True indicates the stock count should be automatically authorized after count.
activeItems	Boolean	True indicates active items should be included on the stock count.
inactiveItems	Boolean	True indicates inactive items should be included on the stock count.
discontinuedItems	Boolean	True indicates discontinued items should be included on the stock count.
deletedItems	Boolean	True indicates deleted items should be included on the stock count.
zeroStockOnHand	Boolean	True indicates items with 0 stock on hand should be included on the stock count.
positiveStockOnHand	Boolean	True indicates items with positive stock on hand should be included on the stock count.
negativeStockOnHand	Boolean	True indicates items with negative stock on hand should be included on the stock count.

Table 4-18 (Cont.) Stock Count Detail

problemLineNegativeAvailable	Boolean	True indicates to include items with negative available quantities. This will only be populated if the product group type is for a problem line stock count.
problemLinePickLessSuggested	Boolean	True indicates to include pick lists less than suggested amount. This will only be populated if the product group type is for a problem line stock count.
problemLineReplenishLessSuggested	Boolean	True indicates to include shelf replenishments that are less than the suggested amount. This will only be populated if the product group type is for a problem line stock count.
problemLineUinDiscrepancy	Boolean	True indicates UIN discrepancies should be included in the stock count. This will only be populated if the product group type is for a problem line stock count.

Table 4-19 Replenishment Detail

Payload	Type	Definition
autoReplenishment	Boolean	True indicates that the product group is eligible for automatic replenishment through a scheduled batch run.
differentiatorMode	Integer(3)	The display type of the diffentiator. See Index.

Table 4-20 Store Order Detail

Payload	Type	Definition
daysBeforeDelivery	Integer(3)	The date calculated from the creation date that the user wants the store order delivery by.
storeOrderAddItems	Boolean	True indicates items can be added to the store order after it has been generated by the system.
storeOrderItemsOnly	Boolean	Y indicates the store order can only include items that are marked as store order replenishment.
restrictSupplierId	Long(12)	Limits the creation of a store order from this product group to only this supplier.
restrictWarehouseId	Long(12)	Limits the creation of a store order from this product group to only this warehouse.
restrictHierarchyId	Long(12)	Limits the creation of a store order from this product group to only this merchandise hierarchy.
restrictAreaId	Long(12)	Limits the creation of a store order from this product group to only this store sequence area.

Table 4-21 Auto Adjustment Detail

Payload	Type	Definition
adjustmentQuantity	BigDecimal(20,4)	The quantity that should be automatically adjusted by this product group.
adjustmentPercent	BigDecimal(20,4)	The percentage of quantity that should be automatically adjusted by this product group.

Table 4-21 (Cont.) Auto Adjustment Detail

adjustmentReasonCode	Long(12)	The unique identifier of a reason code associated to this product group.
updateToZero	Boolean	Update stock to zero when generating adjustment with this product group.

Table 4-22 Auto Ticket Print Detail

Payload	Type	Definition
refreshTicketQuantity	Boolean	True if the ticket quantity should be refreshed prior to printing.

Table 4-23 Item Hierarchy Detail

Payload	Type	Definition
departmentId	Long(12)	The department identifier
classId	Long(12)	The class identifier
subclassId	Long(12)	The subclass identifier

API: createProductGroup

This API is used to create a new product group that will be **in progress** status.

Table 4-24 API Basics

Endpoint URL	{base URL}
Method	POST
Successful Response	200 OK
Processing Type	Synchronous
Input	Product Group
Output	Product Group Status
Max Response Limit	5,000

Table 4-25 Input Data Definition

Payload	Type	Req	Definition
description	String(100)	X	A description of the product group.
type	Integer(3)	X	Indicates the type of product group. See Index.
storeId	Long(10)		The unique identifier of a store associated to the product group. This will be blank/null if the product group is for all stores.
allItems	Boolean		True if the product group is for all items, false otherwise. If all items, any hierarchies and items included will be ignored.

Table 4-25 (Cont.) Input Data Definition

unitOfMeasureMode	Integer	Unit of measure type for pick lists. See Index..
autoAdjustmentDetail	Object	Details about an auto inventory adjustment product group. This is only processed and is required if the type is auto inventory adjustment type.
autoTicketPrintDetail	Object	Details about an auto ticket print product group. This is only processed and is required if the type is auto ticket print.
replenishmentDetail	Object	Details about a replenishment product group. This is only processed and is required if the type is a shelf replenishment type..
storeOrderDetail	Object	Details about a store order product group. This is only processed and is required if the type is the store order type.
unitCountDetail	Object	Details about a unit stock count product group. This is only processed and is required if the type is a stock count type.
unitAmountCountDetail	Object	Details about a unit and amount stock count product group. This is only processed and is required if the type is a stock count type.
problemLineCountDetail	Object	Details about a problem line stock count product group. This is only processed and is required if the type is a stock count type.
hierarchies	List<Object>	A list of up to 5,000 unique merchandise item hierarchies that belong to the group (avoid duplicate or overlapping hierarchies).
itemIds	List<String(25)>	A list of up to 5,000 unique item SKU numbers on the product group.

Table 4-26 Unit Count Detail

Payload	Type	Req	Definition
countingMethod	Integer	X	The method of counting a stock count: See Index (StockCountingMethod)
breakdownType	Integer	X	The method of breaking a stock count down into child counts: See Index (StockCountBreakdownType). Must be "Location" for guided counts.
varianceCount	Integer(8)		The number of units in standard unit of measure considered discrepant on a stock count. Variance count and percent cannot both be empty.
variancePercent	BigDecimal(12,4)		The percentage of units in standard unit of measure considered discrepant on a stock count. Variance count and percent cannot both be empty.
performRecount	Boolean		True indicates the stock count should be recounted if discrepancies are found. This cannot be set to true if the counting method is "Third Party.

Table 4-26 (Cont.) Unit Count Detail

autoAuthorize	Boolean	True indicates the stock count should be automatically authorized after count.
activeItems	Boolean	True indicates active items should be included on the stock count. At least one item status choice is required as true.
inactiveItems	Boolean	True indicates inactive items should be included on the stock count. At least one item status choice is required as true.
discontinuedItems	Boolean	True indicates discontinued items should be included on the stock count. At least one item status choice is required as true.
deletedItems	Boolean	True indicates deleted items should be included on the stock count. At least one item status choice is required as true.
zeroStockOnHand	Boolean	True indicates items with 0 stock on hand should be included on the stock count. At least one stock choice is required as true.
positiveStockOnHand	Boolean	True indicates items with positive stock on hand should be included on the stock count. At least one stock choice is required as true.
negativeStockOnHand	Boolean	True indicates items with negative stock on hand should be included on the stock count. At least one stock choice is required as true.

Table 4-27 Unit and Amount Count Detail

Payload	Type	Req	Definition
countingMethod	Integer	X	The method of counting a stock count: See Index (StockCountingMethod)
breakdownType	Integer	X	The method of breaking a stock count down into child counts: See Index (StockCountBreakdownType). Must be "Location" for guided counts.
varianceCount	BigDecimal(12,4)		The number of units in standard unit of measure considered discrepant on a stock count. Count, percent, and value cannot all be empty.
variancePercent	BigDecimal(12,4)		The percentage of units in standard unit of measure considered discrepant on a stock count. Count, percent, and value cannot all be empty.
varianceValue	BigDecimal(12,4)		The value of units considered discrepant on a stock count. Count, percent, and value cannot all be empty.
performRecount	Boolean		True indicates the stock count should be recounted if discrepancies are found. This cannot be set to true if the counting method is "Third Party".
autoAuthorize	Boolean		True indicates the stock count should be automatically authorized after count.

Table 4-27 (Cont.) Unit and Amount Count Detail

zeroStockOnHand	Boolean		True indicates items with 0 stock on hand should be included on the stock count.

Table 4-28 Problem Line Count Detail

Payload	Type	Req	Definition
countingMethod	Integer	X	The method of counting a stock count: See Index (StockCountingMethod).
breakdownType	Integer	X	The method of breaking a stock count down into child counts: See Index (StockCountBreakdownType). Must be "Location" for guided counts.
autoReplenishment	Boolean		True indicates that the product group is eligible for automatic replenishment through a scheduled batch run.
differentiatorMode	Integer(3)		The display type of the differentiator. See Index.
varianceCount	BigDecimal(12,4)		The number of units in standard unit of measure considered discrepant on a stock count. Variance count and percent cannot both be empty.
variancePercent	BigDecimal(12,4)		The percentage of units in standard unit of measure considered discrepant on a stock count. Variance count and percent cannot both be empty
performRecount	Boolean		True indicates the stock count should be recounted if discrepancies are found. This cannot be set to true if the counting method is "Third Party."
autoAuthorize	Boolean		True indicates the stock count should be automatically authorized after count.
negativeStockOnHand	Boolean		True indicates items with negative stock on hand should be included on the stock count.
negativeAvailable	Boolean		True indicates to include items with negative available quantities.
pickLessSuggested	Boolean		True indicates to include pick lists less than suggested amount.
replenishLessSuggested	Boolean		True indicates to include shelf replenishments that are less than the suggested amount.
uinDiscrepancy	Boolean		True indicates UIN discrepancies should be included in the stock count. This will only be populated if the product group type is for a problem line stock count.

Table 4-29 Store Order Detail

Payload	Type	Req	Definition

Table 4-29 (Cont.) Store Order Detail

daysBeforeDelivery	Integer(3)	The date calculated from the creation date that the user wants the store order delivery by.
storeOrderAddItems	Boolean	True indicates items can be added to the store order after it has been generated by the system.
storeOrderItemsOnly	Boolean	Y indicates can only include items that are marked as store order replenishment.
restrictSupplierId	Long(12)	Limits the creation of a store order from this product group to only this supplier.
restrictWarehouseId	Long(12)	Limits the creation of a store order from this product group to only this warehouse.
restrictHierarchyId	Long(12)	Limits the creation of a store order from this product group to only this merchandise hierarchy.
restrictAreaId	Long(12)	Limits the creation of a store order from this product group to only this store sequence area.

Table 4-30 Auto Adjustment Detail

Payload	Type	Req	Definition
adjustmentQuantity	BigDecimal(20, 4)		The quantity that should be automatically adjusted by this product group. Either quantity or percent is required.
adjustmentPercent	BigDecimal(20, 4)		The percentage of quantity that should be automatically adjusted by this product group. Either quantity or percent is required.
adjustmentReasonCode	Long(12)	X	The unique identifier of a reason code associated to this product group.
updateToZero	Boolean		Update stock to zero when generating adjustment with this product group.

Table 4-31 Auto Ticket Print Detail

Payload	Type	Req	Definition
refreshTicketQuantity	Boolean		True if the ticket quantity should be refreshed prior to printing.

Table 4-32 Item Hierarchy Detail

Payload	Type	Req	Definition
departmentId	Long(12)	X	The department identifier
classId	Long(12)		The class identifier
subclassId	Long(12)		The subclass identifier

Table 4-33 Output Data Definition

Attribute	Type	Definition
productGroupId	Long(12)	The product group identifier.
status	Integer(1)	Indicates the status of the product group: (0) Active, (1) Canceled

Figure 4-1 Example input unitCountDetail


```
{
   "description": "product group type
1",
   "type": 1,
   "storeId": 5000,
   "allItems":false,
   "unitOfMeasureMode":1,
   "unitCountDetail":{
      "countingMethod":4,
      "breakdownType":1,
      "varianceCount":1,
      "variancePercent":1,
      "performRecount":true,
      "autoAuthorize":true,
      "activeItems":true,
      "inactiveItems":true,
      "discontinuedItems":true,
      "deletedItems":true,
      "zeroStockOnHand":true,
      "positiveStockOnHand":true,
      "negativeStockOnHand":true
   },
   "hierarchies": [
      {
         "departmentId":2345,
         "classId":2,
         "subclassId":1
      },{
         "departmentId":4567,
         "classId":1000,
         "subclassId":5000
      }
   ],
   "itemIds": [
"100000001"
   ]
}
```

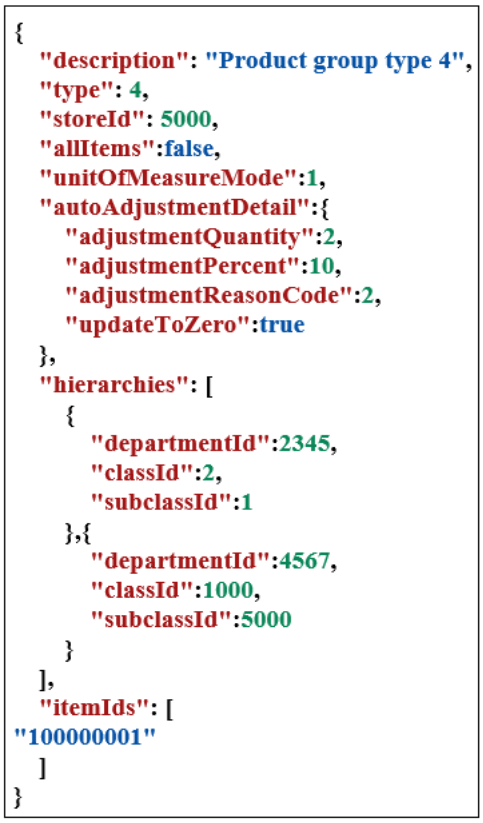
Figure 4-2 Example input unitAmountCountDetail:

```
{
   "description": "Product group type 2",
   "type": 2,
   "storeId": 5000,
   "allItems":false,
   "unitOfMeasureMode":1,
   "unitAmountCountDetail":{
      "countingMethod":4,
      "breakdownType":1,
      "varianceCount":1,
      "variancePercent":1,
      "variannceValue":1,
      "performRecount":false,
      "autoAuthorize":false,
      "zeroStockOnHand":false
   },
   "hierarchies": [
      {
         "departmentId":2345,
         "classId":2,
         "subclassId":1
      },{
         "departmentId":4567,
         "classId":1000,
         "subclassId":5000
      }
   ]
}
```

Figure 4-3 Example input problemLineCountDetail:

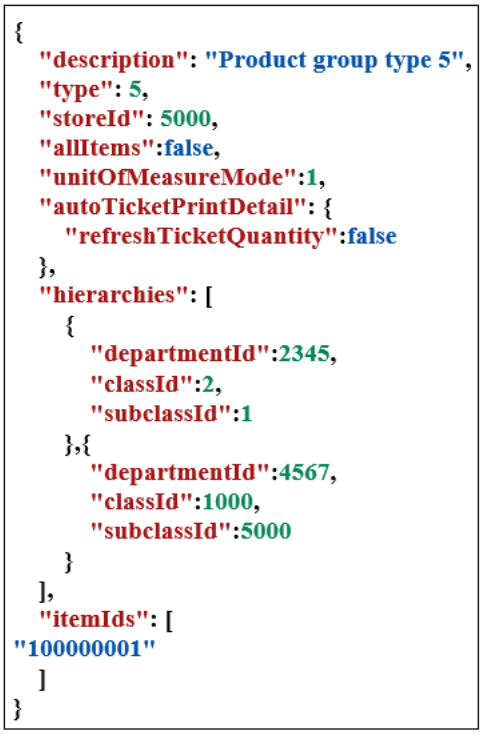

```
{
   "description": "Product group type 3",
   "type": 3,
   "storeId": 5000,
   "allItems":false,
   "unitOfMeasureMode":1,
   "problemLineCountDetail":{
      "countingMethod":4,
      "breakdownType":1,
      "varianceCount":1,
      "variancePercent":1,
      "performRecount":true,
      "autoAuthorize":true,
      "negativeStockOnHand":true,
      "negativeAvailable":true,
      "pickLessSuggested":true,
      "replenishLessSuggested":true,
      "uinDiscrepancy":true
   },
   "hierarchies": [
      {
         "departmentId":2345,
         "classId":2,
         "subclassId":1
      },{
         "departmentId":4567,
         "classId":1000,
         "subclassId":5000
      }
   ],
   "itemIds": [
"100000001"
   ]
}
```

Figure 4-4 Example input autoAdjustmentDetail:



```
{
  "description": "Product group type 4",
  "type": 4,
  "storeId": 5000,
  "allItems":false,
  "unitOfMeasureMode":1,
  "autoAdjustmentDetail":{
    "adjustmentQuantity":2,
    "adjustmentPercent":10,
    "adjustmentReasonCode":2,
    "updateToZero":true
  },
  "hierarchies": [
    {
      "departmentId":2345,
      "classId":2,
      "subclassId":1
    },{
      "departmentId":4567,
      "classId":1000,
      "subclassId":5000
    }
  ],
  "itemIds": [
"100000001"
  ]
}
```

Figure 4-5 Example input autoTicketPrintDetail:



```
{
  "description": "Product group type 5",
  "type": 5,
  "storeId": 5000,
  "allItems":false,
  "unitOfMeasureMode":1,
  "autoTicketPrintDetail": {
    "refreshTicketQuantity":false
  },
  "hierarchies": [
    {
      "departmentId":2345,
      "classId":2,
      "subclassId":1
    },{
      "departmentId":4567,
      "classId":1000,
      "subclassId":5000
    }
  ],
  "itemIds": [
"100000001"
  ]
}
```

Figure 4-6 Example input replenishmentDetail:


```
{
   "description": "Product group type 6",
   "type": 6,
   "storeId": 5000,
   "allItems":false,
   "unitOfMeasureMode":1,
   "replenishmentDetail": {
      "autoReplenishment":false,
      "differentiatorMode":1
   },
   "hierarchies": [
      {
         "departmentId":2345,
         "classId":2,
         "subclassId":1
      },{
         "departmentId":4567,
         "classId":1000,
         "subclassId":5000
      }
   ],
   "itemIds": [
"100000001"
   ]
}
```

Figure 4-7 Example input storeOrderDetail:

```
{
  "description": "Product group type 7",
  "type": 7,
  "storeId": 5000,
  "allItems":false,
  "unitOfMeasureMode":1,
  "storeOrderDetail":{
    "daysBeforeDelivery":4,
    "storeOrderAddItems":true,
    "storeOrderItemsOnly":true,
    "restrictSupplierId":9001,
    "restrictWarehouseId":7001,
    "restrictHierarchyId":173,
    "restrictAreaId":1
  },
  "hierarchies": [
    {
      "departmentId":2345,
      "classId":2,
      "subclassId":1
    },{
      "departmentId":4567,
      "classId":1000,
      "subclassId":5000
    }
  ],
  "itemIds": [
"100000001"
  ]
}
```

API: updateProductGroup

This API is used to modify a product group.

See API createProductGroup for the data definition of the product group type data objects.

API Basics

Table 4-34 API Basics

Endpoint URL	{base URL}/{productGroupId}
Method	POST
Successful Response	200 OK
Processing Type	Synchronous
Path Parameter	The identifier of the product group
Input	Product Group
Output	Product Group Status

Input Data Definition

Payload	Type	Req	Definition
storeId	Long(10)		The unique identifier of a store associated to the product group. This will be blank/null if the product group is for all stores. If this is altered for a unit and amount group already assigned to schedules, it may return a validate error that it can no longer be altered.
description	String(100)	X	A description of the product group.
allItems	Boolean		True if the product group is for all items, false otherwise. If all items, any hierarchies and items included will be ignored. Unit of measure type for pick lists. See Index.
unitOfMeasureMode	Integer		Unit of measure type for pick lists. See Index.
autoAdjustmentDetail	ProductGroupAdjustmentIdo		Details about an auto inventory adjustment product group. This is only processed and is required if the type is auto inventory adjustment type.
autoTicketPrintDetail	ProductGroupTicketPrintIdo		Details about an auto ticket print product group. This is only processed and is required if the type is auto ticket print.
replenishmentDetail	ProductGroupReplenishmentIdo		Details about a replenishment product group. This is only processed and is required if the type is a shelf replenishment type.
storeOrderDetail	ProductGroupStoreOrderIdo		Details about a store order product group. This is only processed and is required if the type is the store order type.
unitCountDetail	ProductGroupUnitCountIdo		Details about a unit stock count product group. This is only processed and is required if the type is a stock count type.
unitAmountCountDetail	ProductGroupUnitAmountCountIdo		Details about a unit and amount stock count product group. This is only processed and is required if the type is a stock count type.
problemLineCountDetail	ProductGroupProblemLineCountIdo		Details about a problem line stock count product group. This is only processed and is required if the type is a stock count type.
hierarchies	ProductGroupHierarchyIdo		A list of up to 5,000 unique merchandise item hierarchies that belong to the group (avoid duplicates or overlapping hierarchies).
itemIds	List<String(25)>		A list of up to 5,000 unique item SKU numbers on the product group.

Figure 4-8 Example Input unitCountDetail:

```
{
   "description": "unitCountDetail",
   "allItems":false,
   "unitOfMeasureMode":1,
   "unitCountDetail":{
      "countingMethod":2,
      "breakdownType":3,
      "varianceCount":100,
      "variancePercent":0,
      "performRecount":122,
      "autoAuthorize":123,
      "activeItems":123,
      "inactiveItems":123,
      "discontinuedItems":123,
      "deletedItems":123,
      "zeroStockOnHand":123,
      "positiveStockOnHand":123,
      "negativeStockOnHand":123
   },
   "hierarchies": [
      {
         "departmentId":1119,
         "classId":1,
         "subclassId":2
      }
   ],
   "itemIds": [
"100000147"
   ]
}
```

Figure 4-9 Example Input unitAmountCountDetail:

```
{
   "description": "unitAmountCountDetail",
   "allItems":false,
   "unitOfMeasureMode":1,
   "unitAmountCountDetail":{
      "countingMethod":4,
      "breakdownType":1,
      "varianceCount":1,
      "variancePercent":12,
      "varianceValue":12,
      "performRecount":false,
      "autoAuthorize":false,
      "zeroStockOnHand":false
   },
   "hierarchies": [
      {
         "departmentId":1119,
         "classId":1,
         "subclassId":2
      }
   ]
}
```

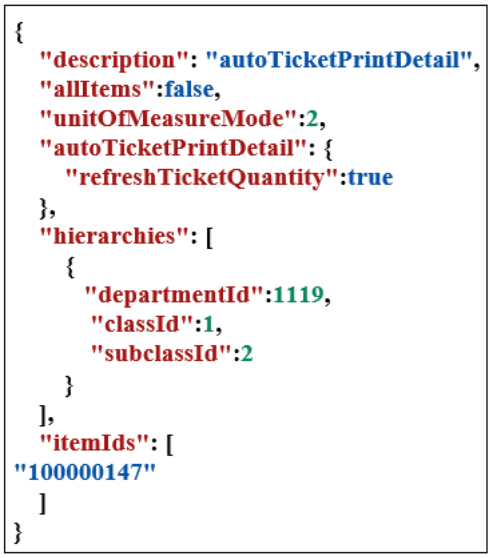
Figure 4-10 Example Input problemLineCountDetail:

```
{
   "description": "problemLineCountDetail",
   "allItems":false,
   "unitOfMeasureMode":2,
   "problemLineCountDetail":{
      "countingMethod":2,
      "breakdownType":2,
      "varianceCount":1,
      "variancePercent":1,
      "performRecount":true,
      "autoAuthorize":true,
      "negativeStockOnHand":true,
      "negativeAvailable":true,
      "pickLessSuggested":true,
      "replenishLessSuggested":true,
      "uinDiscrepancy":123
   },
   "hierarchies": [
      {
         "departmentId":1119,
         "classId":1,
         "subclassId":2
      }
   ],
   "itemIds": [
"100000147"
   ]
}
```

Figure 4-11 Example Input autoAdjustmentDetail:

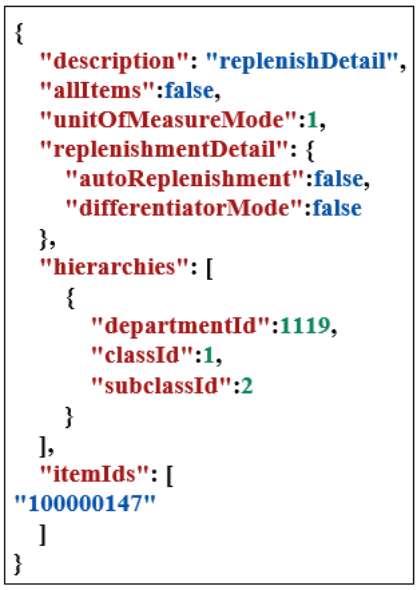
```
{
   "description": "autoAdjustmentDetail",
   "allItems":false,
   "unitOfMeasureMode":2,
   "autoAdjustmentDetail":{
      "adjustmentQuantity":2,
      "adjustmentPercent":2,
      "adjustmentReasonCode":3,
      "updateToZero":false
   },
   "hierarchies": [{
         "departmentId":1119,
         "classId":1,
         "subclassId":2
      }
   ],
   "itemIds": [
"100000147"
   ]
}
```

Figure 4-12 Example Input autoTicketPrintDetail:



```
{
   "description": "autoTicketPrintDetail",
   "allItems":false,
   "unitOfMeasureMode":2,
   "autoTicketPrintDetail": {
      "refreshTicketQuantity":true
   },
   "hierarchies": [
      {
        "departmentId":1119,
         "classId":1,
         "subclassId":2
      }
   ],
   "itemIds": [
"100000147"
   ]
}
```

Figure 4-13 Example Input replenishmentDetail:



```
{
   "description": "replenishDetail",
   "allItems":false,
   "unitOfMeasureMode":1,
   "replenishmentDetail": {
      "autoReplenishment":false,
      "differentiatorMode":false
   },
   "hierarchies": [
      {
         "departmentId":1119,
         "classId":1,
         "subclassId":2
      }
   ],
   "itemIds": [
"100000147"
   ]
}
```

Figure 4-14 Example Input storeOrderDetail:

```
{
  "description": "storeOrderDetail",
  "allItems":false,
  "unitOfMeasureMode":2,
  "storeOrderDetail":{
    "daysBeforeDelivery":5,
    "storeOrderAddItems":false,
    "storeOrderItemsOnly":false,
    "restrictSupplierId":9002,
    "restrictWarehouseId":7002,
    "restrictHierarchyId":false,
    "restrictAreaId":false
  },
  "hierarchies": [
    {
      "departmentId":"1119",
      "classId":"1",
      "subclassId":"2"
    }
  ],
  "itemIds": [
"100000147"
  ]
}
```

API: cancelProductGroup

Cancels the product group.

Table 4-35 API Basics

Endpoint URL	{base URL}/{productGroupId}/cancel
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Path Parameter	The identifier of the product group

Index

Table 4-36 Product Group Type

ID	Description
1	Stock Count: Unit
2	Stock Count: Unit and Amount
3	Stock Count: Problem Line
4	Auto Inventory Adjustment
5	Auto Ticket Print

Table 4-36 (Cont.) Product Group Type

6	Shelf Replenishment
7	Store Order

Table 4-37 Product Group Status

ID	Description
1	Active
2	Canceled

Table 4-38 Unit of Measure Mode

ID	Description
1	Standard
2	Cases
3	Preferred

Table 4-39 Shelf Replenishment Differentiator Mode

ID	Description
1	Differentiator 1
2	Differentiator 2
3	Differentiator 3
4	Differentiator 4

Table 4-40 Stock Counting Method

ID	Description
1	Guided
2	Unguided
3	Third Party
4	Auto

Table 4-41 Stock Count Breakdown Type

ID	Description
1	Department
2	Class
3	Subclass
4	Location
5	None

Rest Service: Security

This service provides an end point for the management of security information within SIOCS.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api

API Definitions

Table 4-42 Security API Definitions

API	Description
Import Users	Imports a collection of user security information changes and processes them synchronously. This allows up to 2,000 users.
Delete Users	Deletes security information for the specified users within SIOCS.

API: Import Users

Imports a collection of user security information changes and processes them synchronously. This allows up to 2,000 users.

Table 4-43 API Basics

Endpoint URL	/security/import/users
Method	POST
Successful Response	202 Accepted
Processing Type	synchronous
Input	Query parameters
Output	N/A
Max Response	2000

Table 4-44 Input Data Definition

Parameter	Format	Required	Definition
users	array	No	A list of security changes to import. See below:
Security Changes to Import			
userName	string(text128)	Yes	The user name of the user.
addedStores	array[Integer](int10)	No	The stores to be added to the user.
removedStores	array[Integer](int10	No	The stores to be removed from the user.

Table 4-44 (Cont.) Input Data Definition

addedRoles	array	No	The roles to be assigned to the user: roleName — The name of the role, string(text128), required storeId — The identifier of the store for a specific store assignment, or no value for assignment applying to all allowed stores., BigDecimal(int10), optional startDate — The date the role assignment should start to be allowed., date-time, optional endDate — The date the role assignment should stop being allowed., date-time, optional
removedRoles	array	No	The role assignments to be removed from the user: roleName — The name of the role, string(text128), required storeId — The identifier of the store for a specific store assignment, or no value for assignment applying to all allowed stores., BigDecimal(int10), optional

Example Input ImportUsers

```
{
  "users": [
    {
      "userName": "abc",
      "addedStores":[1000, 2000],
      "removedStores":[1001, 2001],
      "addedRoles": [
        {
          "roleName": "role_a",
          "storeId": 3000,
          "endDate": "2032-11-19T23:59:59-05:00"
        },
        {
          "roleName": "role_c",
          "startDate": "2022-11-19T23:59:59-05:00"
        },
        {
          "roleName": "role_e",
        }
      ],
      "removedRoles": [
        {
          "roleName": "role_c",
          "storeId": 3001
        },
        {
          "roleName": "role_d"
        }
      ]
    }
  ]
}
```

Table 4-45 Responses

Code	Description
202	Accepted

400	Bad Request
	Example:
	
	[
	{
	"errors": [
	{
	"code": 1,
	"description": "AN_ERROR_TOOK_PLACE",
	"dataElement": "item",
	"dataValue": "1234",
	"referenceElement": "store",
	"referenceValue": "123"
	}
	]
	}
	]

API: Delete Users

Deletes security information for the specified users within SIOCS.

Table 4-46 API Basics

Endpoint URL	/security/users/delete
Method	POST
Successful Response	202 Accepted
Processing Type	
Input	Query parameters
Output	N/A
Max Response	

Table 4-47 Query Parameter Definitions

Property	Format	Required	Definition

Table 4-47 (Cont.) Query Parameter Definitions

userNames	string(text128)	Yes	A list of user names to delete information for.
			Example:
			
			{
			"userNames": [
			"user123",
			"user456"
			]
			}

Table 4-48 Responses

Code	Description
202	Accepted

400	Bad Request
	Example:
	
	[
	{
	"errors": [
	{
	"code": 1,
	"description": "AN_ERROR_TOOK_PLACE",
	"dataElement": "item",
	"dataValue": "1234",
	"referenceElement": "store",
	"referenceValue": "123"
	}
	]
	}
	]

REST Service: Reason Code

This service allows and external system to retrieve available reason codes. Reason codes are attached to inventory adjustments or shipments.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/ reasoncodes

API Definitions

API	Description
Find Adjustment Reason Codes	Finds reason codes available for inventory adjustments.
Find Shipment Reason Codes	Finds reason codes available for shipments.

API: Find Adjustment Reason Codes

This API is used to find reason codes available for inventory adjustments.

API Basics

Endpoint URL	{base URL} /adjustments
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	None
Output	List of inventory adjustment reasons
Max Response Limit	N/A

Output Data Definition

Attribute	Data Type	Description
reasonId	Long	The unique identifier of the inventory adjustment reason code.
reasonCode	Integer	Unique reason code associated to external systems.
description	String	A description of the inventory reason code.
dispositionCode	Integer	The inventory disposition associated to the code.
dispositionDescription	String	A description of the inventory disposition associated to the code (not translated).
fromNonSellableType	Long	From unavailable sub-bucket (indicates the sub-bucket of disposition of stock movement)
fromNonSellableTypeDescription	String	The description of the from unavailable sub-bucket (not translated).
toNonSellableType	Long	To unavailable sub-bucket (indicates the sub-bucket of disposition of stock movement)
toNonSellableTypeDescription	String	The description of the to unavailable sub-bucket (not translated).

systemRequired	Boolean		True indicates the reason code is required for the system to function. A system required reason code cannot be deactivated.
displayable	Boolean		True indicates the reason code can be used by a transactional inventory adjustment created by an entity other than EICS internal service.
publish	Boolean	-	True indicating inventory movements with this reason code should be published to external systems

Example Input

[

{

"reasonId": 1,

"reasonCode": 1,

"description": “invAdjReason.1",

"dispositionCode": 4,

"dispositionDescription": "inventoryDisposition.ATS-DIST",

"systemRequired": true,

"displayable": false,

"publish": true

},

{

"reasonId": 2,

"reasonCode": 81,

"description": "invAdjReason.81",

"dispositionCode": 4,

"dispositionDescription": "inventoryDisposition.ATS-DIST",

"systemRequired": false,

"displayable": true,

"publish": true

}

]

API: Find Shipment Reason Codes

This API is used to find the reason codes available for shipments.

API Basics

Endpoint URL	{base URL} /shipments
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	None
Output	List of reasons codes
Max Response Limit	N/A

Output Data Definition

Attribute	Data Type	Description
reasonId	Long	The unique identifier of the inventory adjustment reason code.
reasonCode	String	Unique reason code associated to external systems.
description	String	A description of the inventory reason code (not translated).
type	Integer	The Shipment Reason Code Type: See the Shipment Reason Code Type.
useAvailable	Boolean	True if it should use available inventory, false otherwise.
nonSellableTypeId	Long	An identifier of associated unavailable sub-bucket (indicates the sub-bucket of disposition of stock movement)

Example Input

[

{

"reasonId": 1,

"reasonCode": "F",

"description": "shipmentReason.4.F",

"type": 4,

"useAvailable": true

},

{

“reasonId": 2,

"reasonCode": "O",

"description": "shipmentReason.4.O",

"type": 4,

"useAvailable": true

},

{

"reasonId": 3,

"reasonCode": "U",

"description": "shipmentReason.4.U",

"type": 4,

"useAvailable": false,

"nonSellableTypeId": 1

}

]

Additional Data Definitions

Shipment Reason Code Type

Value	Definition
1	Store
2	Supplier
3	Warehouse
4	Finisher
5	Customer

Rest Shipping

This service integrates various shipping support information with an external application. This is primarily various lookups of shipping data such as carriers and package sizes to use within shipping transactions.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/shipping

APIs

Table 4-49 APIs

API	Description
findCarriers	Finds all the carriers available for shipping.
findCarrierServices	Finds all the carrier services available for shipping.

Table 4-49 (Cont.) APIs

findCartonSizes	Finds the available sizes of cartons for shipping.
findWeightUoms	Finds all the various units of measurement of the carton measurements.
findMotives	Finds all the motives available for a shipping type.

API: findCarriers

This API is used to find all the carriers available for shipping.

Table 4-50 API Basics

Endpoint URL	/carriers
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Output	Collection of carriers

Table 4-51 Output Data Definition

Attribute	Type	Definition
carrierId	Long(10)	The unique identifier of the carrier.
code	String(4)	A unique character code for the carrier.
description	String(128)	A description or name of the carrier.
manifestType	Integer(1)	The carrier delivery manifest type (see Additional Data Definitions)

API: findCarrierServices

This API is used to find all the carrier services available for shipping.

Table 4-52 API Basics

Endpoint URL	/carrierservices
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Output	Collection of carriers

Table 4-53 Output Data Definition

Attribute	Type	Definition
carrierServiceId	Long(10)	The unique identifier of the carrier service.
code	String(6)	A unique character code for the carrier service.

Table 4-53 (Cont.) Output Data Definition

description	String(128)	A description or name of the carrier service.
averageDeliveryDays	Integer(4)	The average number of days it takes to deliver using this service.
carrierId	Long(10)	The unique identifier of the carrier this service is used with.
defaultService	Boolean	True is this is the default service type for the carrier, false otherwise.
weightRequired	Boolean	True is weight is required for this carrier service, false otherwise.
cartonSizeRequired	Boolean	True if dimensions/size is required for this carrier service, false otherwise.

API: findCartonSizes

This API is used to find all the carton services available for shipping at the particular store.

Table 4-54 API Basics

Endpoint URL	/cartonsizes/{storeId}
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Path Parameter	The store identifier of the store to retrieve carton sizes for.
Output	Collection of carton sizes

Table 4-55 Output Data Definition

Attribute	Type	Definition
cartonSizeId	Long(12)	A unique identifier of this particular carton size definition.
storeId	Long(10)	The store identifier the carton size is used at.
description	String(120)	A description of the carton size.
height	BigDecimal(12,4)	The height of the carton in units of measure.
width	BigDecimal(12,4)	The width of the carton in units of measure.
length	BigDecimal(12,4)	The length of the carton in units of measure.
unitOfMeasure	String(4)	The unit of measure of the height, width, and length.

API: findWeightUoms

This API is used to find all the weight unit of measures available for shipping.

Table 4-56 API Basics

Endpoint URL	/weightuoms
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Output	Collection of shipping weight units of measure

Attribute	Type	Definition
uom	String(4)	The unit of measure
description	String(120)	The unit of measure description

API: findMotives

This API is used to find all the motives available for a shipping type. See index for available shipment types.

Table 4-57 API Basics

Endpoint URL	/motives/(shipmentType}
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Path Parameter	The shipment type to retrieve motives for (see Additional Data Definitions)
Output	Collection of carton sizes

Table 4-58 Output Data Definition

Attribute	Type	Definition
id	Long(18)	The unique identifier of the shipping motive
code	String(6)	The external code of the shipping motive
description	String(120)	A description of the motive
sequence	Long	The sequence in which the motive should appear in a list

Index

ID	Description
1	Parcel
2	Home Fleet
3	Other

ID	Description
1	Fulfillment Order
2	Store Transfer
3	Vendor Return

REST Service: Stock Count

The stock count services handle tasks related to a stock count.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/stockcounts

API Definitions

API	Description
Snapshot Count	Snapshots a stock count capturing the current stock on hand quantities.

API: Snapshot Stock Count

Executes a snapshot of the stock count capturing the current stock on hand quantity of each item on the count. The process of doing a snapshot first determines whether the stock count needs a snapshot and only snapshots those stock counts or stock count children that need a snapshot. If the stock count or stock count child does not need a snapshot, the service is considered successful.

API Basics

Endpoint URL	/{stockCountId}/snapshot
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Path Parameter Definitions

Attribute	Definition
stockCountId	The internal identifier of the stock count header.

REST Service: Store

This service integrates the store foundation data with an external application. Store integration is controlled by the MPS Work Type: DcsStore.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/stores

API Definitions

API	Description
Import Stores	Imports a collection of stores.
Delete Store	Deactivate a single store.
Read Store	Read store information based on an identifier (or link)
Find Stores	Lookup store information based on a collection of store identifiers.
Find Associated Stores	Lookup store associated to the specified input store.
Find Auto Receive Stores	Lookup stores that are allowed to auto receive from the specified input store.
Find Transfer Zone Stores	Lookup stores that are in the same transfer zone as the specified input store.
Find Adjustment Reason Codes	Finds reason codes available for inventory adjustments.
Find Shipment Reason Codes	Finds reason codes available for shipments.

API: Import Stores

Imports a collection of stores. This allows 1,000 stores per input call. All imported stores will be inventory-holding regular stores.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/import
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	List of Store to Import
Output	None
Max Response Limit	1000

Input Data Definition

Attribute	Data Type	Required	Description
stores	List of stores to import	Yes	A collection of up to 1000 stores to import.

Stores Data Definition

Attribute	Data Type	Required	Description
storeId	Long(10)	Yes	The store identifier
storeName	String(150)	Yes	The name of the store.
languageCode	String(3)	-	The language code of the store.
countryCode	String(3)	-	The country code of the store.
currencyCode	String(3)	-	The currency code of the store.
timezone	String(80)	-	The timezone of the store.
transferZoneId	String(128)	Yes	The transfer zone identifier of the store.
organizationUnitId	String(15)	-	The organization unit identifier of the store.
managedStore	Boolean	-	True indicates that EICS manages the inventory, false indicates it does not.
customerOrdering	Boolean	-	True indicates this store can take customer orders, false indicates it does not.

Example Output

{

"stores": [

{

"storeId": 5002,

"storeName": "Leamington Spa",

"languageCode": "EN",

"countryCode": "US",

"currencyCode": "USD",

"timezone": "America/Los_Angeles",

"transferZoneId": "1000",

"organizationUnitId": "1111",

"managedStore": true,

"allowsCustomerOrders": false

},

{

"storeId": 5003,

"storeName": "Leamington Spa",

"languageCode": "EN",

"countryCode": "US",

"currencyCode": "USD",

"timezone": "America/Los_Angeles",

"transferZoneId": "1000",

"organizationUnitId": "1111",

"managedStore": true,

"allowsCustomerOrders": false

}

]

}

API: Delete Store

Delete a store. Prior to placing the request to delete into the MPS queue, it validates that the store exists and that the store contains no items ranged to it.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/{storeId}/delete
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	None
Output	None
Max Response Limit	N/A

Path Parameter Definitions

Attribute	Definition
storeId	The internal identifier of the store.

API: Read Store

Retrieve information about a store based on a single unique store identifier or link.

API Basics

Endpoint URL	{base URL}/{storeId}
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	None

Output	Store
Max Response Limit	N/A

Path Parameter Definitions

Attribute	Definition
storeId	The internal identifier of the store.

Output Data Definition

Attribute	Data Type	Description
storeId	Long	The store identifier
storeName	String	The name of the store.
languageCode	String	The language code of the store
countryCode	String	The country code of the store.
currencyCode	String	The currency code of the store
timezone	String	The timezone of the store
transferZoneId	String	The transfer zone identifier of the store
organizationUnitId	String	The organization unit identifier of the store.
managedStore	boolean	True indicates that EICS manages the inventory, false indicates it does not.
customerOrdering	boolean	True indicates this store can take customer orders, false indicates it does not.

Example Output


```
{
   "links": [
      {
         "href": "/stores/5000",
         "rel": "self"
      },
      {
         "href": "/stores/5000/delete",
         "rel": "delete"
      }
   ],
   "storeId": 5000,
   "storeName": "Solihull",
   "languageCode": "EN",
```

"countryCode": "US",

"currencyCode": "USD",

"timezone": "America/Chicago",

"transferZoneId": "1000",

"organizationUnitId": "1111",

"managedStore": true,

"allowsCustomerOrders": false

}

API: Find Stores

Find stores based on a list of potential unique store identifiers. It allows a maximum of 1500 store identifiers.

API Basics

Endpoint URL	{base URL}/find
Method	POST
Successful Response	200 OK
Processing Type	Synchronous
Input	List of stores ids
Output	List of stores
Max Response Limit	1500

Input Data Definition

Attribute	Data Type	Required	Description
stores	List of stores Ids	Yes	A collection of up to 1500 stores to read.

Example Input

{

"storeIds": [

5000,

5001

]

}

Output Data Definition

Attribute	Data Type	Description
Stores	List of Stores	A collection containing the store information

Stores Data Definition

Attribute	Data Type	Required	Description
storeId	Long	Yes	The store identifier
storeName	String		The name of the store.
languageCode	String		The language code of the store
countryCode	String		The country code of the store.
currencyCode	String		The currency code of the store
timezone	String		The timezone of the store
transferZoneId	String		The transfer zone identifier of the store
organizationUnitId	String		The organization unit identifier of the store.
managedStore	boolean		True indicates that EICS manages the inventory, false indicates it does not.
customerOrdering	boolean		True indicates this store can take customer orders, false indicates it does not.

Example Output


```
[
   {
      "links": [
         {
            "href": "/stores/5000",
            "rel": "self"
         },
         {
            "href": "/stores/5000/delete",
            "rel": "delete"
         }
      ],
      "storeId": 5000,
      "storeName": "Solihull",
      "languageCode": "EN",
      "countryCode": "US",
      "currencyCode": "USD",
      "timezone": "America/Chicago",
      "transferZoneId": "1000",
```

```
        "organizationUnitId": "1111",
        "managedStore": true,
        "allowsCustomerOrders": false
    },
    {
        "links": [
            {
                "href": "/stores/5001",
                "rel": "self"
            },
            {
                "href": "/stores/5001/delete",
                "rel": "delete"
            }
        ],
        "storeId": 5001,
        "storeName": "Nottingham",
        "languageCode": "EN",
        "countryCode": "US",
        "currencyCode": "USD",
        "timezone": "America/New_York",
        "transferZoneId": "1000",
        "organizationUnitId": "1111",
        "managedStore": true,
        "allowsCustomerOrders": false
    }
]
```


API: Find Associated Stores

Find potential associated stores (buddy stores) to the specified input store.

API Basics

Endpoint URL	{base URL}/{storeId}/associated
Method	GET

Successful Response	200 OK
Processing Type	Synchronous
Input	None
Output	List of stores
Max Response Limit	N/A

Path Parameter Definitions

Attribute	Definition
storeId	The internal identifier of the store.

Output Data Definition

Attribute	Data Type	Required	Description
Stores	List of Stores Ids	Yes	A collection containing the store Ids

Stores Data Definition

Attribute	Data Type	Required	Description
storeId	Long	Yes	The internal identifier of the store.

Example Output


```
[
{
"links": [
{
"href": "/stores/5001",
"rel": "self"
},
{
"href": "/stores/5001/delete",
"rel": "delete"
}
],
"storeId": 5001
},
{
"links": [
{
```

"href": "/stores/5002",
"rel": "self"
},
{
"href": "/stores/5002/delete",
"rel": "delete"
}
],
"storeId": 5002
}
]

API: Find Auto Receive Stores

Find stores that are allowed to auto receive from the specified input store.

API Basics

Endpoint URL	{base URL}/{storeId}/autoreceive
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	None
Output	List of stores
Max Response Limit	N/A

Path Parameter Definitions

Attribute	Definition
storeId	The internal identifier of the store.

Example Output

[
{
"links": [
{
"href": "/stores/5001",
"rel": "self"
},
{

"href": "/stores/5001/delete",
"rel": "delete"
}
],
"storeId": 5001
},
{
"links": [
{
"href": "/stores/5002",
"rel": "self"
},
{
"href": "/stores/5002/delete",
"rel": "delete"
}
],
"storeId": 5002
}
]

API: Find Transfer Zone Stores

Find stores that are available within the transfer zone of the input store.

API Basics

Endpoint URL	{base URL}/{storeId}/transferzone
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	None
Output	List of stores
Max Response Limit	N/A

Path Parameter Definitions

Attribute	Definition
storeId	The internal identifier of the store.

Output Data Definition

Attribute	Data Type	Required	Description
Stores	List of Stores Ids	Yes	A collection containing the store Ids

Stores Data Definition

Attribute	Data Type	Required	Description
storeId	Long	Yes	The internal identifier of the store.

Example Output

[
{
"links": [
{
"href": "/stores/5001",
"rel": "self"
},
{
"href": "/stores/5001/delete",
"rel": "delete"
}
],
"storeId": 5001
},
{
"links": [
{
"href": "/stores/5002",
"rel": "self"
},
{
"href": "/stores/5002/delete",
"rel": "delete"
}
],

"storeId": 5002

}

]

REST Service: Store Item

This service integrates the store item foundation data with an external application. Store integration is controlled by the MPS Work Type: DcsItemLocation.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/storeitems

API Definitions

API	Description
Import Store Items	Imports items at a store location.
Remove Store Items	Deactivate items at a store location.
Import Replenishment Items	Imports replenishment item information.
Remove Replenishment Items	Deactivate replenishment item information.

API: Import Store Items

Imports store items into the system.

If more than 10,000 items are included in a single call, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/import
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	List of store Items to Import
Output	None
Max Response Limit	10,000

Input Data Definition

Attribute	Data Type	Required	Description

items	List of stores items to import	Yes	A collection of up to 10,000 stores items to import.

Stores Items Data Definition

Attribute	Data Type	Required	Description
itemId	String(25)	Yes	The unique item identifier (sku number).
shortDescription	String(255)	-	A short description of the item at this store.
longDescirption	String(400)	-	A long description of the item at this store.
status	String	Yes	See Index (Item Status)
primarySupplierId	Long(10)	-	The unique identifier of the primary supplier of the item to this store location.
storeControlPricing	Boolean	-	True indicates the item price can be controlled by the store.
rfid	Boolean	-	True indicates the item is RFID tagged.
defaultCurrencyCode	String(3)	-	The default currency of the item's price at this store.
purchaseType	Long	-	See Index (Purchase Type)
uinType	Integer	-	See Index (UIN Type)
uinCaptureTime	Integer	-	See Index (UIN Capture Time)
uinLabelCode	Long	-	The UIN label unique identifier.
uinExternalCreateAllowed	Boolean	-	True if an external system can create a UIN, false otherwise.

Example Input

{

"items": [

{

"itemId": "100637121",

"defaultCurrencyCode": "USB",

"longDescription": "TestDescriptionAA",

"primarySupplierId": 1,

"purchaseType": 1,

"rfid": true,

"shortDescription": "TestShortAA",

"status": 1,

"storeControlPricing": false,

"uinCaptureTime": 1,

"uinExternalCreateAllowed": true,

"uinLabelCode": "SN",

"uinType": 1

}

]

}

Additional Data Definitions

Item Status

Value	Definition
1	Active
2	Discontinued
3	Inactive
4	Auto Stockable

Purchase Type

Value	Definition
1	Normal Merchandise
2	Consignment Stock
3	Concession Items

UIN Capture Time

Value	Definition
1	Sale
2	Store Receiving

UIN Type

Value	Definition
1	Serial Number
2	Auto-Generated Serial Number

API: Remove Store Items

This will mark items as no longer usable. When all data is cleared out of transactions that reference this data, later batch jobs will eventually delete the material.

If more than 1000 items are included in a single call, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/{storeId}/remove

Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	None
Output	None
Max Input Limit	1000

Path Parameter Definitions

Attribute	Definition
storeId	The internal identifier of the store.

Input Data Definition

Attribute	Data Type	Required	Description
items	List of items to remove	Yes	A collection of up to 1000 items to remove.

Example Input

{

"itemIds": [

"100637121",

"100637113"

]

}

API: Import Replenishment Items

Imports item replenishment information for an item at a store location.

If more than 1000 items are included in a single call, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/{storeId}/replenish/import
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	None
Output	List of Items Replenishment to import
Max Input Limit	1000

Path Parameter Definitions

Attribute	Definition
storeId	The internal identifier of the store.

Input Data Definition

Attribute	Data Type	Required	Description
items	List of Item Replenishment Import	Yes	The item replenishment information to import.

Item Replenishment Import Data Definition

Attribute	Data Type	Required	Description
itemId	String (25)	Yes	The unique item identifier (sku number).
replenishmentMethod	String(6)		A code representing the replenishment method. (SO indicates Store Order).
rejectStoreOrder	Boolean		True indicates store orders must be on or after the next delivery date or should be rejected.
multipleDeliveryPerDayAllowed	Boolean		True indicates the item allows multiple deliveries per day at the location.
nextDeliveryDate	Date		The next delivery date of the time based on its replenishment type.

Example Input

{

"items": [

{

"itemId": "100637121",

"replenishmentMethod": "AB",

"rejectStoreOrder": false,

"multipleDeliveryPerDayAllowed": false,

"nextDeliveryDate": "2022-11-19T23:59:59-05:00"

}

]

}

API: Remove Replenishment Items

Clears the replenishment item information from within the store item setting the replenishment properties to empty, null, or default flag settings.

If more than 1000 items are included in a single call, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/replenish/remove
Method	POST
Successful Response	200 OK
Processing Type	Asynchronous
Input	List of item ids
Output	None
Max Input Limit	1000

Input Data Definition

Attribute	Data Type	Required	Description
Items	List of item Ids	Yes	A collection of up to 1000 items to read.

Example Input

{

"itemIds": [

"100637121",

"100637113"

]

REST Service: Store Order

This service allows query and approval of store orders.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/storeorders

APIs

API	Description
Find Orders	Finds store order header records based on search criteria
Read Order	Reads a store order
Approve Order	Approves the store order and notifies external system

Cancel Order	Cancels the store order and notifies external system

API: Read Order

API Basics

Endpoint URL	{base URL}/{storeOrderId}
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	N/A
Output	The store order
Max Response Limit	NA

Output Data Definition

Attribute	Type	Definition
storeOrderId	Long(12)	The unique identifier of the record.
storeId	Long(10)	The unique identifier of the store.
externalTransferId	String(128)	An external transfer identifier that this store order is associated to.
externalPurchaseOrderId	String(128)	An external purchase order identifier that this store order is associated to.
externalReference	String(128)	A reference to the store order in an external system.
parentReference	Long(12)	Reference to a parent transfer or order.
description	String(2000)	A description of the store order or cause of the store order.
status	Integer(2)	The status: See Index
origin	Integer(2)	The origin: See Index
requestedDeliveryDate	Date	The date that the store requests the delivery arrive by.
autoApproveDate	Date	The date the record was automatically approved in EICS.
addItemsAllowed	Boolean	Y indicates new items can be added to the store order.
replenishmentItemsOnly	Boolean	Y indicates only store order replenishment items can be added to the store order.
contextId	Long(18)	A context identifier associated to the store order.
deliverySlotId	Long(15)	The unique identifier of the delivery time.
productGroupId	Long(12)	The unique identifier of a product group associated to the order.
productGroupDescription	String(250)	The description of the product group associated to the order.
restrictToSupplierId	Long(12)	Allow only items on the store order that are associated to this supplier.

restrictToWarehouseId	Long(12)	Allow only items on the store order that are associated to this warehouse.
restrictToHierarchyId	Long(12)	Allow only items on the store order that belong to this merchandise hierarchy.
restrictToAreaId	Long(12)	Allow only items on the store order that are associated to this store sequence area.
createDate	Date	The date the record was created in EICS.
createUser	String(128)	The user that created the record in EICS.
externalCreateDate	Date	The date the store order was created in an external system.
updateDate	Date	The date the record was last updated in EICS.
approvedDate	Date	The date the record was approved in EICS.
approvedUser	String(128)	The user that approved the record in EICS.
lineItems	Collection	A collection of Store Order Line Items

Store Order Line Item

Column	Type	Definition
lineId	Long(12)	The unique identifier of the record.
itemId	String(25)	The unique identifier of the item.
approvedQuantity	BigDecimal(20,4)	The quantity of item ordered.
suggestedQuantity	BigDecimal(20,4)	The quantity of item ordered from the external system.
caseSize	BigDecimal(10,2)	The case size of the item ordered.
unitCostCurrency	String(3)	The currency of the unit cost.
unitCostAmount	BigDecimal(12,4)	The value of the unit cost.
deliverySlotId	Long(14)	The unique identifier of the delivery time.

API Find Orders

API is used to find lightweight transaction headers for store orders.

API Basics

Endpoint URL	{base URL}
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	Query Parameters
Output	List of store order headers
Max Response Limit	5,000

Query Parameters

Attribute	Type	Definition

storeId	Long(12)	Include only records for this store.
externalReference	String(128)	Include only records with this external transfer identifier.
externalTransferId	String(128)	Include only records with this external transfer identifier.
externalPurchaseOrderId	String(128)	Include only records with this external transfer identifier.
status	Integer	Include only records with this status. See StockOrderCriteriaStatus Index.
itemId	String(25)	Include only records with this item on them.
updateDateFrom	Date	Include records with a last update date on or after this date.
updateDateTo	Date	Include records with a first update date on or before this date.

Output Data Definition

Column	Type	Definition
storeOrderId	Long(12)	The unique identifier of the record.
storeId	Long(10)	The unique identifier of the store.
externalTransferId	String(128)	An external transfer identifier that this store order is associated to.
externalPurchaseOrderId	String(128)	An external purchase order identifier that this store order is associated to.
externalReference	String(128)	A reference to the store order in an external system.
description	String(2000)	A description of the store order or cause of the store order.
status	Integer(2)	The status: See Index
requestedDeliveryDate	Date	The date that the store requests the delivery arrive by.
updateDate	Date	The date the record was last updated in EICS.

API: Approve Order

This operation will approve the store order and send notifications and updates of the approval to an external system.

API Basics

Endpoint URL	{base URL}/{storeOrderId}/approve
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	N/A
Output	The store order
Max Response Limit	N/A

API: Cancel Order

API Basics

Endpoint URL	{base URL}/{storeOrderId}/cancel
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	N/A
Output	The store order
Max Response Limit	N/A

Additional Data Definitions

Store Order Status

ID	Status
1	New
2	In Progress
3	Approved
4	Canceled
5	Submitted

Store Order Criteria Status

ID	Status
1	New
2	In Progress
3	Approved
4	Canceled
5	Submitted
99	Active

Store Order Origin

ID	Status
1	External
2	Manual
3	System

REST Service: Store Sequencing

This service defines operations to manage Store Sequence information.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/sequences

APIs

API	Description
Find Sequence Areas	Finds sequence area header based on search criteria.
Read Sequence Area	Reads the details of a sequence area.
Create Sequence Area	Create a sequence area along with all its details.
Update Sequence Area	Replace the entire sequence area including all its details (a complete sequence area must be sent).
Delete Sequence Area	Delete a sequenced area and all its details.
Create Sequence Item	Create a single sequence item in a sequence area.
Update Sequence Item	Updates a single sequence item from a sequence area.
Delete Sequence Item	Removes a single sequence item from a sequence area.

API: Find Sequence Areas

This API is used to search for sequence area headers. No items or detailed information is returned by this API. At least one criteria is required in order to search.

API Basics

Table 4-59 API Basics

Endpoint URL	{base URL}
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	Query Parameters
Output	Collection of Sequence Areas
Max Response Limit	15,000

Query Parameters

Table 4-60 Query Parameters

Attribute	Type	Definition
storeId	Long(12)	Include only records for this store.
areaType	Integer(9)	Area Type (see Index).

Table 4-60 (Cont.) Query Parameters

departmentId	Long(12)	The unique identifier of a department associated to the area.
classId	Long(12)	The unique identifier of a class within the department associated to the area.
itemId	String(25)	Include only records with this item on them.

Output Data Definition

Table 4-61 Output Data Definition

Attribute	Type	Definition
sequenceAreaId	Long(10)	The unique identifier of the sequence area.
storeId	Long(10)	The unique identifier of the store.
description	String(255)	A description of this store sequence.
areaType	Integer(9)	Area Type (see Index).
departmentId	Long(12)	The unique identifier of a department associated to the area.
classId	Long(12)	The unique identifier of a class within the department associated to the area.
sequenceOrder	Long(20	The order of this store sequence compared to other store sequences at the store.
unsequenced	Boolean	True indicates that this is a default sequence area that contains all the items that have not been sequenced within a different area.

API: Read Sequence Area

This API is used to read an entire sequence area including its details.

API Basics

Table 4-62 API Basics

Endpoint URL	{base URL}/{sequenceAreaId}
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	N/A
Output	Sequence Area
Max Response Limit	N/A

Path Parameter Definition

Table 4-63 Path Parameter Definition

Attribute	Type	Definition
sequenceAreaId	Long(10)	The unique identifier of the sequence area.

Output Data Definition

Table 4-64 Output Data Definition

Attribute	Type	Definition
sequenceAreaId	Long(10)	The unique identifier of the sequence area.
storeId	Long(10)	The unique identifier of the store.
description	String(255)	A description of this store sequence.
areaType	Integer(9)	Area Type (see Index).
departmentId	Long(12)	The unique identifier of a department associated to the area.
classId	Long(12)	The unique identifier of a class within the department associated to the area.
sequenceOrder	Long(20	The order of this store sequence compared to other store sequences at the store.
unsequenced	Boolean	True indicates that this is a default sequence area that contains all the items that have not been sequenced within a different area.
Items	Collection	Contains all the items in the area (see Store Sequence Item)

Store Sequence Item

Table 4-65 Store Sequence Item

Payload	Type	Definition
sequenceItemId	Long(12)	The unique area item identifier.
itemId	String(25)	The unique identifier of the item
sequenceOrder	Long(20)	The order of the item within its sequence.
capacity	BigDecimal(11,2)	The amount of the item that can be contained in the area for that item at the unit of measure.
width	Long(12)	The number of items that can fit across the width of the shelf.
uomMode	Integer(2)	The mode of display of the unit of measure of the item.
primaryLocation	Boolean	Y indicates this is the primary sequence for the item.
ticketQuantity	BigDecimal(11,2)	The quantity of tickets that need to be printed for the item inventory area.
ticketFormatId	Long(10)	The unique identifier of the ticket format used to print the tickets.

API: Create Sequence Area

API Basics

Endpoint URL	{base URL}
Method	POST
Successful Response	200 OK
Processing Type	Synchronous
Input	Sequence Area Create Info
Output	Sequence Area Reference Info
Max Response Limit	1,000 items in the area

Input Data Definition

Attribute	Type	Req	Definition
storeId	Long(10)	X	The unique identifier of the store.
description	String(255)		A description of this store sequence.
areaType	Integer(9)	X	Area Type (see Index).
departmentId	Long(12)		The unique identifier of a department associated to the area.
classId	Long(12)		The unique identifier of a class within the department associated to the area.
sequenceOrder	Long(20)	X	The order of this store sequence compared to other store sequences at the store. The sequence order must be unique for the set of areas within the store.
items	Collections	X	All the items that belong to this sequence area (see Sequence Area Item Create)

Store Sequence Area Item Create

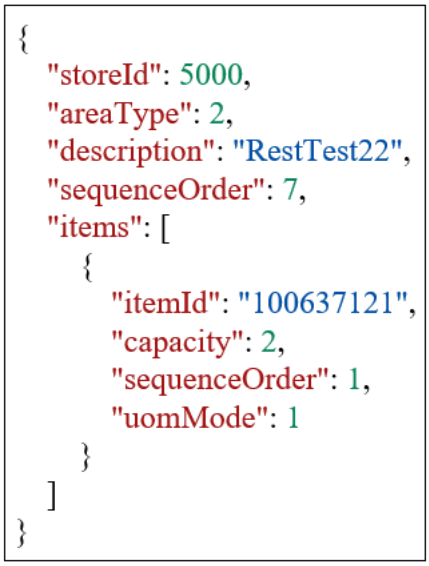
Payload	Type	Req	Definition
itemId	String(25)	X	The unique identifier of the item
sequenceOrder	Long(20)	X	The order of the item within its sequence.
capacity	BigDecimal(11,2)	X	The amount of the item that can be contained in the area for that item at the unit of measure.
width	Long(12)		The number of items that can fit across the width of the shelf.
uomMode	Integer(2)	X	The mode of display of the unit of measure of the item.
primaryLocation	Boolean		Y indicates this is the primary sequence for the item.
ticketQuantity	BigDecimal(11,2)		The quantity of tickets that need to be printed for the item inventory area.

ticketFormatId	Long(10)	The unique identifier of the ticket format used to print the tickets.

Output Data Definition

Attribute	Type	Definition
sequenceAreaId	Long(12)	The newly created adjustment identifier.
storeId	Long(10)	The store identifier.

Example



```
{
  "storeId": 5000,
  "areaType": 2,
  "description": "RestTest22",
  "sequenceOrder": 7,
  "items": [
    {
      "itemId": "100637121",
      "capacity": 2,
      "sequenceOrder": 1,
      "uomMode": 1
    }
  ]
}
```

API: Update Sequence Area

This API is used to update a sequence area within a store.

This API will **replace** the entire sequence area with the new data passed to the API. Only the area identifier and store identifier will be preserved from previous data.

API Basics

Table 4-66 API Basics

Endpoint URL	{base URL}
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	Sequence Area Update Info
Output	N/A
Max Response Limit	1,000 items in the area

Path Parameter Definition

Table 4-67 Path Parameter Definition

Attribute	Type	Definition
sequenceItemId	Long(12)	The unique area item identifier

Input Data Definition

Table 4-68 Input Data Definition

Attribute	Type	Req	Definition
description	String(255)		A description of this store sequence.
areaType	Integer(9)	X	Area Type (see Index).
departmentId	Long(12)		The unique identifier of a department associated to the area.
classId	Long(12)		The unique identifier of a class within the department associated to the area.
sequenceOrder	Long(20)	X	The order of this store sequence compared to other store sequences at the store. The sequence order must be unique for the set of areas within the store.
items	Collections	X	All the items that belong to this sequence area (see Sequence Area Item Update)

Store Sequence Area Item Update

Table 4-69 Store Sequence Area Item Update

Payload	Type	Req	Definition
itemId	String(25)	X	The unique identifier of the item
sequenceOrder	Long(20)	X	The order of the item within its sequence.
capacity	BigDecimal(11,2)	X	The amount of the item that can be contained in the area for that item at the unit of measure.
width	Long(12)		The number of items that can fit across the width of the shelf.
uomMode	Integer(2)	X	The mode of display of the unit of measure of the item.
primaryLocation	Boolean		Y indicates this is the primary sequence for the item.
ticketQuantity	BigDecimal(11,2)		The quantity of tickets that need to be printed for the item inventory area.
ticketFormatId	Long(10)		The unique identifier of the ticket format used to print the tickets.

Example

```
{
  "areaType": 2,
  "description": "RestTest22",
  "sequenceOrder": 7,
  "items": [
    {
      "itemId": "100637121",
      "capacity": 2,
      "sequenceOrder": 1,
      "uomMode": 1
    }
  ]
}
```


API: Delete Sequence Area

This API is used to delete a sequence area and all its items. This will not delete a sequence area currently being used by a sequenced stock count.

API Basics

Table 4-70 API Basics

Endpoint URL	{base URL}
Method	DELETE
Successful Response	204 No Content
Processing Type	Synchronous
Input	N/A
Output	N/A
Max Response Limit	N/A

API: Create Sequence Item

This API is used to create a new sequence item within a sequence area.

API Basics

Table 4-71 API Basics

Endpoint URL	{base URL}/{sequenceAreaId}/items
Method	POST
Successful Response	200 No Content
Processing Type	Synchronous
Input	Store Sequence Item Create

Table 4-71 (Cont.) API Basics

Output	Store Sequence Item
Max Response Limit	N/A

Input Data Definition

Table 4-72 Input Data Definition

Payload	Type	Req	Definition
itemId	String(25)	X	The unique identifier of the item
sequenceOrder	Long(20)	X	The order of the item within its sequence.
capacity	BigDecimal(11,2)	X	The amount of the item that can be contained in the area for that item at the unit of measure.
width	Long(12)		The number of items that can fit across the width of the shelf.
uomMode	Integer(2)	X	The mode of display of the unit of measure of the item. (see Index)
primaryLocation	Boolean		Y indicates this is the primary sequence for the item.
ticketQuantity	BigDecimal(11,2)		The quantity of tickets that need to be printed for the item inventory area.
ticketFormatId	Long(10)		The unique identifier of the ticket format used to print the tickets.

Example

```
{
  "itemId": "100637113",
  "capacity": 2,
  "sequenceOrder": 1,
  "uomMode": 1
}
```

API: Update Sequence Item

This API is used to update a sequence item within a sequence area.

API Basics

Table 4-73 API Basics

Endpoint URL	{base URL}/{sequenceAreaId}/items/{itemId}
Method	POST

Table 4-73 (Cont.) API Basics

Successful Response	204 No Content
Processing Type	Synchronous
Input	N/A
Output	N/A
Max Response Limit	N/A

Path Parameter Definition

Table 4-74 Path Parameter Definition

Attribute	Type	Definition
sequencerAreaId	Long(12)	The unique sequence area identifier.
itemId	String(25)	The item to be deleted from the sequence.

Input Data Definition

Table 4-75 Input Data Definition

Payload	Type	Req	Definition
sequenceOrder	Long(20)	X	The order of the item within its sequence.
capacity	BigDecimal(11,2)	X	The amount of the item that can be contained in the area for that item at the unit of measure.
width	Long(12)		The number of items that can fit across the width of the shelf.
uomMode	Integer(2)	X	The mode of display of the unit of measure of the item.
primaryLocation	Boolean		Y indicates this is the primary sequence for the item.
ticketQuantity	BigDecimal(11,2)		The quantity of tickets that need to be printed for the item inventory area.
ticketFormatId	Long(10)		The unique identifier of the ticket format used to print the tickets.

Example

```
{
  "capacity": 8,
  "sequenceOrder": 2,
  "uomMode": 1
}
```

API: Delete Sequence Item

This API is used to delete a sequence item from an area.

API Basics

Table 4-76 API Basics

Endpoint URL	{base URL}/{sequenceAreaId}/items/{itemId}
Method	DELETE
Successful Response	204 No Content
Processing Type	Synchronous
Input	N/A
Output	N/A
Max Response Limit	N/A

Path Parameter Definition

Table 4-77 Path Parameter Definition

Attribute	Type	Definition
sequencerAreaId	Long(12)	The unique sequence area identifier.
itemId	String(25)	The item to be deleted from the sequence.

Additional Data Definitions

Sequence Area Type

Table 4-78 Sequence Area Type

ID	Status
1	Shopfloor
2	Backroom
3	None

Sequence UOM Mode

Table 4-79 Sequence UOM Mode

ID	Status
1	Units
2	Cases

REST Service: Supplier

This service integrates supplier and supplier item foundation data.

Asynchronous supplier integration is processed through staged messages and is controlled by the MPS Work Type: DcsSupplier.

Asynchronous supplier item integration is processed through staged messages and is controlled by the MPS Work Type: DcsSupplierItem.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/suppliers

API Definitions

API	Description
Import Suppliers	Imports supplier.
Delete Supplier	Deletes a supplier.
Import Items	Imports items for a supplier.
Delete Items	Deletes items from a supplier.
Import Item UOMs	Imports units of measure for a supplier item.
Delete Item UOMs	Deletes units of measure from a supplier item.
Import Item Countries	Imports countries for a supplier item.
Delete Item Countries	Deletes countries from a supplier item.
Import Item Dimensions	Imports items dimensions for a supplier item country.
Delete Item Dimensions	Deletes dimensions from a supplier item country.
Import Item Manufacturers	Imports manufacturers for a supplier item country.
Delete Item Manufacturers	Deletes manufacturers from a supplier item country.

API: Import Suppliers

Imports a list of suppliers.

If the import contains more than 500 suppliers, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/import
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous

Input	List of suppliers to Import
Output	None
Max Response Limit	500

Input Data Definition

Attribute	Data Type	Required	Description
suppliers	List of details	Yes	A collection of up to 500 suppliers to import.

Detail Data Definition

Attribute	Data Type	Required	Description
supplierId	Long (10)	Yes	The supplier identifier.
name	String (240)	-	The supplier's name.
status	Integer	Yes	The supplier’s status: See Supplier Status
dunsNumber	String (9)	-	The nine-digit number assigned and maintained by Dun and Bradstreet.
taxId	String (18)	-	The tax identification number of the supplier.
parentId	Long (128)	-	The parent supplier's identifier.
countryCode	String (3)	-	The 2-3 character ISO code of the country.
languageCode	String (3)	-	The 2-3 character ISO code of the language.
currencyCode	String (3)	-	The 2-3 character ISO code of the currency.
returnAllowed	Boolean	-	True is return is allowed to the supplier, N otherwise.
authorizationRequired	Boolean	-	True indicates an authorization number is required when merchandise is return to this supplier.
orderCreateAllowed	Boolean	-	True indicates at a purchase order can be created for the supplier when processing deliveries from that supplier.
vendorCheck	Boolean	-	True indicates that orders from this supplier requires vendor control.
vendorCheckPercent	BigDecimal	-	Indicates the percentage of items per receipt that will be marked for vendor checking.
quantityLevel	Integer	Yes	The Quantity Level: See Quantity Level
deliveryDiscrepancy	Integer	Yes	The delivery discrepancy: See Supplier Delivery Discrepancy Type
organizationUnitIds	List<Long>	Yes	A complete list of organization unit identifiers for the supplier.

Example Input

```
{
"suppliers": [
```

{
"supplierId": 5000,
"status": 1,
"quantityLevel": 1,
"deliveryDiscrepancy": 1,
"organizationUnitIds": [1]
},
{
"supplierId": 5001,
"status": 1,
"returnAllowed": false,
"quantityLevel": 1,
"deliveryDiscrepancy": 1,
"organizationUnitIds": [1]
}
]
}

Additional Data Definitions

Supplier Delivery Discrepancy Type

Value	Definition
1	Allow Any Discrepancy
2	Allow Overage But Not Short Receipts
3	Discrepancy Not Allowed

Supplier Status

Value	Definition
1	Active
2	Inactive

Quantity Level

Value	Definition
1	Eaches
2	Cases

API: Delete Supplier

Deletes a supplier.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/{supplierId}/delete
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	None
Output	None
Max Response Limit	1000

Path Parameter Definitions

Attribute	Definition
supplierId	The internal identifier of the supplier.

API: Import Items

Imports supplier items. Later during asynchronous processing, it validates that both the supplier and item exist before inserting new records.

If the import contains more than 500 supplier items, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/items
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	List of Supplier Items to import
Output	None
Max Response Limit	500

Path Parameter Definitions

Attribute	Definition
supplierId	The internal identifier of the supplier.

Input Data Definition

Attribute	Data Type	Required	Description

items	List of details	Yes	The supplier item information to import.

Detail Data Definition

Attribute	Data Type	Required	Description
supplierId	Long (10)	Yes	The supplier identifier.
itemId	String (25)	Yes	The item identifier.
vendorProductNumber	String (256)	-	Vendor product number.
primary	Boolean	-	True indicates this supplier is the primary supplier for the item at all locations.

Example Input

{
"items": [
{
"supplierId": 5000,
"itemId": "163715121",
"vendorProductNumber": "abc"
},
{
"supplierId": 5001,
"itemId": "163715121",
"vendorProductNumber": "def"
}
]
}

API: Delete Items

Deletes supplier items.

If the delete contains more than 500 supplier items, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/items/delete
Method	POST
Successful Response	202 Accepted

Processing Type	Asynchronous
Input	List of supplier items
Output	None
Max Response Limit	500

Input Data Definition

Attribute	Data Type	Required	Description
items	List of details	Yes	The supplier item information to delete.

Detail Data Definition

Attribute	Data Type	Required	Description
supplierId	Long (10)	Yes	The supplier identifier.
itemId	String (15)	Yes	The item identifier.

Example Input

```
{
"items":
[
{
"supplierId":"9002",
"itemId": "100637130"
},
{
"supplierId":"9002",
"itemId": "100637148"
}
]
}
```

API: Import Item UOMs

Imports supplier item unit of measure information.

If the import contains more than 500 supplier item units of measure, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/uoms
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	List of supplier items UOMs to import
Output	None
Max Response Limit	500

Input Data Definition

Attribute	Data Type	Required	Description
uoms	List of details	Yes	The UOMs to import.

Detail Data Definition

Attribute	Data Type	Required	Description
supplierId	Long (10)	Yes	The supplier identifier.
itemId	String (25)	Yes	The item identifier.
unitOfMeasure	String (4)	Yes	The unit of measure.
Value	BigDecimal	Yes	Equivalent item/supplier shipping carton value in unit of measure

Example Input

{

"uoms": [

{

"supplierId": 5000,

"itemId": "163715121",

"unitOfMeasure": "KG",

"value": "1.0"

},

{

"supplierId": 5001,

"itemId": "163715121",

"unitOfMeasure": "KG",

"value": "1.0"

}

]

}

API: Delete Item UOMs

Deletes supplier item unit of measure information.

If the delete contains more than 500 supplier item units of measure, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/uoms/delete
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	List of supplier items UOMs to delete
Output	None
Max Response Limit	500

Input Data Definition

Attribute	Data Type	Required	Description
uoms	List of details	Yes	The UOMs to delete.

Detail Data Definition

Attribute	Data Type	Required	Description
supplierId	Long (10)	Yes	The supplier identifier.
itemId	String (25)	Yes	The item identifier.
unitOfMeasure	String (4)	Yes	The unit of measure.

Example Input

{

"uoms":

[

{

"supplierId":"9002",

"itemId": "100637130",

"unitOfMeasure":"EA"

},

{

"supplierId":"9002",

"itemId": "100637148",

"unitOfMeasure":"EA"

}

]

API: Import Item Countries

Imports supplier item country information. If the imported supplier item country is for the primary supplier of the item, the item's case size will be updated on the item master to reflect the imported case size.

If the import contains more than 500 supplier item countries, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/countries
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	List of supplier items country to import
Output	None
Max Response Limit	500

Input Data Definition

Attribute	Data Type	Required	Description
countries	List of details	Yes	The countries to import.

Detail Data Definition

Attribute	Data Type	Required	Description
supplierId	Long (10)	Yes	The supplier identifier.
itemId	String (25)	Yes	The item identifier.
countryCode	String (3)	Yes	The ISO country code of the supplier that produces the item.
caseSize	BigDecimal	Yes	Number of items with a case from the supplier.
unitCostCurrency	String (3)	-	The unit cost currency for that item and supplier in that country.
unitCostValue	BigDecimal	-	The unit cost of the item and supplier in that country.

Example Input

{

```
  "countries": [
    {
      "supplierId": 5100,
      "itemId": "163715121",
      "countryCode": "US",
      "caseSize":7
    },
    {
      "supplierId": 5200,
      "itemId": "163715121",
      "countryCode": "US",
      "caseSize":8
    }
  ]
}
```


API: Delete Item Countries

Deletes supplier item country information.

If the delete contains more than 500 supplier item countries, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/countries/delete
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	List of supplier items country to remove
Output	None
Max Response Limit	500

Input Data Definition

Attribute	Data Type	Required	Description
countries	List of details	Yes	The countries to remove.

Detail Data Definition

Attribute	Data Type	Required	Description
supplierId	Long (10)	Yes	The supplier identifier.
itemId	String (25)	Yes	The item identifier.
countryCode	String (3)	Yes	The ISO country code of the supplier that produces the item.

Example Input

{

"countries":

[

{

"supplierId":"9002",

"itemId": "100637130",

"countryCode":"US"

}

]

}

API: Import Item Dimensions

Imports supplier item country dimension information.

If the import contains more than 500 supplier item dimensions, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/dimensions
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	List of supplier items dimensions to import
Output	None
Max Response Limit	500

Input Data Definition

Attribute	Data Type	Required	Description
dimensions	List of details	Yes	The dimensions to import.

Detail Data Definition

Attribute	Data Type	Required	Description
supplierId	Long (10)	Yes	The supplier identifier.
itemId	String (25)	Yes	The item identifier.
countryCode	String (3)	Yes	The ISO country code of the supplier that produces the item.
dimensionName	String (6)	Yes	Dimension name
presentationMethod	String (6)	-	Describes the packaging method
Length	BigDecimal	-	The length in dimension unit of measure.
Width	BigDecimal	-	The width in dimension unit of measure.
Height	BigDecimal	-	The height in dimension unit of measure.
dimensionUom	String (4)	-	The unit of measure of the dimensions.
Weight	BigDecimal	-	The weight of the object in weight unit of measure.
netWeight	BigDecimal	-	The net weight of the goods without packaging in weight unit of measure.
weightUom	String (4)	-	The weight unit of measure.
liquidVolume	BigDecimal	-	The liquid value or capacity of the object.
liquidVolumeUom	String (4)	-	The liquid volume unit of measure.
statisticalCube	BigDecimal	-	A statistical value of the object's dimensions used for shipment loading purposes.

Example Input

{

"dimensions": [

{

"supplierId": 5100,

"itemId": "163715121",

"countryCode": "US",

"dimensionName": "Hello"

},

{

"supplierId": 7100,

"itemId": "163715121",

"countryCode": "US",

"dimensionName": "Bye"

}

]

}

API: Delete Item Dimensions

Deletes supplier item country dimension information.

If the delete contains more than 500 supplier items, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/dimensions/delete
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	List of supplier items dimensions to remove
Output	None
Max Response Limit	500

Input Data Definition

Attribute	Data Type	Required	Description
Dimensions	List of details	Yes	The dimensions to remove.

Supplier Items Dimensions Data Definition

Attribute	Data Type	Required	Description
supplierId	Long (10)	Yes	The supplier identifier.
itemId	String (25)	Yes	The item identifier.
countryCode	String (3)	Yes	The ISO country code of the supplier that produces the item
dimensionName	String (6)	Yes	Dimension name

Example Input

{

"dimensions":

[

{

"supplierId":"9002",

"itemId": "100637130",

"countryCode":"US",

"dimensionName":"Dimen1"

},

{

"supplierId":"9002",

"itemId": "100637148",

"countryCode":"US",

"dimensionName":"Dimen2"

}

]

}

API: Import Item Manufacturers

Import supplier item country manufacturer information.

If the import contains more than 500 supplier item manufacturers, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/manufacturers
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	List of supplier items manufacturer to import
Output	None
Max Response Limit	500

Input Data Definition

Attribute	Data Type	Required	Description
manufacturers	List of details	Yes	The manufacturers to import.

Detail Data Definition

Attribute	Data Type	Required	Description
supplierId	Long (10)	Yes	The supplier identifier.
itemId	String (25)	Yes	The item identifier.
countryCode	String (3)	Yes	The ISO country code of the supplier that produces the item.
primary	Boolean	-	True indicates it is primary country of manufacture.

Example Input

{
"manufacturers": [
{
"supplierId": 5100,
"itemId": "163715121",
"countryCode": "US",
"primary": true
},
{
"supplierId": 7100,
"itemId": "163715121",
"countryCode": "US",
"primary": true
}
]
}

API: Delete Item Manufacturers

Deletes supplier item country manufacturer information.

If the delete contains more than 500 supplier item manufacturers, an input too large error will be returned.

A "Forbidden" response will occur if application is integrated with MFCS or RMS.

API Basics

Endpoint URL	{base URL}/manufacturers/delete
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	List of supplier items manufacturers to remove
Output	None
Max Response Limit	500

Input Data Definition

Attribute	Data Type	Required	Description

manufacturers	List of details	Yes	The manufacturers to remove.

Detail Data Definition

Attribute	Data Type	Required	Description
supplierId	Long (10)	Yes	The supplier identifier.
itemId	String (25)	Yes	The item identifier.
countryCode	String (3)	Yes	The ISO country code of the supplier that produces the item.

Example Input

{
"manufacturers":
[
{
"supplierId":"9002",
"itemId": "100637130",
"countryCode":"US"
},
{
"supplierId":"9002",
"itemId": "100637148",
"countryCode":"US"
}
]
}

REST Service: Ticket

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/tickets

API Definitions

API	Description
readTicket	This API reads a current actively ticket by its identifier.
findTickets	API is used to find summarized ticket headers for a set of criteria.
readArchivedTicket	This API reads a previously printed and archived ticket by its identifier.
findArchivedTickets	API is used to find archived ticket summaries.
createTickets	Creates new tickets within the system.
updateTickets	Updates current tickets within the system.
printTickets	Prints the requested tickets.
findTicketFormats	Reads all the available ticket formats.
findTicketPrinters	Finds the printers available to print tickets.

API: readTicket

This API reads a current actively ticket by its identifier.

API Basics

Endpoint URL	{base URL}/{ticketId}
Method	GET
Success Response	200 OK
Processing Type	Synchronous
Input	The ticket identifier
Output	The ticket

Output Data Definition

Column	Type	Definition
ticketId	Long(12)	The unique ticket identifier.
itemId	String(25)	The item identifier.
storeId	Long(10)	The store identifier.
originType	Integer(2)	The origin type (See Index)
ticketFormatId	Long(12)	The identifier of the ticket format used for printing.
ticketFormatDescription	String(100)	The description of the ticket format associated to the ticket.
ticketFormatType	Integer(4)	The type of the ticket format associated to the ticket.
ticketFormatReference	String(255)	The format reference used to print the ticket.
ticketFormatZplId	Long(12)	The ZPL format file identifier.
ticketCount	Integer(3)	The number of instances of this ticket to print.
ticketSequence	Integer(3)	The sequence number of a ticket within a ticket grouping.
printQuantity	BigDecimal(12,4)	The quantity to be printed on the ticket.
printDate	Date	The date the ticket should be printed.

groupId	Long(12)	An internal EICS grouping identifier that groups the tickets.
groupIdExternal	String(128)	An external system grouping identifier that groups the tickets.
autoPrint	Boolean	True if the ticket should automatically print, false if the ticket requires manual printing.
autoRefeshQuantity	Boolean	True indicates the ticket count gets refreshed with SOH at the time of printing, false it prints as is.
countryOfManufacture	String(3)	A two letter country code denoting the country of manufacture for the item.
shortDescription	String(255)	The short description of the item.
longDescription	String(400)	The long description of the item.
shortDescriptionLanguage	String(255)	The short description of the item in the language of the store.
longDescriptionLanguage	String(400)	The long description of the item in the language of the store.
differentiatorType1	String(10)	The description of the differentiator type for differentiator 1.
differentiatorType2	String(10)	The description of the differentiator type for differentiator 2.
differentiatorType3	String(10)	The description of the differentiator type for differentiator 3.
differentiatorType4	String(10)	The description of the differentiator type for differentiator 4.
differentiatorDescription1	String(255)	The description of differentiator 1 of the item.
differentiatorDescription2	String(255)	The description of differentiator 2 of the item.
differentiatorDescription3	String(255)	The description of differentiator 3 of the item.
differentiatorDescription4	String(255)	The description of differentiator 4 of the item.
departmentId	Long(12)	The department identifier of the item.
departmentName	String(360)	The department name of the item.
classId	Long(12)	The class identifier of the item.
className	String(360)	The class name of the item.
subclassId	Long(12)	The subclass identifier of the item.
subclassName	String(360)	The subclass name of the item.
primaryUpc	String(25)	The primary Unique Product Code for the item.
priceCurrency	String(3)	The currency code of the ticket price.
priceAmount	BigDecimal(12,4)	The amount of the ticket price.
priceType	Integer(3)	The type of the ticket price. (See Index)
priceUom	String(4)	The unit of measure of the ticket price.
priceActiveDate	Date	The date the ticket price became active.
priceExpireDate	Date	The date the ticket price expires.
mutliUnitPriceCurrency	String(3)	The currency code of the ticket's multi-unit price.
multiUnitPriceAmount	BigDecimal(12,4)	The amount of the ticket's multi-unit price.
multiUnitPriceUom	String(4)	The unit of measure of the ticket's multi-unit price.

multiUnitQuantity	BigDecimal(12,4)	The multi-unit quantity associated to the price.
overridePriceCurrency	String(3)	The override price currency code.
overridePriceAmount	BigDecimal(20,4)	The override price amount.
previousPriceCurrency	String(3)	The currency code of the previous price.
previousPriceAmount	BigDecimal(12,4)	The amount of the previous price.
previousPriceType	Integer(3)	The price type of the previous price.
lowestMonthlyPriceCurrency	String(3)	The currency code of the lowest monthly price.
lowestMonthlyPriceAmount	BigDecimal(12,4)	The amount of the lowest monthly price.
lowestMonthlyPriceType	Integer(3)	The price type of the lowest monthly price.
printedDate	Date	The date that the ticket was printed.
printedUser	String(128)	The user that printed the ticket.
createDate	Date	The date the ticket was created.
createUser	String(128)	The user that created the ticket.
updateDate	Date	The date the ticket was last updated.
updateUser	String(128)	The user that last updated the ticket.
udas	Collection	A group of associated user defined attributes.

UDA

Column	Type	Definition
name	String	The name of the user defined attribute.
value	String	The value of the user defined attributes.

API: findTickets

API is used to find summarized ticket headers for a set of criteria.

If maximum results are exceeded, additional or more limiting input criteria will be required.

API Basics

Endpoint URL	{base URL}
Method	GET
Success Response	200 OK
Processing Type	Synchronous
Input	Query Parameters
Output	List of Ticket Headers
Maximum Results Allowed	5,000

Input Data Definition

Attribute	Type	Definition
itemId	String(25)	Include only records with this item identifier.
storeId	Long(10)	Include only records with this store identifier.
ticketFormatId	Long(12)	Include only records with this ticket format identifier.
groupId	Long(12)	Include only records with this EICS group identifier.
groupIdExternal	String(128)	Include only records with this external system group identifier.
printDateFrom	String	Include only records on or after this scheduled print date and time.
printDateTo	String	Include only records on or before this scheduled print date and time.
updateDateFrom	String	Include only records on or after this scheduled print date and time.
updateDateTo	String	Include only records on or before this scheduled print date and time.

Output Data Definition

Column	Type	Definition
ticketId	Long(12)	The unique ticket identifier.
itemId	String(25)	The item identifier.
storeId	Long(10)	The store identifier.
originType	Integer(2)	The origin type (See Index)
ticketFormatId	Long(12)	The identifier of the ticket format used for printing.
ticketSequence	Integer(3)	The sequence number of a ticket within a ticket grouping.
groupId	Long(12)	An internal EICS grouping identifier that groups the tickets.
groupIdExternal	String(128)	An external system grouping identifier that groups the tickets.
autoPrint	Boolean	True if the ticket should automatically print, false if the ticket requires manual printing.
printDate	Date	The date the ticket should be printed.
printQuantity	BigDecimal(12, 4)	The quantity to be printed on the ticket.
updateDate	Date	The date the ticket was last updated.

API: readArchivedTicket

This API reads a previously printed and archived ticket by its identifier.

API Basics

Endpoint URL	{base URL}/archives/{ticketId}
Method	GET

Success Response	200 OK
Processing Type	Synchronous
Input	The ticket identifier
Output	The ticket details

Output Data Definition

See readTicket() for the output data definition of this API.

API: findArchivedTickets

API is used to find archived ticket summaries.

If the number of tickets found exceeds the limit, additional or more limiting input criteria will be required.

API Basics

Endpoint URL	{base URL}/archives
Method	GET
Success Response	200 OK
Processing Type	Synchronous
Input	Query Parameters
Output	List of Ticket Headers
Maximum Results Allowed	5,000

Input Data Definition

Attribute	Type	Definition
itemId	String(25)	Include only records with this item identifier.
storeId	Long(10)	Include only records with this store identifier.
ticketFormatId	Long(12)	Include only records with this ticket format identifier.
groupId	Long(12)	Include only records with this EICS group identifier.
groupIdExternal	String(128)	Include only records with this external system group identifier.
printedDateFrom	String	Include only records that were printed on or after this date and time.
printedDateTo	String	Include only records that were printed on or before this date and time.

Output Data Definition

Column	Type	Definition
ticketId	Long(12)	The unique ticket identifier.
itemId	String(25)	The item identifier.
storeId	Long(10)	The store identifier.

originType	Integer(2)	The origin type (See Index)
ticketFormatId	Long(12)	The identifier of the ticket format used for printing.
ticketSequence	Integer(3)	The sequence number of a ticket within a ticket grouping.
groupId	Long(12)	An internal EICS grouping identifier that groups the tickets.
groupIdExternal	String(128)	An external system grouping identifier that groups the tickets.
printedDate	Date	The date the ticket was printed.
printQuantity	BigDecimal(12,4)	The quantity printed on the ticket.
priceCurrency	String(3)	The currency of the price of the ticket.
priceAmount	BigDecimal(12,4)	The amount of the price of the ticket.

API: createTickets

Creates new tickets within the system.

API Basics

Endpoint URL	{base URL}
Method	POST
Success Response	200 OK
Processing Type	Synchronous
Input	Ticket group
Output	List of ticket identifying information
Maximum Input Allowed	2,000 tickets within the group

Input Data Definition

Payload	Type	Req	Definition
storeId	Long(10)	X	The store identifier.
groupIdExternal	String(128)		An external system grouping identifier that groups the tickets.
printDate	Date	X	The date the ticket should be printed.
autoPrint	Boolean		True if the ticket should automatically print, false if the ticket requires manual printing. Defaults to false.
autoRefeshQuantity	Boolean		True indicates the ticket count gets refreshed with SOH at the time of printing, false it prints as is. Defaults to false.
tickets	Collection	X	The tickets to create.

Ticket

Payload	Type	Req	Definition
itemId	String(25)	X	The item identifier.

originType	Integer(2)	X	The origin type (See Index)
ticketFormatId	Long(12)	X	The identifier of the ticket format used for printing.
ticketCount	Integer(3)	X	The number of instances of this ticket to print.
ticketSequence	Integer(3)	X	The sequence number of a ticket within a ticket grouping.
printQuantity	BigDecimal(12, 4)	X	The quantity to be printed on the ticket.
overridePriceCurrency	String(3)		The override price currency code. Required if an amount is entered.
overridePriceAmount	BigDecimal(20, 4)		The override price amount.
countryOfManufacture	String(3)		A two letter country code denoting the country of manufacture for the item.

Output Data Definition

Payload	Type	Definition
ticketId	Long(12)	The unique ticket identifier.
storeId	Long(10)	The store identifier.
ticketFormatId	Long(12)	The identifier of the ticket format used for printing.

Example Input

```
{
   "storeId": 5000,
   "groupIdExternal": "123456",
   "printDate": "2023-01-10T23:59:59-05:00",
   "autoPrint": false,
   "autoRefreshQuantity": false,
   "tickets": [
      {
         "itemId": "100637121",
         "originType": 1,
         "ticketFormatId": 1,
         "ticketCount": 2,
         "ticketSequence": 3
      },
      {
         "itemId": "100637113",
```

```
            "originType": 2,
            "ticketFormatId": 2,
            "ticketCount": 4,
            "ticketSequence": 5
        }
    ]
}
```


API: Update Tickets

Updates current tickets within the system.

If a field that is not required contains an empty or null value, the ticket will be updated to that empty or null value.

API Basics

Endpoint URL	{base URL}/update
Method	POST
Success Response	204 No Content
Processing Type	Synchronous
Processing Type	Synchronous
Input	Tickets
Output	N/A
Maximum Input Allowed	2,000 tickets

Input Data Definition

Payload	Type	Req	Definition
ticketId	Long(12)	X	The unique ticket identifier.
originType	Integer(2)	X	The origin type (See Index)
ticketCount	Integer(3)	X	The number of instances of this ticket to print.
ticketSequence	Integer(3)	X	The sequence number of a ticket within a ticket grouping.
printQuantity	BigDecimal(12,4)	X	The quantity to be printed on the ticket.
printDate	Date	X	The date the ticket should be printed.
autoPrint	Boolean		True if the ticket should automatically print, false if the ticket requires manual printing.
autoRefeshQuantity	Boolean		True indicates the ticket count gets refreshed with SOH at the time of printing, false it prints as is.
overridePriceCurrency	String(3)		The override price currency code. Required if an amount is entered.

overridePriceAmount	BigDecimal(20,4)	The override price amount.
countryOfManufacture	String(3)	A two letter country code denoting the country of manufacture for the item.

Example Input

```
{
   "tickets": [
      {
         "ticketId": 8,
         "originType": 2,
         "printDate": "2023-01-10T23:59:59-05:00",
         "ticketCount": 22,
         "ticketSequence": 33
      },
      {
         "ticketId": 9,
         "originType": 1,
         "printDate": "2023-01-10T23:59:59-05:00",
         "ticketCount": 44,
         "ticketSequence": 55
      }
   ]
}
```

API: printTickets

Prints the requested tickets. This can print both current tickets and previously printed tickets.

It is assumed that the calling system will have verified or retrieved the ticket ids prior. The ticket ids and archived ticket ids will not be validated.

This service operation will simply print any tickets that are found that match identifiers passed in and ignore any identifiers for which tickets are not found.

Depending on printing configuration with the system, the printing may occur synchronous and real-time or the tickets may be staged and sent asynchronously to another system after a short delay.

API Basics

Endpoint URL	{base URL}/print

Method	POST
Success Response	204 No Content
Processing Type	Synchronous/Asynchronous
Input	Ticket identifiers
Output	N/A
Maximum Input Allowed	100 total ticket ids

Input Data Definition

Payload	Type	Req	Definition
printerId	Long	X	The identifier of the printer to print the tickets on.
refreshQuantity	Boolean		If true, the quantities of all the tickets will be refreshed prior to printing. If false, they will not.
ticketIds	List<Long>		A list of ticket identifiers of the tickets to print. If this is null or empty, then archived ticket ids must contain a value. Not validated.
archivedTicketIds	List<Long>		A list of archived ticket identifiers of tickets to print. If this is null or empty, then ticket ids must contain a value. Not validated.

API: findTicketFormats

Reads all the available ticket formats.

API Basics

Endpoint URL	{base URL}/formats
Method	GET
Success Response	200 OK
Processing Type	Synchronous
Input	Query parameters
Output	List of ticket formats

Input Data Definition

Column	Type	Definition
storeId	Long(10)	Include only ticket formats available at this store.

Output Data Definition

Column	Type	Definition
formatId	Long(12)	The unique format identifier.
storeId	Long(10)	The store identifier.

description	String(100)	A description of the ticket (possibly from ticket format).
type	Integer(4)	The type of ticket format. (See Index)
zplTemplateId	Long(12)	The unique identifier to the ZPL template to be used for printing.
formatReference	String(255)	A reference to the format content to use for this format.
defaultFormat	Boolean	True if this is the default form for the type, False otherwise.
defaultPrinterId	Long(6)	Teh default printer identifier for the format.
createDate	Date	The date the format was created.
createUser	String(128)	The user that created the format.
updateDate	Date	The date the format was last updated.
updateUser	String(128)	The user that last updated the format.

API: findTicketPrinters

Finds the printers available to print tickets.

API Basics

Endpoint URL	{base URL}/printers
Method	GET
Success Response	200 OK
Processing Type	Synchronous
Input	Query parameters
Output	List of printers

Query Params

Column	Type	Definition
storeId	Long(10)	Include only ticket printers available at this store.

Output Data Definition

Column	Type	Definition
printerId	Long(6)	The unique identifier of the printer.
storeId	Long(10)	The identifier of the store the printer is assigned to.
printerType	Integer(3)	The print type of the printer (see Index).
printerDescription	String(200)	A description of the printer.
printerUri	String(300)	The URI address of the printer.
printerName	String(128)	The logical name of the printer.
defaultManifest	Boolean	True if the printer is the default printer at the store for manifest printing.
defaultPreshipment	Boolean	True if the printer is the default printer at the store for preshipment printing.

Additional Data Definitions

Ticket Format Type

ID	Description
1	Item Basket
2	Shelf Label

Price Type

ID	Description
1	Permanent
2	Promotional
3	Clearance
4	Clearance Reset

Printer Type

ID	Description
1	Item Ticket
2	Shelf Label
3	Postscript

Ticket Origin Type

ID	Description
1	External
2	Price Change
3	Foundation
4	Manual
5	Promotional Price Change
6	Clearance Price Change
7	Permanent Price Change
8	Reset Price Change

REST Service: Transfer

This service allows the creation, modification, and cancellation of transfers from an external application. This does not include allocations.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/transfers

API Definitions

API	Description
readTransfer	Reads the full detail of a transfer.
findTransfers	Finds transfers headers based on a set of input criteria to search for.
findTransferContexts	Retrieves all available transfer contexts for creating a new transfer.
requestTransfer	Moves the status of a new transfer request to requested, notifying the sending location of the transfer.
cancelRequest	Moves the status of a new transfer request to canceled if the request is still new or a work in progress.
rejectRequest	Rejects the requested transfer as long as it is still only requested or request in progress.
approveRequest	Moves the transfer to approved status and does inventory shifts if the transfer is "Requested" or "Request In Progress" and goes into "Approved"
cancelTransfer	Moves the status a transfer to canceled, but only if it is in the "Transfer In Progress" status.
closeTransfer	Closes out a transfer as long as the transfer is open and not yet in shipping status.

API: Read Transfer

Read a Transfer.

Table 4-80 API Basics

Endpoint URL	/{transferId}
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	None
Output	Transfer

Table 4-81 Path Parameter Definitions

Parameter	Definition
transferId	The internal identifier of the transfer header.

Table 4-82 Output Data Definition

Attribute	Type	Definition
transferId	Long(15)	The unique identifier of the transfer.
externalTransferId	String(128)	An external identifier supplied from an external system.

Table 4-82 (Cont.) Output Data Definition

distroNumber	String(128)	A distribution number (used for integration with other systems). It is another type of external identifier.
sourceLocationType	Integer	The location type of the source of the transfer. See Additional Data Definitions.
sourceLocationId	Long(10)	The identifier of the source location of the transfer.
destinationLocationType	Integer	The location type of the destination of the transfer. See Additional Data Definitions.
destinationLocationId	Long(10)	The identifier of the destination location of the transfer.
status	Integer	The current status of the transfer. See Additional Data Definitions
originType	Integer	The origin type of the transfer. See Additional Data Definitions
contextTypeId	Long(18)	Unique identifier of a context associated to the transfer.
contextTypeCode	Long(4)	An external code of the context.
contextValue	String(25)	A value or some information related to the context associated to the transfer.
customerOrderNumber	String(128)	External system identifier of the customer order.
fulfillmentOrderNumber	String(128)	External system identifier of the fulfillment order.
allowPartialDelivery	Boolean	True indicates that the partial delivery is allowed for the transfer, false indicates it is not.
useAvailable	Boolean	True indicates the transfer must use available stock, false indicates it uses unavailable stock.
authorizationCode	String(12)	An authorization code assigned to the transfer.
notAfterDate	Date	Date after which the transfer is no longer valid.
createDate	Date	The date this record was created.
createUser	String(128)	The user that created this record.
updateDate	Date	The date this record was last updated.
updateUser	String(128)	The user that last updated this record.
requestDate	Date	The date the transfer was requested.
requestUser	String(128)	The user that requested the transfer.
approvalDate	Date	The date the transfer was approved.
approvalUser	String(12)	The user that approved the transfer.
importId	String(128)	The identifier from an original data seed import.
lineItems	Collection	A collection of transfer line items.

Table 4-83 Transfer Line Item Data Definition

Column	Type	Definition
lineId	Long(15)	The unique internal identifier of the transfer line item.

Table 4-83 (Cont.) Transfer Line Item Data Definition

itemId	String(25)	The item identifier.
caseSize	BigDecimal(10,2)	The case size associated to this line item.
quantityRequested	BigDecimal(20,4)	The quantity that was requested.
quantityApproved	BigDecimal(20,4)	The quantity that was approved.
quantityShipping	BigDecimal(20,4)	The quantity that is currently in shipping.
quantityShipped	BigDecimal(20,4)	The quantity that has currently shipped.
quantityReceived	BigDecimal(20,4)	The quantity that has been received into stock.
quantityDamaged	BigDecimal(20,4)	The quantity that has been received as damaged.
preferredUom	String(4)	The preferred unit of measure of the transfer line item.

API: Find Transfer

This API is used to find transaction headers for transfers. If the number of transfer found exceeds 10,000, a maximum limit error will be returned. Additional or more limiting input criteria will be required.

Table 4-84 API Basics

Endpoint URL	
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	Query Parameters
Output	List of Transfer Headers

Table 4-85 Query Parameter Definitions

Attribute	Type	Definition
externalTransferId	String(128)	Include only records with this external transfer identifier.
sourceLocationId	Long(10)	Include only records with this source location identifier.
sourceLocationType	Integer	Include only records with this source location type. See Additional Data Definitions
destinationLocationId	Long(10)	Include only records with this destination location identifier.
destinationLocationType	Integer	Include only records with this destination location type. See Additional Data Definitions.
status	Integer	Include only records with this status. See Additional Data Definitions.
itemId	String(25)	Include only records with this item on them.
updateDateFrom	Date	Include records with a last update date on or after this date.

Table 4-85 (Cont.) Query Parameter Definitions

updateDateTo	Date	Include records with a first update date on or before this date.

Attribute	Type	Definition
transferId	Long(15)	The unique identifier of the transfer.
externalTransferId	String(128)	An external identifier supplied from an external system.
sourceLocationType	Integer	The type of the source location of the transfer. See Additional Data Definitions.
sourceLocationId	Long(10)	The identifier of the source location of the transfer.
destinationLocationType	Integer	The type of the destination location of the transfer. See Additional Data Definitions.
destinationLocationId	Long(10	The identifier of the destination location of the transfer.
status	Integer	The current status of the transfer. See Additional Data Definitions.
originType	Integer	The origin type of the transfer. See Additional Data Definitions.
customerOrderNumber	String(128)	The external customer order number.
fulfillmentOrderNumber	String(128)	The external fulfillment order number.
notAfterDate	Date	Date after which the transfer is no longer valid.
createDate	Date	The date this record was created.
updateDate	Date	The date this record was last updated.
requestDate	Date	The date the transfer was requested.
approvalDate	Date	The date the transfer was approved.

API: Request Transfer

This API finalizes the request of the transfer and begins the approval process.

Table 4-86 API Basics

Endpoint URL	/{transferId}/requests
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Table 4-87 Path Parameter Definitions

Parameter	Definition

Table 4-87 (Cont.) Path Parameter Definitions

transferId	The internal identifier of the transfer header.

API: Release Request

This API finalizes the request of the transfer and begins the approval process.

Table 4-88 API Basics

Endpoint URL	/{transferId}/release
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Table 4-89 Path Parameter Definitions

Parameter	Definition
transferId	The internal identifier of the transfer header.

API: Approve Request

Approves a transfer request for shipment moving it to the shipping stage.

Table 4-90 API Basics

Endpoint URL	/{transferId}/requests/approve
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Table 4-91 Path Parameter Definitions

Parameter	Definition
transferId	The internal identifier of the transfer header.

API: Reject Request

Rejects the transfer request closing out the transfer. This is done when the approver of a request decides not to fulfill it.

Table 4-92 API Basics

Endpoint URL	/{transferId}/reject
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Table 4-93 Path Parameter Definitions

Parameter	Definition
transferId	The internal identifier of the transfer header.

API: Cancel Request

Cancels a transfer request closing out the transfer. This is done when the requester decides the transfer is not needed prior to finishing their request.

Table 4-94 API Basics

Endpoint URL	/{transferId}/cancel
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Table 4-95 Path Parameter Definitions

Parameter	Definition
transferId	The internal identifier of the transfer header.

API: Approve Transfer

Approves a transfer for shipment, moving it to the shipping stage.

Table 4-96 API Basics

Endpoint URL	/{transferId}/approve
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None

Table 4-96 (Cont.) API Basics

Output	None

Table 4-97 Path Parameter Definitions

Parameter	Definition
transferId	The internal identifier of the transfer header.

API: Cancel Transfer

Cancels a transfer that has been approved.

Table 4-98 API Basics

Endpoint URL	/{transferId}/cancel
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Table 4-99 Path Parameter Definitions

Parameter	Definition
transferId	The internal identifier of the transfer header.

API: Close Transfer

Closes a transfer that has been approved or approved and partially shipped.

Table 4-100 API Basics

Endpoint URL	/{transferId}/close
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Table 4-101 Path Parameter Definitions

Parameter	Definition
transferId	The internal identifier of the transfer header.

API: Find Transfer Contexts

Reads all the available contexts for a transfer.

Table 4-102 API Basics

Endpoint URL	/contexts
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	None
Output	List of Transfer Contexts

Table 4-103 Output Data Definition

Attribute	Type	Definition
contextId	Long(18)	The unique identifier of the context.
code	String(6)	A short code associated to the context
description	String(120)	A description of the context.
sequence	Long	A sorting sequence for the context.

Additional Data Definitions

Table 4-104 Transfer Status

Value	Definition
1	New Request
2	Requested
3	Request In Progress
4	Rejected Request
5	Canceled Request
6	Transfer In Progress
7	Approved
8	In Shipping
9	Completed Transfer
10	Canceled Transfer

Table 4-105 Transfer Location Type

Value	Definition
1	Store
2	Warehouse
3	Finisher

Table 4-106 Transfer Origin Type

Value	Definition
1	External
2	Internal
3	Adhoc

Table 4-107 Transfer Status For Query Parameters

Value	Definition
1	New Request
2	Requested
3	Request In Progress
4	Rejected Request
5	Canceled Request
6	Transfer In Progress
7	Approved
8	In Shipping
9	Completed Transfer
10	Canceled Transfer
99	Active

REST Service: Transfer Shipment

This service allows the creation, modification, and cancellation of transfer shipments from an external application.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/tsfshipments

API Definitions

API	Definition
readShipment	Reads the transfer shipment. This does not include containers or items.
findShipments	Finds transfer shipment header information based on a set of criteria.
submitShipment	Submits the transfer shipment.
cancelSubmitShipment	Cancels the submission of a transfer shipment.
dispatchShipment	Dispatches a transfer shipment.
cancelShipment	Cancels a transfer shipment.

confirmCarton	Confirms a container for shipment.
cancelCarton	Cancels a container, removing it from the shipment.
openCarton	Opens a confirmed container so that it can be modified again.
createShipment	Creates a shipment.
updateShipment	Updates shipment header information.
createCarton	Adds a carton to the shipment.
updateCarton	Updates a carton on the shipment.

API: Read Shipment

Read a transfer shipment.

Table 4-108 API Basics

Endpoint URL	/{shipmentId}
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	None
Output	Transfer Shipment

Table 4-109 Path Parameter Definitions

Parameter	Definition
shipmentId	The internal identifier of the transfer shipment header.

Table 4-110 Output Data Definition

Column	Type	Definition
shipmentId	Long(15)	The unique internal identifier of the transfer shipment.
storeId	Long(10)	The unique store identifier that is the source of the shipment.
destinationType	Integer(2)	The location type of the destination. See Additional Data Definitions-Destination Type.
destinationId	Long(10)	The unique identifier of the destination.
asn	String(128)	The advance shipment notification number.
status	Integer(2)	The current status of the shipment. See Additional Data Definitions: Status
authorizationCode	String(12)	An authorization code associated to the shipment.
billOfLadingId	Long(12)	The unique identifier of the bill of lading.
adhocDocumentId	Long(15)	An adhoc transfer document identifier associated to the shipment.
billOfLadingId	Long(15)	The bill of lading number.

Table 4-110 (Cont.) Output Data Definition

alternateAddress	String(2000)	An alternate destination address.
carrierRole	Integer(2)	The type of carrier for the shipment. See Additional Data Definitions
carrierId	Long(10)	A unique identifier of a carrier for the shipment.
carrierServiceId	Long(10)	A unique identifier of a carrier service for the shipment.
alternateCarrierName	String(240)	The name of a third-party shipping company.
alternateCarrierAddress	String(2000)	The address of a third-party shipping company.
motive	String(120)	A motive for the shipment.
taxId	String(18)	The tax identifier of the supplier it is being shipped to.
trackingNumber	String(128)	A tracking number associated to the shipment.
dimensionId	Long(12)	The identifier of a dimension associated to the shipment.
weight	BigDecimal(12, 4)	The weight of the shipment.
weightUom	String(4)	The unit of measure of the weight of the shipment.
requestedPickupDate	Date	The requested pickup date.
fiscalDocumentRequestId	Long(20)	The identifier of the request for a fiscal document.
fiscalDocumentReferenceId	Long(20)	The unique identifier of the fiscal document.
fiscalDocumentNumber	String(255)	The fiscal document number.
fiscalDocumentStatus	Integer(4)	The status of the fiscal document.
fiscalDocumentRejectReason	String(255)	A reason the fiscal document was rejected.
fiscalDocumentUrl	String(255)	A URL to the fiscal document.
createUser	String(128)	The user that created the shipment record.
createDate	Date	The date the shipment record was created.
updateUser	String(128)	The user that last updated the shipment.
updateDate	Date	The last date the shipment was updated.
submitUser	String(128)	The user that submitted the shipment record.
submitDate	Date	The date the shipment was submitted within EICS.
dispatchUser	String(128)	The user that dispatched the shipment.
dispatchDate	Date	The date the shipment was dispatched within EICS.
cartons	Collection	A collection of cartons on the shipment.

Table 4-111 Output Data Definition (Cartons)

Column	Type	Definition
cartonId	Long(10)	The unique identifier of the record.
externalCartonId	String(128)	A container identifier from an external system.
status	Integer(4)	The current status of the container.

Table 4-111 (Cont.) Output Data Definition (Cartons)

dimensionId	Long(10)	The shipment container dimension identifier.
weight	BigDecimal(12,4)	The weight of the container.
weightUom	String(4)	The unit of measure of the weight of the container.
trackingNumber	String(128)	A tracking number for the container.
useAvailableInventory	Boolean	True indicates use only available inventory, False indicates use non-available inventory.
restrictionLevel	Integer(4)	A hierarchy restriction level for items in the container.
customerOrderRelated	Integer(4)	The customer order related value (see Additional Data Definitions).
createUser	String(128)	The user that created the container in EICS.
createDate	Date	The date the container was created in EICS.
updateUser	String(128)	The user that last updated the container in EICS.
updateDate	date	The date the container was last updated in EICS.
approvalUser	String(128)	The user that approved the container in EICS.
approvalDate	Date	The date the container was approved in EICS.
lineItems	Collection	The line items in the container.

Table 4-112 Output Data Definition (Line Item)

Column	Type	Definition
lineId	Long(15)	The unique identifier of the record.
itemId	String(25)	The unique identifier of the item.
caseSize	BigDecimal(10,2)	The case size of this line item.
quantity	BigDecimal(20,4)	The quantity being shipped.
transferId	Long(12)	The unique identifier of the associated transfer.
externalTransferId	Long(15)	The external system identifier of the associated transfer.
customerOrderNumber	String(128)	The customer order number if line item is for a customer order.
fulfillmentOrderNumber	String(128)	The fulfillment order number if line item is for a customer order.
preferredUom	String(4)	The preferred unit of measure for the shipment quantity.
shipmentReasonId	Long(15)	A unique identifier of a shipment reason associated to this item.
uins	List<String>	A list of UINs that are shipped. The number of UINs must match the quantity attribute.

Example

Figure 4-15 Example


```
"links": [
    {
      "href": "/tsfshipments/1",
      "rel": "self"
    },
    {
      "href": "/tsfshipments/1/submit",
      "rel": "submit"
    },
    {
      "href": "/tsfshipments/1/cancelsubmit",
      "rel": "cancelsubmit"
    },
    {
      "href": "/tsfshipments/1/dispatch",
      "rel": "dispatch"
    },
    {
      "href": "/tsfshipments/1/cancel",
      "rel": "cancel"
    }
  ],
  "shipmentId": 1,
  "storeId": 5000,
  "destinationType": 1,
  "destinationId": 5001,
  "asn": "1",
  "status": 2,
  "billOfLading": 1,
  "adhocDocumentId": 1,
  "createUser": "15000",
  "createDate": "2024-02-07T09:02:31-06:00",
  "updateUser": "15000",
  "updateDate": "2024-02-07T09:02:31-06:00",
  "cartons": [
    {
      "cartonId": 1,
      "externalCartonId": "1",
      "status": 2,
      "useAvailable": true,
      "restrictionLevel": 4,
      "customerOrderRelated": 3,
      "createUser": "15000",
      "createDate": "2024-02-07T09:02:31-06:00",
      "updateUser": "15000",
      "updateDate": "2024-02-07T09:02:31-06:00",
      "lineItems": [
        {
          "lineId": 1,
          "itemId": "100701234",
          "caseSize": 50.0000,
          "quantity": 1.0000,
```

Figure 4-16 Example, continued

```
        "transferId": 1
      },
      {
        "lineId": 7,
        "itemId": "100801236",
        "caseSize": 50.0000,
        "quantity": 1.0000,
        "transferId": 1
      }
    ]
  },
  {
    "cartonId": 2,
    "externalCartonId": "2",
    "status": 2,
    "trackingNumber": "1111",
    "useAvailable": true,
    "restrictionLevel": 4,
    "customerOrderRelated": 3,
    "createUser": "15000",
    "createDate": "2024-02-07T09:02:31-06:00",
    "updateUser": "15000",
    "updateDate": "2024-02-07T09:02:31-06:00",
    "lineItems": [
      {
        "lineId": 8,
        "itemId": "101386248",
        "caseSize": 50.0000,
        "quantity": 1.0000,
        "transferId": 1
      },
      {
        "lineId": 15,
        "itemId": "100681077",
        "caseSize": 100.0000,
        "quantity": 1.0000,
        "transferId": 1
      }
    ]
  }
]
}
```


API: Find Transfer Shipment

Search for shipments based on a set of criteria.

If more than 10,000 shipments are found a "Results Too Large" response is returned and search criteria will need to be more restrictive.

Table 4-113 API Basics

Endpoint URL	
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	Query Parameters
Output	List of Transfer Shipments Headers

Table 4-114 Query Parameter Definitions

Attribute	Type	Definition
storeId	Long(10)	Include only transfer shipments from this store.

Table 4-114 (Cont.) Query Parameter Definitions

destinationType	Integer(2)	Include only transfer shipments to this destination type.
destinationId	Long(10)	Include only transfer shipments to this destination identifier.
asn	String(128)	Include only transfer shipments with this ASN.
status	Integer(2)	Include only transfer shipments in this status.
updateDateFrom	String	Include only transfer shipments last updated on or after this date.
updateDateTo	String	Include only transfer shipments last updated on or before this date.

Table 4-115 Output Data Definition

Column	Type	Definition
shipmentId	Long(15)	The unique internal identifier of the transfer shipment.
storeId	Long(10)	The unique store identifier that is the source of the shipment.
destinationType	Integer(2)	The location type of the destination. See Additional Data Definitions: Destination Type.
destinationId	Long(10)	The unique identifier of the destination location.
asn	String(128)	The advance shipment notification number.
status	Integer(4)	The current status of the shipment. See Additional Data Definitions: Status
createDate	Date	The date the shipment record was created.
updateDate	Date	The last date the shipment was updated.
submitDate	Date	The date the shipment was submitted within EICS.
dispatchDate	Date	The date the shipment was dispatched within EICS.

API: Submit Shipment

Submits the transfer shipment moving it to a noneditable state while it is waiting to be dispatched.

Table 4-116 API Basics

Endpoint URL	/{shipmentId}/submit
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Table 4-117 Path Parameter Definitions

Parameter	Definition
shipmentId	The internal identifier of the transfer shipment header.

API: Cancel Submit Shipment

Cancels the submissions of the transfer shipment return it to an in progress and editable state.

Table 4-118 API Basics

Endpoint URL	/{shipmentId}/cancelsubmit
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Table 4-119 Path Parameter Definitions

Parameter	Definition
shipmentId	The internal identifier of the transfer shipment header.

API: Dispatch Shipment

Dispatches the transfer shipment updating all the inventory positions and closing the shipment.

Table 4-120 API Basics

Endpoint URL	/{shipmentId}/dispatch
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Table 4-121 Path Parameter Definitions

Parameter	Definition
shipmentId	The internal identifier of the transfer shipment header.

API: Cancel Shipment

Cancels the transfer shipment reversing any reserved inventory currently picked.

Table 4-122 API Basics

Endpoint URL	/{shipmentId}/cancel
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Table 4-123 Path Parameter Definitions

Parameter	Definition
shipmentId	The internal identifier of the transfer shipment header.

API: Confirm Carton

Confirms a transfer shipment container, completing the container and making it non-editable and awaiting dispatch.

Table 4-124 API Basics

Endpoint URL	cartons/{cartonId}/confirm
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Table 4-125 Path Parameter Definitions

Parameter	Definition
cartonId	The internal identifier of the transfer shipment carton.

API: Cancel Carton

Cancels a transfer shipment container.

Table 4-126 API Basics

Endpoint URL	cartons/{cartonId}/cancel
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None

Table 4-126 (Cont.) API Basics

Output	None

Table 4-127 Path Parameter Definitions

Parameter	Definition
cartonId	The internal identifier of the transfer shipment carton.

API: Open Carton

Opens a transfer shipment container.

Table 4-128 API Basics

Endpoint URL	cartons/{cartonId}/open
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Table 4-129 Path Parameter Definitions

Parameter	Definition
cartonId	The internal identifier of the transfer shipment carton.

API: createShipment

This API is used to create a new transfer shipment whose status is "In Progress."

> **Note:**
> Container item/reason combination cannot be duplicated within the container.

Table 4-130 API Basics

Endpoint URL	{base URL}
Method	POST
Successful Response	200 OK
Processing Type	Synchronous
Input	Transfer Shipment
Output	Transfer Shipment Status

Table 4-130 (Cont.) API Basics

Maximum Input Limit	1,000 overall line items on shipment

Table 4-131 Input Data Definition

Attribute	Type	Req	Definition
storeId	Long(10	X	The identifier of the store shipping the goods.
destinationType	Integer(2)	X	The location type of the destination (see Index).
destinationId	Long(10)	X	The location identifier of the destination
asn	String(15)		The advance shipping number from an external system.
authorizationCode	String(12)		A vendor authorization code. It may be required for some suppliers.
displayCartons	Boolean		True if cartons should be displayed, otherwise false.
addressType	Integer		The type of return address. See Index.
carrierRole	Integer(2)	X	The type of carrier for the shipment. See Index
carrierId	Long(10)		A unique identifier of a carrier for the shipment.
carrierServiceId	Long(10)		A unique identifier of a carrier service for the shipment.
alternateAddress	String(2000)		An alternate destination address.
alternateCarrierName	String(240)		The name of a third-party shipping company.
alternateCarrierAddress	String(2000)		The address of a third-party shipping company.
motiveId	Long(18)		The unique identifier of a bill of lading motive.
taxId	taxId		The tax identifier of the supplier it is being shipped to.
trackingNumber	String(128)		A tracking number associated to the shipment.
dimensionId	Long(12)		The identifier of a dimension associated to the shipment.
weight	BigDecimal(12,4)		The weight of the shipment.
weightUom	String(4)		The unit of measure of the weight of the shipment.
requestedPickupDate	Date		The requested pickup date.
cartons	Collection	X	A group of cartons to create along with the shipment.
notes	Collections of Strings		A collection of up to 100 notes.

Table 4-132 Shipment Carton Data Definition

Payload	Type	Req	Definition
externalCartonId	String(128)		A container identifier from an external system.

Table 4-132 (Cont.) Shipment Carton Data Definition

trackingNumber	String(128)		The tracking number of the container.
restrictionLevel	Integer(4)		A hierarchy restriction level for items in the container.
cartonSizeId	Long(10)		The shipment container dimension identifier.
weight	Long(12,4)		The weight of the container.
weightUom	String(4)		The unit of measure of the weight of the container.
useAvailableInventory	Boolean		True indicates available inventory will be used, false indicates unavailable inventory should be used.
lineItems	Collection	X	A collection of line items to create along with the carton.

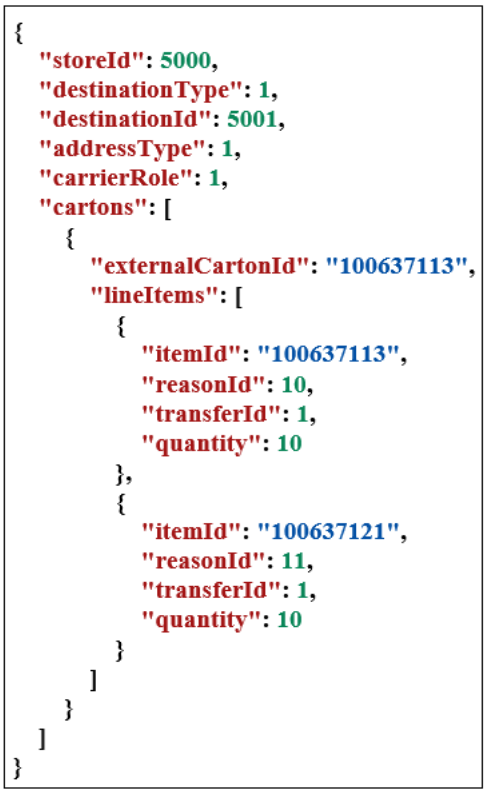
Table 4-133 Shipment Lien Item Data Definition

Payload	Type	Req	Definition
itemId	String(25)	X	The unique identifier of the SKU item.
transferId	Long(12)	X	The unique identifier of a transfer this item is being shipped for.
reasonId	Long(15)		The unique identifier of the shipment reason associated to this line item. It is not allowed for available inventory and required for unavailable inventory.
quantity	BigDecimal(12,4)	X	The quantity to ship.
caseSize	BigDecimal(10,2)		The case size of the item for this particular shipment.
uins	Collecton<String>		A collection of UINs to ship. This number of UINs must match the quantity.

Table 4-134 Output Data Definition

Attribute	Type	Definition
shipmentId	Long(15)	The unique identifier of the shipment.
storeId	Long(10)	The store identifier of the store shipping the goods.
asn	String(15)	The Advancing Shipping Number.
status	Integer(2)	The shipment status. See Index.

Figure 4-17 Example Code



```
{
  "storeId": 5000,
  "destinationType": 1,
  "destinationId": 5001,
  "addressType": 1,
  "carrierRole": 1,
  "cartons": [
    {
      "externalCartonId": "100637113",
      "lineItems": [
        {
          "itemId": "100637113",
          "reasonId": 10,
          "transferId": 1,
          "quantity": 10
        },
        {
          "itemId": "100637121",
          "reasonId": 11,
          "transferId": 1,
          "quantity": 10
        }
      ]
    }
  ]
}
```

API: updateShipment

This API is used to update to the header portion of a shipment as well as its bill of lading information.

The shipment header cannot be updated while containers/cartons are currently confirmed for shipping.

Table 4-135 API Basics

Endpoint URL	{base URL}/{shipmentId}
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Path Parameter	The identifier of the transfer shipment
Input	Transfer Shipment

Table 4-136 Input Data Definition

Column	Type	Definition
authorizationCode	String(12)	An authorization code associated to the shipment.
displayCartons	Boolean	True if cartons should be displayed, otherwise false.

Table 4-136 (Cont.) Input Data Definition

addressType	Integer(2)	The type of return address. See Index.
carrierRole	Integer(2)	The type of carrier for the shipment. See Index.
carrierId	Long(10)	A unique identifier of a carrier for the shipment.
carrierServiceId	Long(10)	A unique identifier of a carrier service for the shipment.
alternateAddress	String(2000)	An alternate destination address.
alternateCarrierName	String(240)	The name of a third-party shipping company.
alternateCarrierAddress	String(2000)	The address of a third-party shipping company
motiveId	Long(18)	The unique identifier of a bill of lading motive.
taxId	String(18)	The tax identifier of the supplier it is being shipped to.
trackingNumber	String(128)	A tracking number associated to the shipment.
dimensionId	Long(12)	The identifier of a dimension associated to the shipment.
weight	BigDecimal(12, 4)	The weight of the shipment.
weightUom	String(4)	The unit of measure of the weight of the shipment.
requestedPickupDate	Date	The requested pickup date.
notes	Collection(String)	A collection of up to 100 notes to add to the notes associated to the shipment.

Figure 4-18 Code Example

```
{
  "carrierRole": 2,
  "authorizationCode": "6543",
  "trackingNumber": "A67B",
  "addressType": 2
}
```

API: createCarton

This API is used to add a new carton/container to a "New" or "In Progress" shipment.

Container item/reason combination cannot be duplicated within the container.

Table 4-137 API Basics

Endpoint URL	{base URL}/{shipmentId}/cartons
Method	POST
Successful Response	200 OK
Processing Type	Synchronous
Path Parameter	The identifier of the transfer shipment
Input	Transfer Shipment Carton

Table 4-137 (Cont.) API Basics

Output	Transfer Shipment Carton Status
Maximum Input Limit	1,000 overall line items on carton

Table 4-138 Input Data Definition

Column	Type	Req	Definition
externalCartonId	String(128)		A carton identifier or barcode label from an external system.
cartonSizeId	Long(10)		The shipment container dimension identifier.
weight	BigDecimal(12,4)		The weight of the container.
weightUom	String(4)		The unit of measure of the weight of the container.
trackingNumber	String(128)		A tracking number for the container.
useAvailableInventory	Boolean		True indicates use only available inventory, False indicates use non-available inventory.
restrictionLevel	Integer(4)		A hierarchy restriction level for items in the container.
lineItems	Collection	X	The line items in the container.

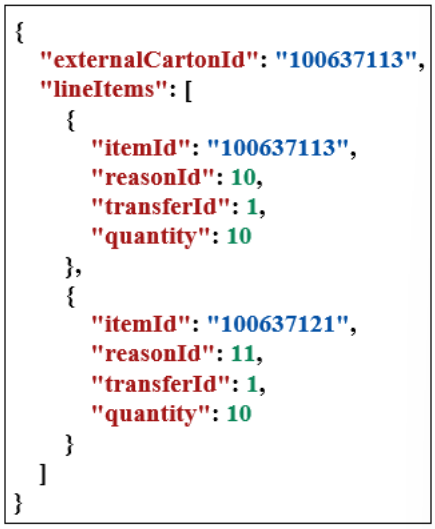
Table 4-139 Shipment Line Item Data Definition

Column	Type	Req	Definition
itemId	String(25)	X	The unique identifier of the item.
caseSize	BigDecimal(10,2)		The case size of this line item.
quantity	BigDecimal(20,4)	X	The quantity to be shipped.
transferId	Long(12)	X	The unique identifier of the associated transfer.
reasonId	Long(15)		A unique identifier of a shipment reason associated to this item. It is not allowed for available inventory and required for unavailable inventory.
uins	List<String>		A list of UINs to be shipped. This must match the quantity.

Table 4-140 Output Data Definition

Attribute	Type	Definition
shipmentId	Long(15)	The unique identifier of the shipment.
cartonId	Long(15)	The unique identifier of the new carton.
externalId	String(128)	The external identifier or barcode label of the container.
status	Integer(2)	The carton status. See Index.

Figure 4-19 Code Example



```
{
  "externalCartonId": "100637113",
  "lineItems": [
    {
      "itemId": "100637113",
      "reasonId": 10,
      "transferId": 1,
      "quantity": 10
    },
    {
      "itemId": "100637121",
      "reasonId": 11,
      "transferId": 1,
      "quantity": 10
    }
  ]
}
```

API: updateCarton

This API is used to update an existing carton that is in "New" or "In Progress" shipment.

Table 4-141 API Basics

Endpoint URL	{base URL}/cartons/{cartonId}
Method	POST
Successful Response	200 OK
Processing Type	Synchronous
Path Parameter	The identifier of the carton
Input	Transfer Shipment Carton
Output	Transfer Shipment Carton Status
Maximum Input Limit	1,000 overall line items on carton

Table 4-142 Input Data Definition

Payload	Type	Req	Definition
externalCartonId	String(128)		A container identifier from an external system.
trackingNumber	String(128)		The tracking number of the container.
restrictionLevel	Integer(4)		A hierarchy restriction level for items in the container.
cartonSizeId	Long(10)		The shipment container dimension identifier.
weight	Long(12,4)		The weight of the container.
weightUom	String(4)		The unit of measure of the weight of the container.

Table 4-142 (Cont.) Input Data Definition

lineItems	Collection		A collection of up to 1,000 line items to update or add within the carton. See TransferShipmentUpdateCartontemIdo.

Payload	Type	Req	Definition
itemId	String(25)	X	The unique identifier of the SKU item.
transferId	Long(12)	X	The unique identifier of the associated transfer.
reasonId	Long(15)		The unique identifier of the shipment reason associated to this line item. It is not allowed for available inventory and required for unavailable inventory.
quantity	BigDecimal(12,4)	X	The quantity shipped. Reducing this to 0 will remove the line item from the shipment.
caseSize	BigDecimal(10,2)		The case size of the item for this particular shipment.
uins	Collection<String>		The UINs associated to the item quantities. The number of UINS must match the quantity shipped.

Figure 4-20 Code Example

```
{
  "externalCartonId": "100637113",
  "lineItems": [
    {
      "itemId": "100637113",
      "reasonId": 10,
      "transferId": 1,
      "quantity": 10
    },
    {
      "itemId": "100637121",
      "reasonId": 11,
      "transferId": 1,
      "quantity": 10
    }
  ]
}
```

Index

Table 4-143 Destination Type

Value	Description
1	Store

Table 4-143 (Cont.) Destination Type

2	Warehouse
3	Finisher

Table 4-144 Status

Value	Description
1	New
2	In Progress
3	Submitted
4	Shipped
5	Canceled

Table 4-145 Customer Order Related

Value	Description
1	Yes
2	Mixed
3	No

REST Service: Translations

This page captures the service APIs related to retrieving translations. It allows the translation of such things as labels and item descriptions.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/translations

API Definitions

API	Description
Find Locales	Finds all locales that can be used to translation a series of text keys.
Find Translations	Finds the translations for a series of text keys, translating into text for the locale if it is available.
Find Item Descriptions	Finds the translations for item descriptions, translating it to the text of the locale if it is available.

API: Find Locales

Finds all locales available for use in translation.

API Basics

Endpoint URL	{base URL}/locales
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	Criteria
Output	List of locales
Max Response Limit	N/A

Output Data Definition

Attribute	Data Type	Description
localeId	Long	The SIOCS internal unique identifier of the record.
language	String	A code representing a language (ISO 639 alpha-2 or alpha-3 language code).
country	String	A code representing a country of the language (ISO 3166 alpha-2 country code or UN M.49 numeric-3 area code.)
variant	String	A code representing a variant of the country of the language (an arbitrary value indication the variant).
description	String	A description of the locale.

Example Output

[

{

"localeId": 1,

"language": "en",

"description": "English"

},

{

"localeId": 2,

"language": "de",

"description": "German"

},

{

"localeId": 3,

"language": "fr",

"description": "French"

}

]

API: Find Translations

Searches for translations for text keys and a given locale.

API Basics

Endpoint URL	{base URL}/find
Method	POST
Successful Response	200 OK
Processing Type	Synchronous
Input	Criteria
Output	A map of translation key and its translation
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
localeId	Long(12)	Yes	Unique identifier of the Locale to translate keys for (see find Locales).
keys	List<String(600)>	Yes	A list of text keys to attempt to translate.

Example Input

{

"localeId": 1,

"keys": [

"invAdjReason.1",

"invAdjReason.2"

]

}

Output Data Definition

Attribute	Data Type	Description
values	Map<String, String>	A map where the key is the translation key and the value is the translation value.

Example Output

{

"invAdjReason.2": "Shrinkage",

"invAdjReason.1": "Wastage"

}

API: Find Items Descriptions

Finds the translations for item descriptions, translating it to the text of the locale if it is available.

API Basics

Endpoint URL	{base URL}/items
Method	POST
Successful Response	200 OK
Processing Type	Synchronous
Input	Criteria
Output	A list of translation items
Max Response Limit	N/A

Input Data Definition

Attribute	Data Type	Required	Description
LocaleId	Long (12)	Yes	Unique identifier of the Locale to translate descriptions for (see findLocales).
itemIds	List<String(25)>	Yes	A list of items to get descriptions for within the locale.

Example Input

{

"localeId": 1,

"itemIds": [

"100637121",

"100637113"

]

}

Output Data Definition

Attribute	Data Type	Required	Description
itemId	String		The item identifier.
shortDescription	String		The short description in the locale's text if available.
longDescription	String		The long description in the locale's text if available.

Example Output

[
{
"itemId": "100637121",
"shortDescription": "translation value for 100637121",
"longDescription": "translation value for 100637121"
},
{
"itemId": "100637113",
"shortDescription": "translation value for 100637113",
"longDescription": "translation value for 100637113"
}
]

REST Service: Vendor Delivery

This service allows the import and handling of direct store deliveries from vendors/suppliers.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/dsds

API Definitions

API	Description
readDelivery	Read the full details of a vendor delivery.
findDeliveries	Find vendor delivery headers based on input search criteria.
receiveDelivery	Receives the expected quantities of a vendor delivery so that it is ready to be confirmed.
confirmDelivery	Confirm the receipt of a vendor delivery updating inventory positions with the receipt information.
rejectDelivery	Rejects the vendor delivery and do not allow it to be received.
cancelDelivery	Cancel a vendor delivery.
submitCarton	Moves the status of the carton to submitted and prevents further updates. The carton must still be confirmed. No inventory positions are updated via this operation.
cancelSubmitCarton	Opens a submitted carton for further updates, moving the status back to in-progress.
confirmCarton	Confirms the receipt a vendor delivery carton.

cancelCarton	Cancels a vendor delivery carton.
openCarton	Re-opens a completed carton after receipt allowing it to be received a second time (possibly adjusting quantities).

API: Read Delivery

Retrieves a vendor delivery.

Table 4-146 API Basics

Endpoint URL	/{deliveryId}
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	None
Output	Transfer Shipment

Table 4-147 Path Parameter Definitions

Parameter	Definition
deliveryId	The internal identifier of the vendor delivery header.

Table 4-148 Output Data Definition

Column	Type	Definition
deliveryId	Long(12)	The unique identifier of the delivery record.
storeId	Long(10)	The unique identifier of the store receiving the inventory.
supplierId	Long(10)	The unique identifier of the supplier shipping the inventory.
status	Integer(4)	The current status of the delivery. See Vendor Delivery Status.
originType	Integer(2)	The origin type of the delivery. See Vendor Delivery Origin Type.
purchaseOrderId	Long(12)	The purchase order that the delivery is associated to.
receiptNumber	Long	The global receipt number of the delivery used to identifier the receipt across applications through integration.
asn	String(128)	The advanced shipping notification of the delivery.
invoiceNumber	String(128)	A unique identifier of an invoice associated to this delivery.
invoiceCurrency	String	A currency code of the invoice cost.
invoiceAmount	BigDecimal	The value of the invoice cost.

Table 4-148 (Cont.) Output Data Definition

customerOrderId	String(128)	A customer order identifier (from an external system) associated to the delivery.
fulfillmentOrderId	String(128)	A fulfillment order identifier (from an external system) associated to the delivery.
billOfLadingId	String(128)	An external identifier of a bill of lading record.
carrierName	String(128)	The name of the carrier.
carrierType	Integer(2)	The type of the carrier. See Carrier Type.
carrierCode	String(4)	Unique code that identifies the carrier.
countryCode	String(3)	A country code.
sourceAddress	String(1000)	The address of the source shipping location sending the delivery to the store.
licensePlate	String(128)	The license plate of the delivery vehicle.
freightId	String(128)	A freight identifier associated to the delivery.
fiscalDocumentRequestId	Long(20)	The identifier of the request for a fiscal document.
fiscalDocumentReferenceId	Long(20)	The unique identifier of the fiscal document.
fiscalDocumentNumber	String(255)	The fiscal document number.
createDate	Date	The date the delivery record was created.
updateDate	Date	The date the delivery record was last updated.
expectedDate	Date	The expected date of the delivery.
invoiceDate	Date	The date of the delivery invoice.
receivedDate	Date	The date the delivery was received.
createUser	String(128)	The user that created the delivery .record.
updateUser	String(128)	The user who last updated the delivery record.
receivedUser	String(128)	The user who received the delivery record.
cartons	Collection	A list of cartons.

Table 4-149 Open Data Definition (Carton)

Column	Type	Definition
cartonId	Long(12)	The unique identifier of the carton record.
externalCartonId	String(128)	An external identifier of the carton.
referenceId	String(128)	A reference identifier to the carton.
status	Integer(4)	The current status of the carton (See Additional Data Definitions Vendor Delivery Carton Status)
damagedReason	String(128)	A reason for the carton damage that took place.
serialCode	Long(18)	A serial code for the carton.
trackingNumber	String(128)	The tracking number of the carton.

Table 4-149 (Cont.) Open Data Definition (Carton)

damageRemaining	Boolean	indicates all remaining quantities should be damaged on final receipt.
uinRequired	Boolean	True if a UIN item exists within the carton, otherwise false.
receiveAtShopFloor	Boolean	True if receive the inventory at shop floor, otherwise false.
qualityControl	Boolean	True indicates that the carton requires detailed receiving.
externalCreate	Boolean	True indicates the carton was externally created, false indicates it was created by EICS.
adjusted	Boolean	True indicates the carton is adjusted, otherwise false.
customerOrderRelated	Integer(4)	Customer Order Related Type (See Additional Data Definitions)
createUser	String(128)	The user who created the carton.
updateUser	String(128)	The user who last updated the carton.
receivedUser	String(128)	The user who received the carton.
createDate	Date	The date the carton was created.
updateDate	Date	The date the carton was last updated.
receivedDate	Date	The date the carton was received.
lineItems	Collection	The line items associated with the container.

Table 4-150 Output Data Definition (Line Item)

Column	Type	Definition
lineId	Long(12)	The unique identifier of the line item record.
itemId	String(25)	The unique identifier of the item.
caseSize	BigDecimal(10,2)	A number of units in the case that this item was shipped with.
quantityExpected	BigDecimal(20,4)	The total number of units expected on the delivery.
quantityReceived	BigDecimal(20,4)	The total number of units received on the delivery.
quantityDamaged	BigDecimal(20,4)	The total number of units received as damaged on the delivery.
quantityReceivedOverage	BigDecimal(20,4)	Amount of received inventory over expected quantities.
quantityDamagedOvarage	BigDecimal(20,4)	Amount of received damage inventory over expected quantities.
quantityPreviouslyReceived	BigDecimal(20,4)	Units previously received (captured at time container is re-opened after receipt)
quantityPreviouslyDamaged	BigDecimal(20,4)	Units previously received as damaged (captured at time container is re-opened after receipt)
unitCostCurrency	String(3)	The unit cost currency of this item delivery.

Table 4-150 (Cont.) Output Data Definition (Line Item)

unitCostAmount	BigDecimal(12,4)	The unit cost value of this item delivery.
unitCostOverrideCurrency	String(3)	The override unit cost currency of this item delivery.
unitCostOverrideAmount	BigDecimal(12,4)	The override unit cost value of this item delivery.
purchaseOrderId	Long (12)	The internal unique identifier of the purchase order of this particular item delivery.
purchaseOrderNumber	String(128)	The external purchase order number of this particular item delivery.
customerOrderNumber	String(128)	The unique external customer order identifier of this particular item delivery.
fulfillmentOrderNumber	String(128)	The unique external fulfillment order identifier of this particular item delivery.
vendorProductNumber	String(256)	The vendor product number of the item.
uins	Collection<String>	A list of UINs associated to the line item.

API: Find Vendor Delivery

API is used to find transaction headers for vendor deliveries.

If more than 10,000 deliveries are found, a "results too large" error will be returned. Limit the results with further search criteria.

Table 4-151 API Basics

Endpoint URL	
Method	GET
Successful Response	200 OK
Processing Type	Synchronous
Input	Query Parameters
Output	List of Vendor Shipments

Table 4-152 Query Parameter Definitions

Attribute	Type	Definition
storeId	Long(10)	Include only records where the delivery is to this store.
asn	String(128)	Include only records for this advanced shipping notification.
originType	Integer(2)	Include only records for this origin type. (See Additional Data Definitions: Vendor Delivery Origin Type).
status	Integer(4)	Include only records where the delivery is in this status. (See Additional Data Definitions: Vendor Delivery Criteria Status)

Table 4-152 (Cont.) Query Parameter Definitions

customerOrderNumber	String(128)	Include only records for this customer order number.
fulfillmentOrderNumber	String(128)	Include only records for this fulfilment order number.
invoiceNumber	String(128)	Include only records for this invoice number.
supplierId	Long(10)	Include only records for this supplier identifier.
purchaseOrderNumber	String(12)	Include only records where a line item is associated to this external purchase order number.
updateDateFrom	String	Include only records with a last update date on or after this date.
updateDateto	String	Include only records with a first update date on or before this date.

Table 4-153 Output Data Definition

Column	Type	Definition
deliveryId	Long(12)	The unique identifier of the delivery record.
storeId	Long(10)	The unique identifier of the store receiving the inventory.
supplierId	Long(10)	The unique identifier of the supplier shipping the inventory.
status	Integer(4)	The current status of the delivery. See Vendor Delivery Status.
originType	Integer(2)	The origin type of the delivery. See Vendor Delivery Origin Type.
purchaseOrderId	Long(12)	The purchase order that the delivery is associated to.
receiptNumber	Long	The global receipt number of the delivery used to identifier the receipt across applications through integration.
asn	String(128)	The advanced shipping notification of the delivery.
invoiceId	String(128)	A unique identifier of an invoice associated to this delivery.
customerOrderId	String(128)	A customer order identifier associated to the delivery.
fulfillmentOrderId	String(128)	A fulfillment order identifier (from an external system) associated to the delivery.
createDate	Date	The date the delivery record was created.
updateDate	Date	The date the delivery record was last updated.
expectedDate	Date	The expected date of the delivery.
invoiceDate	Date	The date of the delivery invoice.
receivedDate	Date	The date the delivery was received.

API: Receive Delivery

Updates the received quantities to quantities that were expected so that it is ready to be confirmed. It puts the delivery in the "In Progress" status.

Table 4-154 API Basics

Endpoint URL	/{deliveryId}/receive
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Table 4-155 Path Parameter Definitions

Parameter	Definition
deliveryId	The internal identifier of the vendor delivery header.

API: Confirm Delivery

Confirms the delivery and receives the goods into inventory.

Table 4-156 API Basics

Endpoint URL	/{deliveryId}/confirm
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Table 4-157 Path Parameter Definitions

Parameter	Definition
deliveryId	The internal identifier of the vendor delivery header.

API: Reject Delivery

Rejects the delivery without receiving goods, placing it in rejected status.

Table 4-158 API Basics

Endpoint URL	/{deliveryId}/reject
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous

Table 4-158 (Cont.) API Basics

Input	None
Output	None

Table 4-159 Path Parameter Definitions

Parameter	Definition
deliveryId	The internal identifier of the vendor delivery header.

API: Cancel Delivery

Cancels the delivery without receiving the goods and places it in canceled status.

Table 4-160 API Basics

Endpoint URL	/{deliveryId}/cancel
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Table 4-161 Path Parameter Definitions

Parameter	Definition
deliveryId	The internal identifier of the vendor delivery header.

API: Submit Carton

Moves the status of the carton to submitted and prevents further updates. The carton may still be confirmed. No inventory positions are updated via this operation.

Table 4-162 API Basics

Endpoint URL	cartons/{cartonId}/submit
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Table 4-163 Path Parameter Definitions

Parameter	Definition

Table 4-163 (Cont.) Path Parameter Definitions

cartonId	The internal identifier of the vendor delivery carton.

API: Cancel Submit Carton

Opens a submitted carton for further updates, moving the status to in-progress.

Table 4-164 API Basics

Endpoint URL	cartons/{cartonId}/cancelsubmit
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Table 4-165 Path Parameter Definitions

Parameter	Definition
cartonId	The internal identifier of the vendor delivery carton.

API: Confirm Carton

Confirms the final receipt of a vendor delivery carton.

Table 4-166 API Basics

Endpoint URL	cartons/{cartonId}/confirm
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Table 4-167 Path Parameter Definitions

Parameter	Definition
cartonId	The internal identifier of the vendor delivery carton.

API: Cancel Carton

Cancels a vendor delivery carton.

Table 4-168 API Basics

Endpoint URL	cartons/{cartonId}/cancel
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Table 4-169 Path Parameter Definitions

Parameter	Definition
cartonId	The internal identifier of the vendor delivery carton.

API: Open Carton

Re-open a completed carton after receipt allowing it to be received again.

Table 4-170 API Basics

Endpoint URL	cartons/{cartonId}/open
Method	POST
Successful Response	204 No Content
Processing Type	Synchronous
Input	None
Output	None

Table 4-171 Path Parameter Definitions

Parameter	Definition
cartonId	The internal identifier of the vendor delivery carton.

Additional Data Definitions

Table 4-172 Vendor Delivery Carton Status

Value	Status
1	New
2	In Progress
3	Submitted
4	Received
5	Damaged
6	Missing

Table 4-172 (Cont.) Vendor Delivery Carton Status

7	Canceled

Table 4-173 Customer Order Related Type

Value	Status
1	Yes
2	Mix
3	No

Table 4-174 Vendor Delivery Status

Value	Status
1	New
2	In Progress
3	Received
4	Canceled
5	Rejected

Table 4-175 Vendor Delivery Criteria Status

Value	Status
1	New
2	In Progress
3	Received
4	Canceled
5	Rejected
99	Active

Table 4-176 Vendor Delivery Origin Type

Value	Status
1	Advanced Shipping Notification
2	Purchase Order
3	Dex-Nex
4	Manual/On the Fly

Table 4-177 Vendor Delivery Create Origin Type

Value	Status
2	Purchase Order
3	Dex-Nex
4	Manual/On the Fly

Table 4-178 Carrier Type

Value	Status
1	Corporate
2	Third Party

Rest Service: Vendor Return

A return request is a return to vendor that comes into the store inventory system from an external system. A return request is a made for items to be returned to the supplier from the store. The items and quantities requested to be returned and reasons (for RTV) will be listed. Updating the return allows the approved quantity to be assigned. Once the request has been approved, the items are shipped to the supplier.

The service allows for the integration of return requests with an external system.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api

API Definitions

Table 4-179 Vendor Return API Definitions

API	Description
approveReturn	Approves a vendor return for shipment. Use the Update Return operation to assign and persist desired quantities before approval.
closeReturn	Closes out the vendor return.
findReturn	Finds up to 10,000 return headers for the input criteria.
importReturn	Imports a vendor return request managed by an external system.
importReturnDelete	This will place a delete notification in the system (in the MPS work queue with DcsRtv work type) for asynchronous processing.
readReturn	Retrieves all the details about a vendor return.
updateReturn	Update vendor return approved quantities prior to approving the vendor return.

API: Approve Return

Approves a vendor return for shipment. Use the Update Return operation to assign and persist desired quantities before approval.

Table 4-180 API Basics

Endpoint URL	/vendorreturns/{returnId}/approve
Method	POST
Successful Response	204 No Content
Processing Type	
Input	Path parameter
Output	None

Table 4-181 Path Parameter Definitions

Parameter	Definition
returnId	The unique identifier of the vendor return.

Table 4-182 Responses

Code	Description
204	No Content

400	Bad Request
	Example:
	
	[
	{
	"errors": [
	{
	"code": 1,
	"description": "AN_ERROR_TOOK_PLACE",
	"dataElement": "item",
	"dataValue": "1234",
	"referenceElement": "inventory adjustment",
	"referenceValue": "123"
	}
	]
	}
	]

API: Close Return

Closes out the vendor return..

Table 4-183 API Basics

Endpoint URL	/vendorreturns/{returnId}/close
Method	POST
Successful Response	204 No Content
Processing Type	

Table 4-183 (Cont.) API Basics

Input	Path parameter
Output	None

Table 4-184 Path Parameter Definitions

Parameter	Definition
returnId	The unique identifier of the vendor return.

Table 4-185 Responses

Code	Description
204	No Content

400	Bad Request
	Example:
	
	[
	{
	"errors": [
	{
	"code": 1,
	"description": "AN_ERROR_TOOK_PLACE",
	"dataElement": "item",
	"dataValue": "1234",
	"referenceElement": "inventory adjustment",
	"referenceValue": "123"
	}
	]
	}
	]

API: Find Return

Finds up to 10,000 return headers for the input criteria.

Table 4-186 API Basics

Endpoint URL	/vendorreturns
Method	Get
Successful Response	200 Successful
Processing Type	
Input	Query parameters
Output	List of returns
Max Response	10,000

Table 4-187 Query Parameter Definitions

Name	Format	Description
storeId	number(int10)	Include only records from this store.
supplierId	number(int10)	Include only returns to this supplier.
externalReturnId	string(text128)	Include only returns for this vendor return.
status	integer(int4)	Include only returns with this status. Valid values are: 1 - Requested 2 - Request In Progress 3 - RTV In Progress 4 - Approved 5 - In Shipping 6 - Completed 7 - Rejected 8 - Canceled Request 9 - Canceled RTV 99 - Active
itemId	string(text25)	Include only returns that contain this item.
shipmentReasonId	number(int15)	Include only returns that contain this shipment reason.
updateDateFrom	string(date-time)	Include only returns with an update date on or after this date.
updateDateTo	string(date-time	Include only returns with an update date on or before this date.

Table 4-188 Responses

Code	Description

200	Successful
	Example:
	
	[
	  {
	  "returnId": 11111,
	  "storeId": 5000,
	  "supplierId": 5101,
	  "externalReturnId": "90000",
	  "status": 1,
	  "notAfterDate": "2024-07-18T18:23:46.917Z",
	  "approvedDate": "2024-07-18T18:23:46.917Z",
	  "closedDate": "2024-07-18T18:23:46.917Z",
	  "createDate": "2024-07-18T18:23:46.917Z",
	  "updateDate": "2024-07-18T18:23:46.917Z"
	  }
	]

Table 4-188 (Cont.) Responses

400	Bad Request
	Example:
	
	[
	{
	"errors": [
	{
	"code": 1,
	"description": "AN_ERROR_TOOK_PLACE",
	"dataElement": "item",
	"dataValue": "1234",
	"referenceElement": "inventory adjustment",
	"referenceValue": "123"
	}
	]
	}
	]

API: Import Return

Imports a vendor return request managed by an external system. Processing assumes that any notification to other systems about the creation of this information will be managed by the external system. The imported return is processed asynchronous through our MPS sub-system and is controlled by the DcsRtv work type. If more than 1,000 items are included in the return, a "input too large" error will be returned. This operation will return a forbidden error code if the system is integrated with Oracle merchandising.

Table 4-189 API Basics

Endpoint URL	/vendorreturns/imports
Method	POST
Successful Response	202 Accepted
Processing Type	asynchronous
Input	Query parameters
Output	N/A
Max Response	1000

Table 4-190 Query Parameter Definitions

Parameter	Format	Required	Definition
externalReturnId	string(text128)	Yes	A unique identifer of the vendor return from an external system. This identifier is used to determine if the import will be created or updated.
storeId	number(int10)	Yes	The identifier of the store returning the inventory. This value is ignored if the return already exists.

Table 4-190 (Cont.) Query Parameter Definitions

supplierId	number(int10)	Yes	The identifier of the supplier the goods are being shipped to. This value is ignored if the return already exists.
notAfterDate	string(date-time)	No	A date after which the return should not be shipped. If not included, this value will be defaulted through system configuration.
authorizationCode	string(text12)	No	An authorization number that may be required for the return.
addressLine1	string(text240)	No	The first line of the destination address.
addressLine2	string(text240)	No	The second line of the destination address.
addressLine3	string(text240)	No	The third line of the destination address.
addressCity	string(text120	No	The city of the destination address.
addressState	string(text3)	No	The state of the destination address.
addressCountry	string(text3)	No	The country code of the destination address.
addressPostalCode	string(text30)	No	The postal code of the destination address.
comment	string(text2000)	No	A comment to associate to the vendor return request.
createDate	string(date-time)	Yes	The date the return was created.
lineItems	Array	Yes	Vendor Return Import Line Items itemId — the unique identifier of the SKU item. format: string(text25), required reasonCode — a reason code associated to the item return. format:string(text6), required quantity — the quantity requested to return. format: BigDecimal(decimal (20,4)), required sequence — An external sequence number for the item on the return. format: BigDecimal(int15), optional

Table 4-191 Responses

Code	Description
202	Accepted

Table 4-191 (Cont.) Responses

400	Bad Request
	Example:
	
	[
	{
	"errors": [
	{
	"code": 1,
	"description": "AN_ERROR_TOOK_PLACE",
	"dataElement": "item",
	"dataValue": "1234",
	"referenceElement": "inventory adjustment",
	"referenceValue": "123"
	}
	]
	}
	]

API: Import Return Delete

This will place a delete notification in the system (in the MPS work queue with DcsRtv work type) for asynchronous processing. When processed, it will cancel the vendor return if it qualifies for cancellation (it is not already in progress). This operation will return a forbidden error code if the system is integrated with Oracle merchandising.

Table 4-192 API Basics

Endpoint URL	/vendorreturns/imports
Method	DELETE
Successful Response	202 Accepted
Processing Type	asynchronous
Input	Path parameters
Output	N/A
Max Response	1000

Table 4-193 Path Parameters

Parameter	Format	Required	Definition
externalReturnId	string(text128)	Yes	A unique identifer of the vendor return from an external system.

Table 4-194 Responses

Code	Description
202	Accepted

Table 4-194 (Cont.) Responses

400	Bad Request
	Example:
	
	[
	{
	"errors": [
	{
	"code": 1,
	"description": "AN_ERROR_TOOK_PLACE",
	"dataElement": "item",
	"dataValue": "1234",
	"referenceElement": "inventory adjustment",
	"referenceValue": "123"
	}
	]
	}
	]

API: Read Return

Retrieves all the details about a vendor return.

Table 4-195 API Basics

Endpoint URL	/vendorreturns/{returnId}
Method	Get
Successful Response	200 Successful
Processing Type	
Input	Path parameters
Output	return information
Max Response	10,000

Table 4-196 Path Parameters

Name	Format	Required	Description
returnId	number(int12)	Yes	The unique identifier of the vendor return

Table 4-197 Responses

Code	Description

Table 4-197 (Cont.) Responses

200	Successful
	Example:
	
	[
	  {
	  "returnId": 11111,
	  "storeId": 5000,
	  "supplierId": 5101,
	  "externalReturnId": "90000",
	  "externalLocked": true,
	  "originType": 1,
	  "status": 1,
	  "notAfterDate": "2024-07-18T21:08:37.871Z",
	  "authorizationCode": "JID112",
	  "addressLine1": "460 Summer Street",
	  "addressLine2": "Apartment 2",
	  "addressLine3": "c/o John Smith",
	  "aAddressCity": "Coolsville",
	  "addressState": "NY",
	  "addressCountry": "US",
	  "addressPostalCode": "55555-1111",
	  "approvedUser": "smith123",
	  "approvedDate": "2024-07-18T21:08:37.871Z",
	  "closedUser": "smith123",
	  "closedDate": "2024-07-18T21:08:37.871Z",
	  "createUser": "smith123",
	  "createDate": "2024-07-18T21:08:37.871Z",
	  "updateUser": "smith123",
	  "updateDate": "2024-07-18T21:08:37.871Z",
	  "lineItems": [
	  {
	  "lineId": 2222222,
	  "externalLineId": 456,
	  "itemId": "10045600",
	  "shipmentReasonId": 82236,
	  "caseSize": 3,
	  "quantityRequested": 100,
	  "quantityApproved": 100,
	  "quantityShipping": 100,
	  "quantityShipped": 100
	  }
	  ]
	  }
	]

Table 4-197 (Cont.) Responses

400	Bad Request
	Example:
	
	[
	{
	"errors": [
	{
	"code": 1,
	"description": "AN_ERROR_TOOK_PLACE",
	"dataElement": "item",
	"dataValue": "1234",
	"referenceElement": "inventory adjustment",
	"referenceValue": "123"
	}
	]
	}
	]

API: Update Return

Update vendor return approved quantities prior to approving the vendor return.

Table 4-198 API Basics

Endpoint URL	/vendorreturns/{returnId}/
Method	POST
Successful Response	204 No Content
Processing Type	
Input	Path parameter, Request
Output	N/A

Table 4-199 Path Parameter

Parameter	Format	Required	Definition
returnId	number(int12)	Yes	The unique identifier of the vendor return.

Request

in the Request body, enter the vendor return details to update.

Example:

```
{
  "authorizationCode": "JID112",
  "lineItems": [
    {
      "itemId": "10045600",
```

```
      "shipmentReasonId": 82236,
      "quantity": 100
    }
  ]
}
```

Table 4-200 Responses

Code	Description
204	No Content

400	Bad Request
	Example:
	
	[
	  {
	  "errors": [
	  {
	  "code": 1,
	  "description": "AN_ERROR_TOOK_PLACE",
	  "dataElement": "item",
	  "dataValue": "1234",
	  "referenceElement": "inventory adjustment",
	  "referenceValue": "123"
	  }
	  ]
	  }
	]

Vendor Shipment

Vendor shipment is the process of shipping inventory out of the store back to a supplier for redistribution or disposal. If the store plans to return items to the vendor, it can be done by creating a vendor shipment (RTV shipment). The items, quantities to be returned, and reasons for the return, can be added to the shipment. When the shipment is dispatched, the goods are removed from inventory.

To create a shipment, a supplier must be identified for this return. The supplier must allow for returns and must be a DSD supplier, and if the system is configured to use multiple sets of books, then the supplier is limited to the same operating unit as that of the store.

Items along with reasons can be added to the return. For Return to Vendors, the system allows a mix of available and unavailable inventory status items on the return; therefore, all reasons for the Return to Vendor type will be listed. All items on the return must be sourced by the supplier designated on the RTV. Authorization Number, which is required for certain suppliers, would need to be entered before dispatching.

The service defines operations to manage vendor shipment information by integration with an external system.

Service Base URL

The Cloud service base URL follows the format:

```
https://<external_load_balancer>/<cust_env>/siocs-int-services/api
```

Find Shipments — GET

This section describes the Find Shipments API. This API finds up to 10,000 shipments headers for the input criteria.

Method

```
GET
```

URL

```
/rtvshipments
```

Request

This section describes the request parameters.

Table 4-201 Find Shipments— Request Parameters

Parameter	Required	Data Type	Description
storeId	NA	number ($int10) (query)	Include only records where the shipment is from this store.
supplierId	NA	number ($int10) (query)	Include only shipments where the shipment is to this supplier.
vendorReturnId	NA	number ($int15) (query)	Include only shipments for this vendor return (by internal identifier).
vendorReturnExternalId	NA	string ($text128) (query)	Include only shipments for this vendor return (by external identifier).
status	NA	integer ($int4) (query)	Include only shipments with this delivery status. Valid values are 1 – New 2 - In Progress 3 – Submitted 4 – Shipped 5 – Canceled 99 – Active *Available values:* 1, 2, 3, 4, 5, 99
itemId	NA	string ($text128) (query)	Include only shipments that contain this item.
updateDateFrom	NA	string ($date-time) (query)	Include only shipments with an update date on or after this date.

Table 4-201 (Cont.) Find Shipments— Request Parameters

Parameter	Required	Data Type	Description
updateDateTo	NA	string ($date-time) (query)	Include only shipments with an update date on or before this date.

Responses

This section describes the responses of the Find Shipments API.

Response Code: 200

The request has been successful.

The media type is application/json.

Example Value


```
[
  {
    "shipmentId": 11111,
    "storeId": 5000,
    "supplierId": 5101,
    "vendorReturnId": 789012,
    "vendorReturnExternalId": "90000",
    "status": 1,
    "notAfterDate": "2024-08-30T07:40:22.421Z",
    "submitDate": "2024-08-30T07:40:22.421Z",
    "dispatchDate": "2024-08-30T07:40:22.422Z",
    "updateDate": "2024-08-30T07:40:22.422Z"
  }
]
```


Schema — VendorShipmentHeaderIdo

This section describes the VendorShipmentHeaderIdo schema.

Table 4-202 VendorShipmentHeaderIdo — Object

Element Name	Required	Data Type	Description
shipmentId	NA	number($int12) example: 11111	The unique identifier of the vendor shipment.
storeId	NA	number($int10) example: 5000	The identifier of the store shipping the inventory.
supplierId	NA	number($int10) example: 5101	The identifier of the supplier the goods are being shipped to.
vendorReturnId	NA	number($int15) example: 789012	The unique identifier of the vendor return this shipment is for.

Table 4-202 (Cont.) VendorShipmentHeaderIdo — Object

Element Name	Required	Data Type	Description
vendorReturnExternalId	NA	string($text128) example: 90000	A unique identifer of the vendor return from an external system.
status	NA	integer($int4) example: 1	The current status of the vendor shipment. Valid values are: 1 – New 2 - In Progress 3 – Submitted 4 – Shipped 5 – Canceled Enum: [1, 2, 3, 4, 5]
notAfterDate	NA	string($date-time)	A date after which the shipment should not be shipped.
submitDate	NA	string($date-time)	The date the shipment was submitted.
dispatchDate	NA	string($date-time)	The date the shipment was dispatched.
updateDate	NA	string($date-time)	The date the delivery was last updated.

Response Code: 400

Bad request

The media type is application/json.

See the JSON Error Element and Error Codes section.

Create Shipment — POST

This section describes the Create Shipment API. It creates a new vendor shipment whose status is in progress. When creating the vendor shipment, it only allows up to 1,000 items in the overall shipment.

Method

```
POST
```

URL

```
/rtvshipments
```

Request

There are no request parameters.

The request body is application/json.

The shipment details of the new shipment.

Example Value


```
{
  "storeId": 5000,
  "supplierId": 5101,
  "supplierCountryCode": "USA",
  "vendorReturnId": 789012,
  "authorizationCode": "H112",
  "notAfterDate": "2024-08-30T07:55:27.191Z",
  "contextId": 90000,
  "contextValue": "Spring Return",
  "addressType": 1,
  "carrierRole": 1,
  "carrierId": 123,
  "carrierServiceId": 456,
  "alternateAddress": "Another Street, Different City, NY, 44444",
  "alternateCarrierName": "Speedy Delivery",
  "alternateCarrierAddress": "Carrier Street, Carrier City, NY, 44444",
  "motiveId": 123,
  "taxId": "11-33500",
  "trackingNumber": "44000567002",
  "dimensionId": 100,
  "weight": 12.8,
  "weightUom": "KG",
  "requestedPickupDate": "2024-08-30T07:55:27.191Z",
  "cartons": [
    {
      "externalCartonId": "4500300",
      "trackingNumber": "876300456",
      "restrictionLevel": 1,
      "cartonSizeId": 1000,
      "weight": 4.5,
      "weightUom": "LBS",
      "lineItems": [
        {
          "itemId": "10045600",
          "reasonId": 2222222,
          "caseSize": 3,
          "quantity": 100,
          "uins": [
            111345000,
            222220000
          ]
        }
      ]
    }
  ],
  "notes": [
    "The first note",
    "The second note"
  ]
}
```

Schema — VendorShipmentCreateIdo

This section describes the VendorShipmentCreateIdo schema.

Table 4-203 VendorShipmentCreateIdo — Object

Element Name	Required	Data Type/ Example	Description
storeId	Yes	number($int10) example: 5000	The identifier of the store shipping the inventory.
supplierId	Yes	number($int10) example: 5101	The identifier of the supplier the goods are being shipped to.
supplierCountryCode	NA	string($text3) example: USA	The country code of the supplier for this shipment. All items must be available with this country code. If left blank, a country code will be selected from the available supplier items giving preference to a country code that matches the store.
vendorReturnId	NA	number($int15) example: 789012	The unique identifier of the vendor return this shipment is for.
authorizationCode	NA	string($text12) example: H112	A vendor authorization code
notAfterDate	NA	string($date-time)	A date after which the shipment should not be shipped.
contextId	NA	number($int18) example: 90000	A unique identifer of a context for the shipment (see shipping service).
contextValue	NA	string($text25) example: Spring Return	An additional value associated to the context for the shipment.
addressType	NA	integer($int4) example: 1	The hierarchy restriction level for items in the container. Valid values are: 1 - Department 2 - Class 3 - Subclass 4 - None Enum: [1, 2, 3, 4
carrierRole	Yes	integer($int2) example: 1	The type of carrier for the shipment. Valid values are: 1 - Sender 2 - Receiver 3 - Third Party Enum: [1, 2, 3]
carrierId	NA	integer($int10) example: 123	A unique identifier of a carrier for the shipment.
carrierServiceId	NA	integer($int10) example: 456	A unique identifier of a carrier service for the shipment.
alternateAddress	NA	string($text2000) example: Another Street, Different City, NY, 44444	An alternate destination address.

Table 4-203 (Cont.) VendorShipmentCreateIdo — Object

Element Name	Required	Data Type/ Example	Description
alternateCarrierName	NA	string($text240) example: Speedy Delivery	The name of a third-party shipping company.
alternateCarrierAddress	NA	string($text240) example: Carrier Street, Carrier City, NY, 44444	The address of a third-party shipping company.
motiveId	NA	number($int18) example: 123	The identifier ofa motive for the bill of lading.
taxId	NA	string($text18) example: 11-33500	The tax identifier of the supplier it is being shipped to.
trackingNumber	NA	string($text128) example: 44000567002	A tracking number associated to the shipment.
dimensionId	NA	integer($int12) example: 100	The identifier of a dimension object associated to the shipment.
weight	NA	number($decimal(12,4)) example: 12.8	The weight of the shipment.
weightUom	NA	string($text4) example: KG	The unit of measure of the weight of the shipment.
requestedPickupDate	NA	string($date-time)	The requested pickup date.
cartons	Yes	object	cartons
notes	NA	string ($text2000) example: List ["The first note", "The second note"]	A collection of up to 100 notes to be created.

Table 4-204 VendorShipmentCreateCartonIdo — Object

Element Name	Required	Data Type/Example	Description
externalCartonId	NA	string($text128) example: 4500300	A container identifier from an external system.
trackingNumber	NA	string($text128) example: 876300456	The tracking number of the container.
restrictionLevel	NA	integer($int4)	example: 1 The hierarchy restriction level for items in the container. Valid values are: 1 - Department 2 - Class 3 - Subclass 4 - None Enum: [1, 2, 3, 4]

Table 4-204 (Cont.) VendorShipmentCreateCartonIdo — Object

Element Name	Required	Data Type/Example	Description
cartonSizeId	NA	integer($int10) example: 1000	The unique identifier of the container dimension object.
weight	NA	number($decimal(12,4)) example: 4.5	The weight of the container.
weightUom	NA	string($text4) example: LBS	The unit of measure of the weight of the container.
lineItems	Yes	object	line items

Table 4-205 VendorShipmentCreateLineItemIdo — Object

Element Name	Required	Data Type/Example	Description
itemId	Yes	string($text25) example: 10045600	The unique identifier of the SKU item.
reasonId	Yes	integer($int15) example: 2222222	The unique identier of the shipment reason.
caseSize	NA	number($decimal(10,2)) example: 3	The case size of this record.
quantity	Yes	number($decimal(20,4)) example: 100	The quantity that was shipped.
uins	NA	string($text128)] example: List [111345000, 222220000]	The UINs that were shipped. They total number must match the quantity shipped.

Responses

This section describes the responses of the Create Shipment API.

Response Code: 200

Successful response.

The media type is application/json.

Example Value

```
[
  {
    "shipmentId": <shipmentId>,
    "returnId": <returnId>,
    "status": 1
  }
]
```

Schema — VendorShipmentStatusIdo

This section describes the VendorShipmentStatusIdo schema.

Table 4-206 VenderoShipmentStatusIdo

Element Name	Required	Data Type/ Example	Description
shipmentId	NA	number($int15) example: 11111	The unique identifier of the vendor shipment.
returnId	NA	number($int15) example: 5000	The identifier of the associated vendor return.
status	NA	integer($int4) example: 1	The current status of the vendor shipment. Valid values: 1 - New 2 - In Progress 3 - Submitted 4 - Shipped 5 - Canceled Enum: [1, 2, 3, 4, 5]

Response Code: 400

Bad request

The media type is application/json.

See the JSON Error Element and Error Codes section.

Read Shipment — GET

This section describes the Read Shipment API. This API retrieves all the details about an vendor shipment.

Method

```
GET
```

URL

```
/rtvshipments/{shipmentId}
```

Request

This section describes the request parameters.

Table 4-207 Read Shipment — Request Parameters

Parameters	Required	Data Type	Description
shipmentId	Yes	number($int12) (path)	The unique identifier of the vendor shipment.

Responses

This section describes the responses of the Read Shipment API.

Response Code: 200

Successful response

This media type is application/json.

Example Value


```
[
  {
    "shipmentId": <shipmentId>,
    "storeId": 5000,
    "supplierId": 5101,
    "vendorReturnId": 789012,
    "vendorReturnExternalId": "90000",
    "status": 1,
    "notAfterDate": "2024-08-28T09:29:33.327Z",
    "authorizationCode": "H112",
    "contextId": 90000,
    "contextValue": "Spring Return",
    "destinationAddress1": "460 Summer Street",
    "destinationAddress2": "Apartment 2",
    "destinationAddress3": "c/o John Smith",
    "destinationAddressCity": "Coolsville",
    "destinationAddressState": "NY",
    "destinationAddressCountry": "US",
    "destinationAddressPostalCode": "55555-1111",
    "billOfLadingId": 55601,
    "alternateAddress": "Another Street, Different City, NY, 44444",
    "carrierRole": 1,
    "carrierId": 123,
    "carrierServiceId": 456,
    "alternateCarrierName": "Speedy Delivery",
    "alternateCarrierAddress": "Carrier Street, Carrier City, NY, 44444",
    "motive": "Overstock",
    "taxId": "11-33500",
    "trackingNumber": "44000567002",
    "dimensionId": 100,
    "weight": 12.8,
    "weightUom": "KG",
    "requestedPickupDate": "2024-08-28T09:29:33.327Z",
    "fiscalDocumentRequestId": 22233344455566,
```

```
    "fiscalDocumentReferenceId": 77788899100,
    "fiscalDocumentNumber": "128745",
    "fiscalDocumentStatus": 1,
    "fiscalDocumentRejectReason": "Missing Requirements",
    "fiscalDocumentUrl": "http://fiscaldoc/doc123",
    "createUser": "smith123",
    "createDate": "2024-08-28T09:29:33.327Z",
    "submitUser": "smith123",
    "submitDate": "2024-08-28T09:29:33.327Z",
    "dispatchUser": "smith123",
    "dispatchDate": "2024-08-28T09:29:33.327Z",
    "updateUser": "smith123",
    "updateDate": "2024-08-28T09:29:33.327Z",
    "cartons": [
      {
        "cartonId": 100200300,
        "externalCartonId": "4500300",
        "status": 1,
        "cartonDimensionId": 1000,
        "weight": 4.5,
        "weightUom": "LBS",
        "trackingNumber": "876300456",
        "restrictionLevel": 1,
        "createDate": "2024-08-28T09:29:33.328Z",
        "createUser": "smith123",
        "updateDate": "2024-08-28T09:29:33.328Z",
        "updateUser": "smith123",
        "approvalDate": "2024-08-28T09:29:33.328Z",
        "approvalUser": "smith123",
        "lineItems": [
          {
            "lineId": 2222222,
            "itemId": "10045600",
            "shipmentReasonId": 2222222,
            "caseSize": 3,
            "quantity": 100,
            "uins": [
              111345000,
              222220000
            ]
          }
        ]
      }
    ]
  }
]
```


Schema — VendorShipmentIdo

This section describes the VendorShipmentIdo schema.

Table 4-208 VendorShipmentIdo — Object

Element Name	Required	Data Type/Example	Description
shipmentId	NA	number($int12) example: 11111	The unique identifier of the vendor shipment.
storeId	NA	number($int10) example: 5000	The identifier of the store shipping the inventory.
supplierId	NA	number($int10) example: 5101	The identifier of the supplier the goods are being shipped to.
vendorReturnId	NA	number($int15) example: 789012	The unique identifier of the vendor return this shipment is for.
vendorReturnExternalId	NA	string($text128) example: 90000	A unique identifer of the vendor return from an external system.
status	NA	integer($int4) example: 1	The current status of the vendor shipment. Valid values are 1 - New 2 - In Progress 3 - Submitted 4 - Shipped 5 - Canceled Enum: [1, 2, 3, 4, 5]
notAfterDate	NA	string($date-time)	A date after which the shipment should not be shipped.
authorizationCode	NA	string($text12) example: H112	A vendor authorization code
contextId	NA	number($int18) example: 90000	A unique identifer of a context for the shipment (see shipping service).
contextValue	NA	string($text25) example: Spring Return	An additional value associated to the context for the shipment.
destinationAddress1	NA	string($text240) example: 460 Summer Street	The first line of the destination address.
destinationAddress2	NA	string($text240) example: Apartment 2	The second line of the destination address.
destinationAddress3	NA	string($text240) example: c/o John Smith	The third line of the destination address.
destinationAddressCity	NA	string($text120) example: Coolsville	The city of the destination address.
destinationAddressState	NA	string($text3) example: NY	The state of the destination address.
destinationAddressCountry	NA	string($text3) example: US	The country code of the destination address.
destinationAddressPostalCode	NA	string($text30) example: 55555-1111	The postal code of the destination address.
billOfLadingId	NA	number($int15) example: 55601	A bill of lading identifier.

Table 4-208 (Cont.) VendorShipmentIdo — Object

Element Name	Required	Data Type/Example	Description
alternateAddress	NA	string($text2000) example: Another Street, Different City, NY, 44444	An alternate destination address.
carrierRole	NA	integer($int2) example: 1	The type of carrier for the shipment. Valid values are 1 - Sender 2 - Receiver 3 - Third Party Enum: [1, 2, 3]
carrierId	NA	integer($int10) example: 123	A unique identifier of a carrier for the shipment.
carrierServiceId	NA	integer($int10) example: 456	A unique identifier of a carrier service for the shipment.
alternateCarrierName	NA	string($text240) example: Speedy Delivery	The name of a third-party shipping company.
alternateCarrierAddress	NA	string($text240) example: Carrier Street, Carrier City, NY, 44444	The address of a third-party shipping company.
motive	NA	string($text120) example: Overstock	A motive for the shipment.
taxId	NA	string($text18) example: 11-33500	The tax identifier of the supplier it is being shipped to.
trackingNumber	NA	string($text128) example: 44000567002	A tracking number associated to the shipment.
dimensionId	NA	integer($int12) example: 100	The identifier of a dimension object associated to the shipment.
weight	NA	number($decimal(12,4)) example: 12.8	The weight of the shipment.
weightUom	NA	string($text4) example: KG	The unit of measure of the weight of the shipment.
requestedPickupDate	NA	string($date-time)	The requested pickup date.
fiscalDocumentRequestId	NA	number($int20) example: 22233344455566	The identifier of the request for a fiscal document.
fiscalDocumentReferenceId	NA	number($int20) example: 77788899100	The unique identifier of the fiscal document
fiscalDocumentNumber	NA	string($text255) example: 128745	The fiscal document number.

Table 4-208 (Cont.) VendorShipmentIdo — Object

Element Name	Required	Data Type/Example	Description
fiscalDocumentStatus	NA	integer($int4) example: 1	The status of the fiscal document. Valid values are 1 - Approved 2 - Submitted 3 - Rejected 4 - Canceled Enum: [1, 2, 3, 4]
fiscalDocumentRejectReason	NA	string($text255) example: Missing Requirements	A reason the fiscal document was rejected.
fiscalDocumentUrl	NA	string($text255) example:	A URL to the fiscal document.
createUser	NA	string($text128) example: smith123	The user that created the shipment.
createDate	NA	string($date-time)	The date the shipment was created.
submitUser	NA	string($text128) example: smith123	The user that submitted the shipment.
submitDate	NA	string($date-time)	The date the shipment was submitted.
dispatchUser	NA	string($text128) example: smith123	The user that dispatched the shipment.
dispatchDate	NA	string($date-time)	The date the shipment was dispatched.
updateUser	NA	string($text128) example: smith123	The user that last updated the shipment.
updateDate	NA	string($date-time)	The date the delivery was last updated.
cartons	NA	object	cartons

Table 4-209 VendorShipmentCartonIdo — Object

Element Name	Required	Data Type/Example	Description
cartonId	NA	integer($int15) example: 100200300	The unique identifier of the carton record.
externalCartonId	NA	string($text128) example: 4500300	A container identifier from an external system.
status	NA	integer($int4) example: 1	The current status of the carton. Valid values are 1 - New 2 - In Progress 3 - Completed 4 - Shipped 5 - Canceled Enum: [1, 2, 3, 4, 5]
cartonDimensionId	NA	integer($int10) example: 1000	The unique identifier of the container dimension object.

Table 4-209 (Cont.) VendorShipmentCartonIdo — Object

Element Name	Required	Data Type/Example	Description
weight	NA	number($decimal(12,4)) example: 4.5	The weight of the container.
weightUom	NA	string($text4) example: LBS	The unit of measure of the weight of the container.
trackingNumber	NA	string($text128) example: 876300456	The tracking number of the container.
restrictionLevel	NA	integer($int4) example: 1	The hierarchy restriction level for items in the container. Valid values are1 - Department 2 - Class 3 - Subclass 4 - None Enum: [1, 2, 3, 4]
createDate	NA	string($date-time)	The date the carton was created.
createUser	NA	string($text128) example: smith123	The user that created the carton.
updateDate	NA	string($date-time)	The date the carton was last updated.
updateUser	NA	string($text128) example: smith123	The user that last updated the carton.
approvalDate	NA	string($date-time)	The date the carton was approved.
approvalUser	NA	string($text128) example: smith123	The user that approved the carton.
lineItems	NA	object	line items

Table 4-210 VendorShipmentLineItemIdo — Object

Element Name	Required	Data Type/Example	Description
lineId	NA	integer($int15) example: 2222222	The unique identier of the line record.
itemId	NA	string($text25) example: 10045600	The unique identifier of the SKU item.
shipmentReasonId	NA	integer($int15) example: 2222222	The unique identier of the shipment reason.
caseSize	NA	number($decimal(10,2)) example: 3	The case size of this record.
quantity	NA	number($decimal(20,4)) example: 100	The quantity that was shipped.
uins	Na	string($text128) example: List [111345000, 222220000]	The UINs that were shipped. They total number must match the quantity shipped.

Response Code: 400

Bad request

The media type is application/json.

See the JSON Error Element and Error Codes section.

Update Shipment — POST

This section describes the Update Shipment API. This API updates the header portion of a shipment as well as its bill of lading information. See carton related operations for creating or updating cartons on the shipment.

Method

```
POST
```

URL

```
/rtvshipments/{shipmentId}
```

Request

This section describes the request parameters.

Table 4-211 Update Shipment — Request Parameters

Parameters	Required	Data Type	Description
shipmentId	Yes	number($int12) (path)	The unique identifier of the vendor shipment.

The request body is application/json.

Shipment header details to update

Example Value

```
{
  "notAfterDate": "2024-08-28T12:54:48.823Z",
  "authorizationCode": "H112",
  "contextId": 90000,
  "contextValue": "Spring Return",
  "addressType": 1,
  "carrierRole": 1,
  "carrierId": 123,
  "carrierServiceId": 456,
  "alternateAddress": "Another Street, Different City, NY, 44444",
  "alternateCarrierName": "Speedy Delivery",
  "alternateCarrierAddress": "Carrier Street, Carrier City, NY, 44444",
  "motiveId": 123,
  "taxId": "11-33500",
  "trackingNumber": "44000567002",
```

```
  "dimensionId": 100,
  "weight": 20.4,
  "weightUom": "KG",
  "requestedPickupDate": "2024-08-28T12:54:48.823Z",
  "notes": [
    "The first note",
    "The second note"
  ]
}
```


Schema — VendorShipmentUpdateIdo

This section describes the VendorShipmentUpdateIdo schema.

Table 4-212 VendorShipmentUpdateIdo — Object

Element Name	Required	Data Type/ Example	Description
notAfterDate	NA	string($date-time)	A date after which the shipment should not be shipped.
authorizationCode	NA	string($text12) example: H112	A vendor authorization code
contextId	NA	number($int18) example: 90000	A unique identifer of a context for the shipment (see shipping service).
contextValue	NA	string($text25) example: Spring Return	An additional value associated to the context for the shipment.
addressType	NA	integer($int4) example: 1	The hierarchy restriction level for items in the container. Valid values are 1 - Department 2 - Class 3 - Subclass 4 - None Enum: [1, 2, 3, 4]
carrierRole	Yes	integer($int2) example: 1	The type of carrier for the shipment. Valid values are 1 - Sender 2 - Receiver 3 - Third Party Enum: [1, 2, 3]
carrierId	NA	integer($int10) example: 123	A unique identifier of a carrier for the shipment.
carrierServiceId	NA	integer($int10) example: 456	A unique identifier of a carrier service for the shipment.
alternateAddress	NA	string($text2000) example: Another Street, Different City, NY, 44444	An alternate destination address.
alternateCarrierName	NA	string($text240) example: Speedy Delivery	The name of a third-party shipping company.

Table 4-212 (Cont.) VendorShipmentUpdateIdo — Object

Element Name	Required	Data Type/ Example	Description
alternateCarrierAddress	NA	string($text240) example: Carrier Street, Carrier City, NY, 44444	The address of a third-party shipping company.
motiveId	NA	number($int18) example: 123	The unique identifier of a mtoive for the bill of lading.
taxId	NA	string($text18) example: 11-33500	The tax identifier of the supplier it is being shipped to.
trackingNumber	NA	string($text128) example: 44000567002	A tracking number associated to the shipment.
dimensionId	NA	integer($int12) example: 100	The idnetifier of a dimension object associated to the shipment.
weight	NA	number($decimal(12,4)) example: 20.4	The weight of the shipment.
weightUom	NA	string($text4) example: KG	The unit of measure of the weight of the shipment.
requestedPickupDate	NA	string($date-time)	The requested pickup date.
notes	NA	string($text2000) example: List ["The first note", "The second note"]	A collection of up to 100 notes to be added to the list of notes.

Responses

This section describes the responses of the Update Shipment API.

Response Code: 204

No Content

Response Code: 400

Bad request

The media type is application/json.

See the JSON Error Element and Error Codes section.

Submit Shipment — POST

This section describes the Submit Shipment API. This API submits a vendor shipment placing it into submitted status awaiting review and eventual dispatch. Fiscal document data capture can occur as part of this process.

Method

```
POST
```

URL

```
/rtvshipments/{shipmentId}/submit
```

Request

This section describes the request paramters.

Table 4-213 Submit Shipment — Request Parameters

Parameters	Required	Data Type	Description
shipmentId	Yes	number($int12) (path)	The unique identifier of the vendor shipment.

The request body is application/json.

Fiscal document information to include as part of the submission.

Example Value

```
{
  "vehicleNumber": "ABC-222",
  "vehicleStateOrCountry": "New York",
  "driverName": "John Smith",
  "driverLicenseNumber": "HBAC-123456"
}
```

Schema — VendorShipmentFiscalSubmitIdo

This section describes the VendorShipmentFiscalSubmitIdo schema.

Table 4-214 VendorShipmentFiscalSubmitIdo — Object

Element Name	Required	Data Type/ Example	Description
vehicleNumber	NA	string($text25) example: ABC-222	The vehicle license plate or identifying number.

Table 4-214 (Cont.) VendorShipmentFiscalSubmitIdo — Object

Element Name	Required	Data Type/ Example	Description
vehicleStateOrCountry	NA	string($text25) example: New York	The vehicle's state or country.
driverName	NA	string($text30) example: John Smith	The name of the driver.
driverLicenseNumber	NA	string($text30) example: HBAC-123456	The driver's liscence number.

Responses

This section describes the responses of the Submit Shipment API.

Response Code: 204

No Content

Response Code: 400

Bad request

The media type is application/json.

See the JSON Error Element and Error Codes section.

Cancel Submit Shipment — POST

This section describes the Cancel Submit Shipment API. Cancels the submission of a shipment moving it back to in progress state.

Method

```
POST
```

URL

```
/rtvshipments/{shipmentId}/cancelSubmit
```

Request

This section describes the request parameters.

Table 4-215 Cancel Submit Shipment — Request Parameters

Parameters	Required	Data Type	Description
shipmentId	Yes	number($int12) (path)	The unique identifier of the vendor shipment.

Responses

This section describes the responses of the Cancel Submit Shipment API.

Response Code: 204

No Content

Response Code: 400

Bad request

The media type is application/json.

See the JSON Error Element and Error Codes section.

Dispatch Shipment — POST

This section describes the Dispatch Shipment API. This API dispatches the shipment to the destination, closing the shipment, and updating inventory.

Method

```
POST
```

URL

```
/rtvshipments/{shipmentId}/dispatch
```

Request

This section describes the request parameters.

Table 4-216 Dispatch Shipment — Request Parameters

Parameters	Required	Data Type	Description
shipmentId	Yes	number($int12) (path)	The unique identifier of the vendor shipment.

Responses

This section describes the responses of the Dispatch Shipment API.

Response Code: 204

No Content

Response Code: 400

Bad request

The media type is application/json.

See the JSON Error Element and Error Codes section.

Cancel Shipment — POST

This section describes the Cancel Shipment API. This API cancels the shipment. This will close the associated vendor return.

Method

```
POST
```

URL

```
/rtvshipments/{shipmentId}/cancel
```

Request

This section describes the request parameters.

Table 4-217 Cancel Shipment — Request Parameters

Parameters	Required	Data Type	Description
shipmentId	Yes	number($int12) (path)	The unique identifier of the vendor shipment.

Responses

This section describes the responses of the Cancel Shipment API.

Response Code: 204

No Content

Response Code: 400

Bad request

The media type is application/json.

See the JSON Error Element and Error Codes section.

Create Carton — POST

This section describes the Create Carton API. This API adds a new container to a new or in progress shipment.

Method

```
POST
```

URL

```
/rtvshipments/{shipmentId}/cartons
```

Request

This section describes the request parameters.

Table 4-218 Create Carton — Request Parameters

Parameters	Required	Data Type	Description
shipmentId	Yes	number($int12) (path)	The unique identifier of the shipment.

The request body is applicaton/json.

Details for the carton to create.

Example Value

```
{
  "externalCartonId": "4500300",
  "trackingNumber": "876300456",
  "restrictionLevel": 1,
  "cartonSizeId": 1000,
  "weight": 4.5,
  "weightUom": "LBS",
  "lineItems": [
    {
```

```
      "itemId": "10045600",
      "reasonId": 2222222,
      "caseSize": 3,
      "quantity": 100,
      "uins": [
        111345000,
        222220000
      ]
    }
  ]
}
```

Schema — VendorShipmentCartonCreateIdo

This section describes the VendorShipmentCartonCreateIdo schema.

Table 4-219 VendorShipmentCartonCreateIdo — Object

Element Name	Required	Data Type/ Example	Description
externalCartonId	NA	string($text128) example: 4500300	A container identifier from an external system.
trackingNumber	NA	string($text128) example: 876300456	The tracking number of the container.
restrictionLevel	NA	integer($int4) example: 1	The hierarchy restriction level for items in the container. Valid values are 1 - Department 2 - Class 3 - Subclass 4 - None Enum: [1, 2, 3, 4]
cartonSizeId	NA	integer($int10) example: 1000	The unique identifier of the container dimension object.
weight	NA	number($decimal(12,4)) example: 4.5	The weight of the container.
weightUom	NA	string($text4) example: LBS	The unit of measure of the weight of the container.
lineItems	Yes	object	line items

Table 4-220 VendorShipmentCartonCreateLineItemIdo — Object

Element Name	Required	Data Type/ Example	Description
itemId	Yes	string($text25) example: 10045600	The unique identifier of the SKU item.

Table 4-220 (Cont.) VendorShipmentCartonCreateLineItemIdo — Object

Element Name	Required	Data Type/ Example	Description
reasonId	Yes	integer($int15) example: 2222222	The unique identifier of the shipment reason.
caseSize	NA	number($decimal(10,2)) example: 3	The case size of this record.
quantity	Yes	number($decimal(20,4)) example: 100	The quantity that was shipped.
uins	NA	string($text128 example: List [111345000, 222220000]	The UINs that were shipped. The total number must match the quantity shipped.

Responses

This section describes the responses of the Create Carton API.

Reason Code: 200

Successful response

The media type is application/json.

Example Value

```
[
  {
    "shipmentId": 11111,
    "cartonId": 5000,
    "status": 1
  }
]
```

Schema — VendorShipmentCartonStatusIdo

This section describes the VendorShipmentCartonStatusIdo schema.

Table 4-221 VendorShipmentCartonStatusIdo — Object

Element Name	Required	Data Type/ Example	Description
shipmentId	NA	number($int15) example: 11111	The unique identifier of the vendor shipment.

Table 4-221 (Cont.) VendorShipmentCartonStatusIdo — Object

Element Name	Required	Data Type/ Example	Description
cartonId	NA	number($int15) example: 5000	The identifier of the new carton.
status	NA	integer($int4) example: 1	The current status of the vendor shipment. Valid values are 1 - New 2 - In Progress 3 - Completed 4 - Shipped 5 - Canceled Enum: [1, 2, 3, 4, 5]

Response Code: 400

Bad request

The media type is application/json.

See the JSON Error Element and Error Codes section.

Update Carton — POST

This section describes the Update Carton API. Updates a previously existing carton that is in a new or in progress state, on a shipment that is also new or in progress.

Method

```
POST
```

URL

```
/rtvshipments/cartons/{cartonId}
```

Request

This section describes the request parameters.

Table 4-222 Update Create — Request Parameters

Parameters	Required	Data Type/ Example	Description
cartonId	Yes	number($int12) (path)	The unique identifier of the shipment carton.

The request body is application/json.

Details for the carton to update.

Example Value

```
{
  "externalCartonId": "4500300",
  "trackingNumber": "876300456",
  "restrictionLevel": 1,
  "cartonSizeId": 1000,
  "weight": 4.5,
  "weightUom": "LBS",
  "lineItems": [
    {
      "itemId": "10045600",
      "reasonId": 2222222,
      "caseSize": 3,
      "quantity": 100,
      "uins": [
        111345000,
        222220000
      ]
    }
  ]
}
```

Schema — VendorShipmentCartonUpdateIdo

This section describes the VendorShipmentCartonUpdateIdo schema.

Table 4-223 VendorShipmentCartonUpdateIdo — Object

Element Name	Required	Data Type/Example	Description
externalCartonId	NA	string($text128) example: 4500300	A container identifier from an external system.
trackingNumber	NA	string($text128) example: 876300456	The tracking number of the container.

Table 4-223 (Cont.) VendorShipmentCartonUpdateIdo — Object

Element Name	Required	Data Type/Example	Description
restrictionLevel	NA	integer($int4) example: 1	The hierarchy restriction level for items in the container. Valid values are 1 - Department 2 - Class 3 - Subclass 4 - None Enum: [1, 2, 3, 4]
cartonSizeId	NA	integer($int10) example: 1000	The unique identifier of the container dimension object.
weight	NA	number($decimal(12,4)) example: 4.5	The weight of the container.
weightUom	NA	string($text4) example: LBS	The unit of measure of the weight of the container.
lineItems	Yes	object	Line items

Table 4-224 VendorShipmentCreateLineItemIdo — Object

Element Name	Required	Data Type/Example	Description
itemId	Yes	string($text25) example: 10045600	The unique identifier of the SKU item.
reasonId	Yes	integer($int15) example: 2222222	The unique identier of the shipment reason.
caseSize	NA	number($decimal(10,2)) example: 3	The case size of this record.
quantity*	Yes	number($decimal(20,4)) example: 100	The quantity that was shipped.
uins	NA	string($text128) example: List [111345000, 222220000]	The UINs that were shipped. They total number must match the quantity shipped.

Responses

This section describes the responses of the Update Carton API.

Response Code: 204

No Content

Response Code: 400

Bad request

The media type is application/json.

See the JSON Error Element and Error Codes section.

Confirm Carton — POST

This section describes the Confirm Carton API. This API confirms a carton indicating the carton is ready for dispatch. This sets the carton so it can no longer be modified.

Method

`POST`

URL

`/rtvshipments/cartons/{cartonId}/confirm`

Request

This section describes the request parameters.

Table 4-225 Confirm Carton — Request Parameters

Parameters	Required	Data Type/ Example	Description
cartonId	Yes	number($int12) (path)	The unique identifier of the shipment carton.

Responses

This section describes the responses of the Confirm Carton API.

Response Code: 204

No Content

Response Code: 400

Bad request

The media type is application/json.

See the JSON Error Element and Error Codes section.

Cancel Carton — POST

This section describes the Cancel Carton API. This API cancels a carton effectively removing its contents from the shipment.

Method

POST

URL

/rtvshipments/cartons/{cartonId}/cancel

Request

This section describes the request parameters.

Table 4-226 Cancel Carton — Request Parameters

Parameters	Required	Data Type/ Example	Description
cartonId	Yes	number($int12) (path)	The unique identifier of the shipment carton.

Responses

This section describes the responses of the Cancel Carton API.

Response Code: 204

No Content

Response Code: 400

Bad request

The media type is application/json.

See the JSON Error Element and Error Codes section.

Open Carton — POST

This section describes the Open Carton API. This API opens a previously confirmed carton so that it can be modified again.

Method

POST

URL

```
/rtvshipments/cartons/{cartonId}/open
```

Request

This section describes the request parameters.

Table 4-227 Open Carton — Request Parameters

Parameters	Required	Data Type/ Example	Description
cartonId	Yes	number($int12) (path)	The unique identifier of the shipment carton.

Responses

This section describes the responses of the Open Carton API.

Response Code: 204

No Content

Response Code: 400

Bad request

The media type is application/json.

See the JSON Error Element and Error Codes section.

REST Service: Warehouse

This service integrates warehouse and warehouse item foundation data as well as warehouse item inventory adjustments.

Asynchronous warehouse integration is processed through staged messages and is controlled by the MPS Work Type: DcsWarehouse.

Service Base URL

The Cloud service base URL follows the format:

https://<external_load_balancer>/<cust_env>/siocs-int-services/api/warehouses

API Definitions

API	Description
Import Warehouses	Imports a collection of warehouses into the system.
Delete Warehouse	Deletes a warehouse from the system.
Import Items	Imports a collection of warehouse items.
Delete Items	Deletes warehouse items from the system.
Import Adjustments	Imports a collection of warehouse items adjustments that took place.
Import Inventory	Imports a collection of warehouse item inventory to update the inventory positions.

API: Import Warehouses

This will import warehouses through foundation warehouse processing.

If more than 500 warehouses are sent in a single call, an input too large error will be returned.

API Basics

Endpoint URL	{base URL}/import
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	List of warehouses to import
Output	None
Max Response Limit	500

Input Data Definition

Attribute	Data Type	Required	Description
warehouses	List of details	Yes	A list of warehouses to import.

Detail Data Definition

Attribute	Data Type	Required	Size	Description
warehouseId	Long (10)	Yes		The warehouse identifier.
name	String (150)	Yes	150	The name of the warehouse.
organizationUnit	String (15)		15	The organization the warehouse belongs to.
countryCode	String (3)		3	The ISO country code of the warehouse.
currencyCode	String (40)		3	The ISO currency code of the warehouse.

Example Output

```
{
```

"warehouses": [
{
"warehouseId": 64,
"name": "DownTownWarehouse-1",
"organizationUnit": "70001",
"countryCode": "IN",
"currencyCode": "INR"
},
{
"warehouseId": 65,
"name": "CitynWarehouse-1",
"organizationUnit": "70001",
"countryCode": "IN",
"currencyCode": "INR"
}
]
}
]

API: Delete Warehouse

Deletes a warehouse. The warehouse will not be deleted if any items remain ranged to the warehouse.

API Basics

Endpoint URL	{base URL}{warehouseId}/delete
Method	POST
Successful Response	202 Accepted
Processing Type	Synchronous
Input	None
Output	None
Max Response Limit	N/A

Path Parameter Definitions

Attribute	Definition
warehouseId	The internal identifier of the warehouse.

API: Import Items

Imports a collection of warehouse items.

If more than 5000 items are sent in a single call, an input too large error will be returned.

API Basics

Endpoint URL	{base URL}/{warehouseId}/items/import
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous (High Volume)
Input	A list of warehouse items to import
Output	None
Max Response Limit	5000

Path Parameter Definitions

Attribute	Definition
warehouseId	The internal identifier of the warehouse.

Input Data Definition

Attribute	Data Type	Required	Description
items	List of details	Yes	A list of warehouse items to import.

Detail Import Data Definition

Attribute	Data Type	Required	Description
itemId	String (25)	Yes	The item identifier.
standardUom	String (4)	Yes	The standard unit of measure of the item.
status	Integer	Yes	The status (See Index: Warehouse Item Import Status)
clearInventory	Boolean	Yes	True indicates that the inventory positions should all be set to zero.

Example Input

{

"items": [

{

"itemId": "100000147",

"standardUom": "EA",

"status": 1,

"clearInventory": true

},

{

"itemId": "100000148",

"standardUom": "KG",

"status": 2,

"clearInventory": false

}

]

}

Additional Data Definitions

Warehouse Import Item Status

Value	Definition
1	ACTIVE
2	DISCONTINUED
3	INACTIVE

API: Delete Items

Marks warehouse items for later deletion.

If more than 5000 items are sent in a single call, an input too large error will be returned.

API Basics

Endpoint URL	{base URL}{warehouseId}/delete
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous
Input	List of item ids to delete
Output	None
Max Response Limit	5000

Path Parameter Definitions

Attribute	Definition
warehouseId	The internal identifier of the warehouse.

Input Data Definition

Attribute	Data Type	Required	Description
items	List<String(25)>	Yes	A collection of up to 5000 items to remove.

Example Input

{

"itemIds": ["100000301", "100000147"]

}

API: Import Adjustments

API: Import Adjustments

A list of warehouse adjustments is processed, inventory is updated for the warehouse items, and then the adjustments are discarded.

They are not persisted anywhere and this process does not produce a transaction history record.

If more than 5000 items are sent in a single call, an input too large error will be returned.

API Basics

Endpoint URL	{base URL} {warehouseId}/adjustments/import
Method	POST
Successful Response	202 Accepted
Processing Type	Asynchronous (High Volume)
Input	A list of warehouse adjustments to import
Output	None
Max Response Limit	5000

Path Parameter Definitions

Attribute	Definition
warehouseId	The internal identifier of the warehouse.

Input Data Definition

Attribute	Data Type	Required	Description
adjustments	List of details	Yes	A list of adjustments that occurred for that warehouse.

Detail Import Data Definition

Attribute	Data Type	Required	Description
itemId	String (25)	Yes	The unique item identifier.
quantity	BigDecimal	Yes	The quantity to be adjusted.
reasonCode	Integer	Yes	The unique reason code of an inventory adjustment reason code.

Example Input

```
{
"adjustments": [
{
"itemId": "100000147",
"quantity": 100,
"reasonCode": 182
},
{
"itemId": "100000024",
"quantity": 50,
"reasonCode": 183
}
]
}
```

API: Import Inventory

This operation updates the inventory positions of a warehouse.

API Basics

Endpoint URL	{base URL}/{warehouseId}/inventory/import
Method	POST
Successful Response	200 Accepted
Processing Type	Asynchronous
Input	List of items with their inventory
Max Input	10,000 items
Output	N/A
Max Response Limit	N/A

Input Data Definition

Attribute	Type	Definition
Items	List of Warehouse Inventory	A list of items to overwrite inventory for at the warehouse.

Warehouse Inventory Ido

Attribute	Type	Definition
itemId	String(25)	The unique item identifier.
quantityTotal	BigDecimal(12,4)	The total quantity of the item in inventory.
quantityReserved	BigDecimal(12,4)	he quantity reserved for outgoing shipping.
quantityUnavailable	BigDecimal(12,4)	The quantity unavailable for usage.
quantityInTransit	BigDecimal(12,4)	The quantity in transit to the warehouse.

Example Input

{

"items

{

"itemId": "100000147",

"quantityTotal”: 100,

"quantityReserved”: 8

},

{

"itemId": "100000024",

"quantityTotal”: 50,

"quantityInTransit”: 183

}

]

}

5
Sales Integration

EICS integrates with POS systems and Sales Audit systems to ensure that the inventory positions are accurate. This is especially important where accurate up-to-date inventory positions are required to reduce customer disappointment when trying to locate items that appear in inventory or delays in filling customer orders.

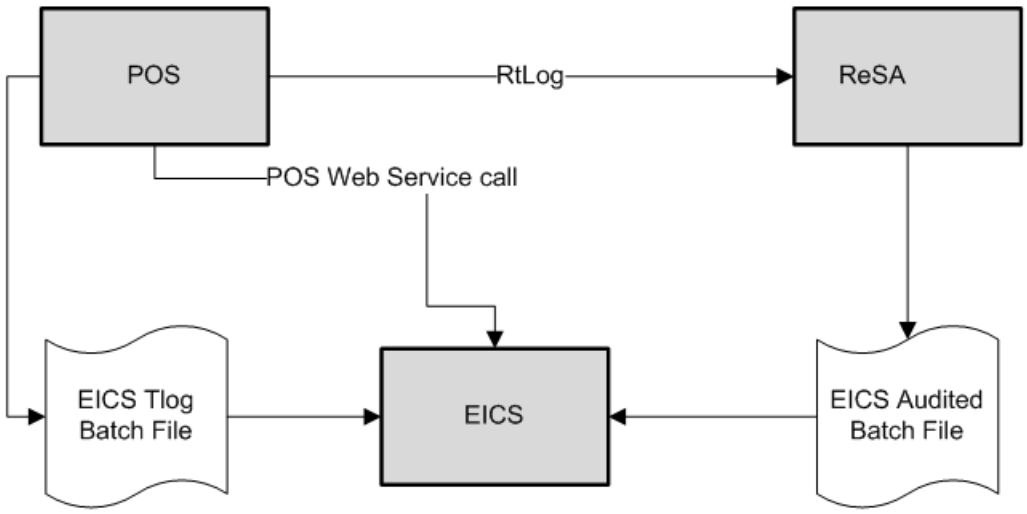
POS is the primary source of sales, returns, void, and some customer order transaction information to EICS.

ReSA sends only modified or new POS transaction records to EICS.

POS systems integrated with EICS can do the transaction notifications using a web service.

Sales Audit systems can only communicate through a file import process.

Figure 5-1 POS and Sales Audit Integration



The following features are part of this integration:

- Real-time web service integration
- Batch integration
- Audited sales data integration
- Automatic disposition processing for returns

Batch processing and ReSA processing are discussed elsewhere as are the store and system configurations that might determine how the sale is processes.

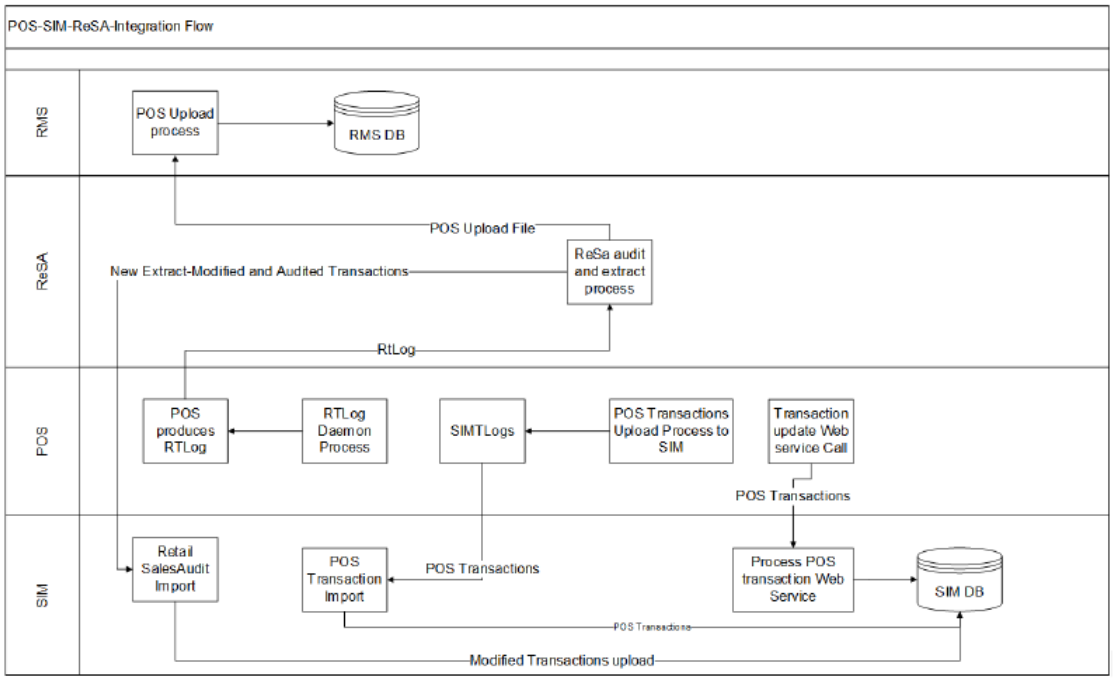
POS and Sales Audit Process Flow

The following figure shows how a POS, Retail Sales Audit, and EICS are integrated. A POS generates an RTLog containing all the POS transactions and sends it to the Oracle Retail Sales Audit system (ReSA). ReSA sends the audited modified or new transactions to EICS. ReSA also sends the POS transaction upload file to merchandising to update inventory.

Please note that Oracle Retail Xstore is interfaced with EICS to update the inventory transactions near real time only through web service. It does not use batch.

Non-Oracle POS systems can use a batch to import transactions directly into EICS. EICS also processes the POS transactions that have been changed or entered into the sales audit system and updates the inventory based on the delta.

Figure 5-2 POS and Sales Audit Process Flow



There are two reasons for POS to send sales data directly to EICS and not to the auditing system:

- Real-time inventory updates to support Commerce Anywhere are critical. A possible round trip from POS to ReSA to EICS takes too long in the dynamic inventory environment of today.
- POS is the application that owns sales data and ReSA owns audited data. Architecturally, it makes more sense to have data supplied by the owner of that data. POS sends sales data and ReSA sends audit changes to EICS.

Sales and Return Processing

As part of the sales processing, EICS updates the inventory depending on the nature of the transaction. The following are the supported transaction types for the sales processing: Sale,

Return, and Post Void of these transactions. The audit system should not modify the post void transactions. A change to a void is not supported by EICS.

Customer Order Processing

In EICS, the Retail Sales Audit import process, POS Transaction import process, and POS Transaction web service process support the following types of customer orders.

- For layaway and on hold, EICS supports create, update, cancel, and pickup/delivery. For external web order type, only pickup transactions performed in POS are sent to EICS.
- Pickup transactions, both in-store and external, cannot be voided or modified by sales audit and if these transactions are modified by sales audit system, EICS just drops the transaction and does not process.

> **Note:**
>
> Current Xstore functionality is limited to only layaway and on hold orders. Web order processing is not supported in this release.

Item Disposition

POS can move inventory for return and post void transactions to 'unavailable' or 'out of stock'. This is especially useful in some environments where items returned must be disposed of or must be reprocessed.

The external sale transaction coming into EICS may include a reason code that is mapped to the inventory adjustment reason codes in EICS. Point of Service maps the EICS reason codes, and the reason codes are sent to EICS in the web service or file extract for the return and post void transactions. EICS first processes the return or post void and updates stock on hand. Next, if the reason code exists, EICS checks this reason code with the one in inventory adjustment reason code table. If a valid match is found, EICS generates an inventory adjustment to notify external systems and execute the disposition instructions tied to the inventory adjustment reason code. Based on the disposition mapped to the reason code, EICS moves the returned inventory to not for sale or out of stock and updates the history trail. If sub-buckets are used, they are also updated if the movement is to not for sale.

If the reason code received is invalid/not present/mapped incorrectly, the system writes an error log and continues to process the stock on hand part of the transaction.

Drop Ship

When the sales records indicate the record is a drop ship, EICS does not perform any processing of this record since the drop ship process implies the inventory is shipped from a third-party location and not from the store.

Item Types

EICS only processes SKU or UPC numbers. GS1 databars, or any other smart barcodes such as VPLUs or Type-E barcodes, should have been extracted to their SKU or UPC number by the POS system.

In addition, EICS only updates inventory for stock holding items. Non-inventory items do not update any stock on hand and are not processed.

Items with the store pack inventory indicator turned off are automatically broken down and the inventory of the component items is updated.

RFID

If the point-of-sale record for an item includes an RFID tag, the tag will be moved to a SOLD status indicating it should be out-of-store.

6

Outbound Integration

This chapter provides information about the following outbound integration processes:

Integration with Customer Order System

Integration with Manifesting Systems

Integration for Notifications

Integration for Sales Forecast

Integration for Store Order

Integration for Ticket Printing

Integration with Customer Order System

CustomerOrderAddressService

When shipping to customer during the fulfillment order workflow, EICS retrieves the address for the order delivery from an external order managements system. When viewing delivery address information within the client application, it also retrieves it from an external system. The web service is defined to connect to an OrderManagementService.

Service Operation	Description
queryCustomerOrderAddress	Retrieves detailed address information for the order and customer information passed to it.

CustomerOrderService

This service connects to OrderManagementService to manage customer orders. It includes operations to create a customer order, query for customer orders, pickup/cancel items from a customer order and return items from customer orders.

Service Operation	Description
requestNewCustomerOrderId	Requests new customer order Id.
cancelNewCustomerOrderId	Cancels the new customer order id.
createCustomerOrder	Creates customer order.
queryCustomerOrder	Queries the customer order present in the system.
PickupCustomerOrderItems	Pickup items from the customer order.
ReturnCustomerOrderItems	Returns items from the customer order.
UpdateReceipt	Updates the receipt of customer order.

Integration with Manifesting Systems

In order for access to an external manifesting system to take place, the customer must first setup Carrier Type as "Third Party" and the Carrier Service (Manifest Type) must be Parcel (P). Configuration controls whether manifesting is done for a transfer to store, finisher, or warehouse. In addition, configuration controls manifesting for a return to vendor shipment or a customer order delivery.

Carrier services with manifest type of "O" (Other) and "H" (Home Fleet) do not go through the manifesting system. When Manifest Type is "O," EICS prompts the user to enter the carrier address where the shipment is to be sent for fulfillment. Manifest Type of "H" is within the company and therefore, does not prompt the user for an address.

Some carriers require weight, dimension, or both values to be sent in the manifest payload. If so, the carrier's service should have either the weight indicator or carton dimension indicate set to active (or both) during their carrier service setup.

EICS supplies an outbound and inbound Shipment Manifest SOAP web service. The following are supported service operations:

A web service is used to send all the shipment information to the external manifesting system and also to receive close shipment requests from external systems.

A web service accepts requests from external systems to close shipments. It is used to find those "Submitted" shipments for the provided tracking ID, carrier, service and date, and dispatch those shipments.

> **Note:**
>
> EICS supplies a WSDL and XSD that defines the web service, operation, and data content. This web service will need to be implemented either for the manifesting system or a plug-in set up.

ShipmentManifestService

This web service notifies an external manifesting system that a manifest needs to be created.

Service Operation	Description
createManifest	Requests the external manifesting system to create a new parcel manifest for an input transaction.

StoreShipmentManifestService

This web service receives a message from an external manifesting system that the items on the manifest have been picked up.

Service Operation	Description
closeManifest	Instructs EICS that submitted shipments have been picked up by the carrier.

Integration for Notifications

StoreExtNotificationService

When store order with external ID is approved, EICS sends notification to the external system.

This service is applicable only for externally created store orders.

Service Operation	Description
createNotification	Sends notification to external system on approving the externally created store orders with its items information.

Integration for Sales Forecast

SalesForecastService

EICS may retrieves item sales forecasting information from a third-party sales forecasting system.

Service Operation	Description
retrieveSalesForecast	Retrieves sales forecast data for the next 30 days for a particular item and store.

Integration for Store Order

OrderApproveNotificationService

When store order is approved, EICS sends notification to a third-party item management system.

This notification will be sent out for store orders that are created manually or system generated.

It is not applicable to store orders created by external system.

Service Operation	Description
orderRequestApproved	Sends notification to external item management system that the order request is approved.

StoreExtNotificationService

When store order with external ID is approved, EICS sends notification to the external system.

This service is applicable only for externally created store orders.

Service Operation	Description
createNotification	Sends notification to external system on approving the externally created store orders with its items information.

Integration for Ticket Printing

When printing tickets, EICS sends ticket information to an external system for printing. This web service needs to be implemented for printing tickets to a physical printer. In the JET administration screen for configuration external service, this endpoint can be configured to connect to either a SOAP or a REST service implemntation.

SOAP Ticket Printing

The details of the SOAP ticket printing endpoint is captured in the associated web service WSDL.

TicketPrintService

Service Operation	Description
printTickets	Sends item tickets to an external system to be printed. It must be implemented by the external system to receive the tickets.

Rest Ticket Printing

This web service defines an endpoint that can be developed by a third party in order to allow EICS to send item ticket printing information to an end system service that handles ticket printing.

The endpoint inputs and outputs must be adhered to by the provider.

API Publish Tickets

This API receives ticket printing information from EICS.

API Basics

Endpoint URL	{base URL}
Method	POST
Success Response	200 OK
Input	Input Print Request For Store and Printer
Output	None

Input Data Definition (Ticket Print Request)

Payload	Type	Definition
storeId	Long(10)	The identifier of the store.
printerName	String(200)	The name of the printer.
printerAddress	String(300)	The URI (or network address) of the printer.
printerId	String(5)	The identifier of the printer.
formatType	Integer(4)	The type of ticket format to print. See Index
formatReference	String(255)	A reference to the format content to use.
templateId	Long(12)	The identifier of a template to use to print the tickets.

zplContent	**String(3500)**	**The content of the ZPL print template.**
tickets	List<Tickets>	A collection of tickets to print.

Tickets

Payload	Type	Definition
ticketId	Long(12)	The identifier of the ticket.
itemId	String(25)	The identifier of the item/sku.
primaryUpc	Strin(25)	The primary Unique Produce Code for the item.
originType	Integer	The origin of the ticket.
sequenceNumber	Integer(3)	The sequence number of the ticket within its grouping.
ticketCount	Integer(3)	The number of instances of this ticket to print.
printQuantity	BigDecimal(12,4)	The quantity to be printed on each ticket.
shortDescription	String(255)	A short description for the item.
longDescription	String(400)	A long description for the item.
shortDescriptionLang	String(255)	A short descripton for the item at the store.
longDescriptionLang	String(400)	A long descripton for the item at the store.
diffType1	String(255)	The description of the first differentiator type.
diffType2	String(255)	The description of the second differentiator type.
diffType3	String(255)	The description of the third differentiator type.
diffType4	String(255)	The description of the fourth differentiator type.
diffDescription1	String(255)	The description of the first differentiator.
diffDescription2	String(255)	The description of the second differentiator.
diffDescription3	String(255)	The description of the third differentiator.
diffDescription4	String(255)	The description of the fourth differentiator.
departmentId	Long(12)	The department identifier of the item.
departmentName	String(360)	The department name.
classId	Long(12)	The class identifier of the item.
className	String(360)	The class name.
subclassId	Long(12)	The subclass identifier of the item.
subclassName	String(360)	The subclass name.
priceCurrency	String(3)	The currency code of the ticket price.
priceValue	BigDecimal(12,4)	The value of the ticket price.
priceType	Integer(3)	The type of the ticket price. See Index.
priceUom	String(4)	The unit of measure of the ticket price.
priceActiveDate	Date	The date the ticket price became active.
priceExpireDate	Date	The date the ticket price expired.
overridePriceCurrency	String(3)	An override price currency code.
overridePriceValue	BigDecimal(20,4)	The amount of an override price.
previousPriceCurrency	String(3)	A previous price currency code.

previousPriceValue	BigDecimal(20,4)	The amount of a previous price.
previousPriceType	Integer(3)	The price type of the previous price. See Index.
lowestMonthlyPriceCurrency	String(3)	The currency code of the lowest monthly price.
lowestMontlyPriceValue	BigDecimal(20,4)	The amount of the lowest monthly price.
lowestMonthlyPriceType	Integer(3)	The price type of the lowest montly price. See Index.
multiUnitPriceCurrency	String(3)	The currency code of the multi-unit price.
multiUnitValue	BigDecimal(20,4)	The amount of the multi-unit price.
multiUnitUom	String(4)	The unit of measure of the multi-unit price.
multiUnitQuantity	BigDecimal(12,4)	The multi-unit quantity associated to the price.
countryOfManufacture	String(3)	The country code of the country of manufacture of the item.
udas	List<TicketUdaExtIdo>	A list of user defined attributes associated to the ticket.

TicketUdaExtIdo

Payload	Type	Definition
name	String(120)	The name of the user defined attribute.
value	String(250)	The value of the user defined attribute.

Additional Data Definitions

Table Format Type

ID	Origin
1	Item Ticket
2	Shelf Label

Ticket Price Type

ID	Origin
1	Permanent
2	Promotional
3	Clearance
4	Clearance Reset

Ticket Origin Type

ID	Origin
1	External
2	Price Change

3	Foundation
4	Manual
5	Promotional Price Change
6	Clearance Price Change
7	Permanent Price Change

7

File Transfer Services

This chapter covers the following topics:

- Overview
- How to Call FTS APIs
- Handling Import Data Files
- Handling Export Data Files
- File Transfer Service UI
- FTS API Specifications
- File Transfer Service Troubleshooting
- Test FTS API using Postman

Overview

Oracle Cloud Infrastructure Object Storage is an internet-scale, high-performance storage platform that offers reliable and cost-efficient data durability.

File Transfer Service (FTS) for the Store Inventory Cloud Services is available as JSON REST services. These APIs allows you to manage uploading and downloading files to Object Storage.

Access to files is through a Pre-Authenticated Request (PAR), which is a URL that requires no further authentication to upload or download to the application's object storage. To retrieve a PAR, you must use the appropriate FTS services.

The FTS APIs enables external application to import files to and export files from Object Storage used by the solutions.

These APIs provides following services:

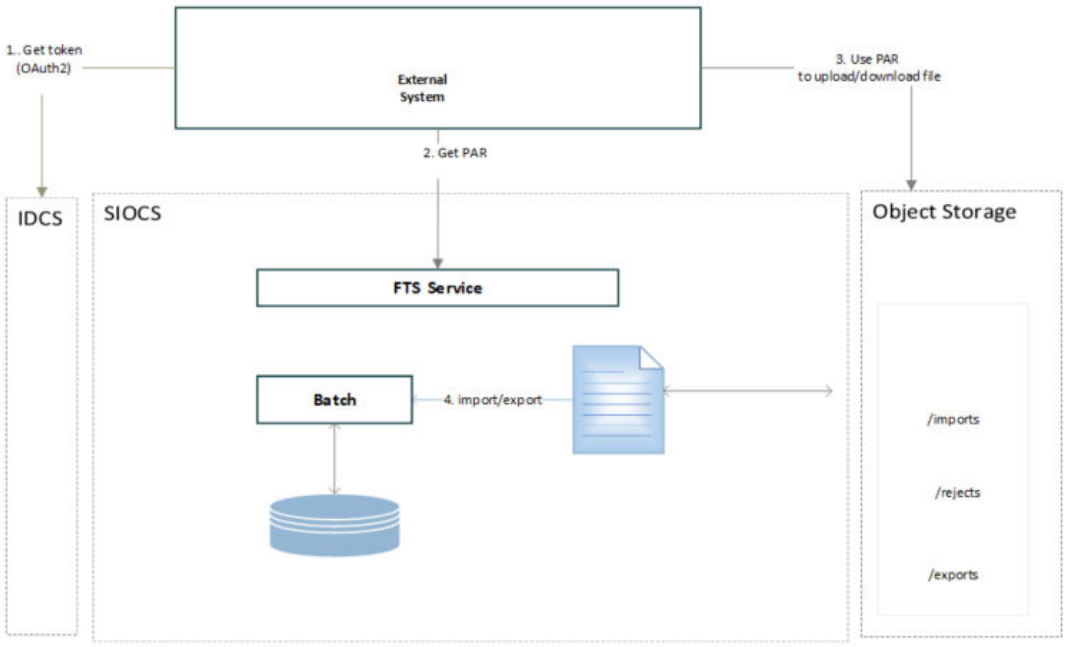
- Ping to check FTS Service health
- List storage prefixes
- List files in object storage
- Move files from object storage
- Delete Files from object storage
- Request Upload PAR
- Request Download PAR

The general process flow below describes how the external solution application interacts with FTS service for transferring files to cloud solution service:

1. The external application gets an Oauth2 token from IDCS.
2. The external application makes an FTS request with the Oauth2 token to request Pre-Authentication.

3. Once the PAR is received, the external application uploads a file to object storage using the URL included within the response.
4. The file uploads to application object storage and will be processed by the application batch jobs.

Figure 7-1 File Transfer Service Process Flow



In addition to public FTS endpoints, SIOCS also provides a File Transfer Service User Interface to view files in cloud solution object storage, to upload and download file interactively once logged into the SIOCS web client. Refer to File Transfer Service UI section for details.

How to Call FTS APIs

To interact with FTS, you must use the REST APIs provided. The endpoints URLs are relative to cloud solution integration base URL, and endpoints also include the object storage bucket name which is allocated for your environment for file services.

- Service Base URL
- FTS Bucket Name
- FTS Endpoints
- Preparing for Authorization
- Retrieving Access Client Token
- FTS API Call Common Headers
- How to Use FTS API to find Object Storage Prefixes
- How to Use FTS APIs to Upload Files to Object Storage
- How to Use FTS API to List Files in Object Storage

- How to Use FTS APIs to Download Files from Object Storage

Service Base URL

The Cloud service base URL follows the format:

https://rex.retail.<Region Name>.ocs.oraclecloud.com/<Customer Subnamespace>/siocs-int-services/api/

> **Note:**
>
> The <Region Name> and <Customer Subnamespace> part of the URL should be replaced with the one specific to your environment. This will be the same as your cloud service Application URL provided in the Welcome email.

FTS Bucket Name

For each customer environment, logical containers (buckets) are created in Object Storage for storing objects used by the cloud application. The file transfer bucket name is created and set when the environment is provisioned. The bucket name is required to move files between Oracle Cloud and your local system using file transfer services.

rgbu_rex_cnprod_<cust_env>

Example:

rgbu_rex_cnprod_rgbu-rex-custA-stg1-siocs

The ' File Transfer Service Bucket Name' is a restricted system configuration parameter on the EICS System Configuration screen. A customer Admin user (with the IDCS application role *sim_admin_users*) can perform the following steps to view the bucket name.

1. Log in to EICS web client as customer Admin user.
2. Go to **Configuration System**.
3. Check the values setting for name **File Transfer Service Bucket Name**.

FTS Endpoints

Open API documents can be viewed via the following URL:

https://{external_load_balancer}/{cust_env}/siocs-int-services/public/api/Fts.json

The table below lists the API end points for different file operations. See FTS API Specifications for details.

Table 7-1 FTS Endpoints

Service	Method	FTS Endpoint URLs
Ping	GET	{Service Base URL}/fts/ping
List Prefixes	GET	{Service Base URL}/fts/{FTS Bucket Name}/listprefixes
List Files	GET	{Service Base URL}/fts/{FTS Bucket Name}/listfiles
Move Files	POST	{Service Base URL}/fts/{FTS Bucket Name}/movefiles

Table 7-1 (Cont.) FTS Endpoints

Service	Method	FTS Endpoint URLs
Delete Files	POST	{Service Base URL}/fts/{FTS Bucket Name}/delete
Request Upload PAR	POST	{Service Base URL}/fts/{FTS Bucket Name}/upload
Request Download PAR	POST	{Service Base URL}/fts/{FTS Bucket Name}/download

> **Note:**
>
> The example in this section uses curl command line tools. You may also use Postman to test the FTS REST APIs for testing purpose. Refer to Test FTS API using Postman.

Preparing for Authorization

FTS Client Id and Client Secret

FTS APIs use OAuth2.0 for authorization.

To generate a token from IDCS, an IDCS application client will need to be created for you to use.

The Customer Administration users must create their own client credential IDCS application using the Oracle Retail Home Cloud Service. For additional details, refer to Oracle® Retail Home Administration Guide- Chapter: Oauth Application Configuration chapter – Section: Creating OAuth Client Applications.

The App name and scope that should be used for the FTS IDCS application creation should be environment specific using the format :

App Name- RGBU_SIOCS_<ENV>_EICS_FTS_INT

Scope- rgbu:siocs:integration-<ENV>

Example:

App Name- RGBU_SIOCS_STG1_ EICS_FTS_INT

Scope- rgbu:siocs:integration-STG1

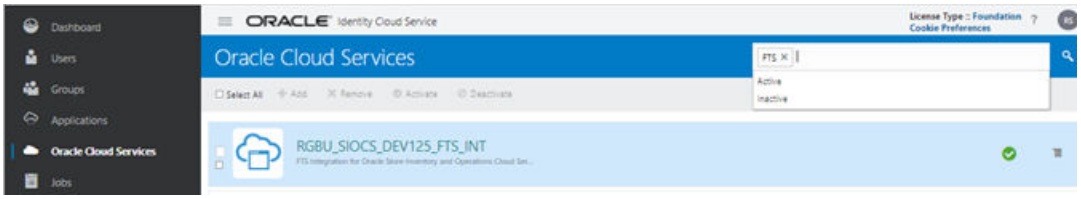
Steps to retrieve the FTS Client ID and Client Secret from IDCS:

1. Customer's IDCS Administrator log into Oracle Identity Cloud Service (IDCS) console.
2. In the left navigation panel, select **Oracle Cloud Service.**
3. On the search field, type in "FTS".
4. From the search result, find your FTS client application for cloud environment.

   FTS Client ID is like: RGBU_SIOCS_<ENV>_EICS_FTS_INT_APPID

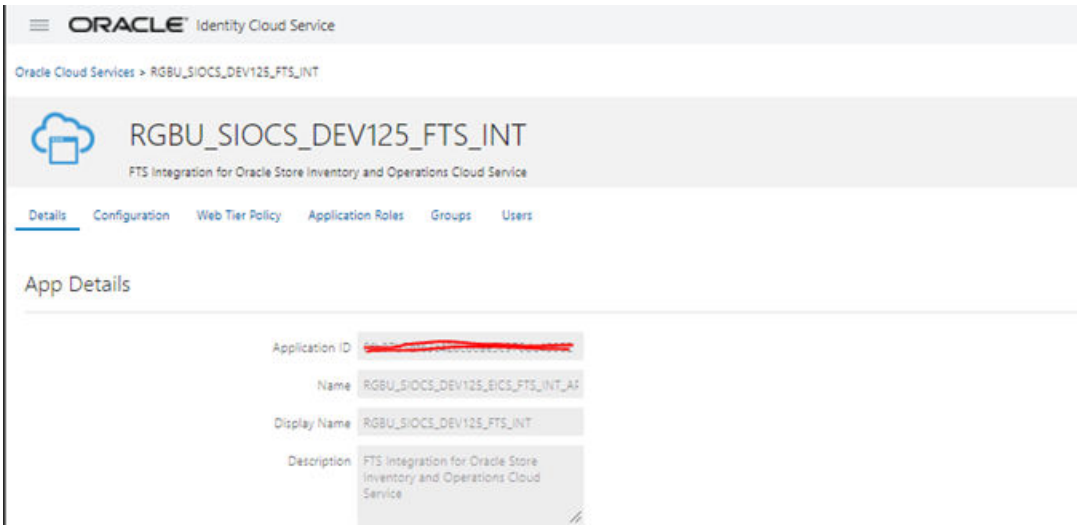
   (Example <ENV>: DEV1, STG1, PROD1 ..)

Figure 7-2 FTS Client Application



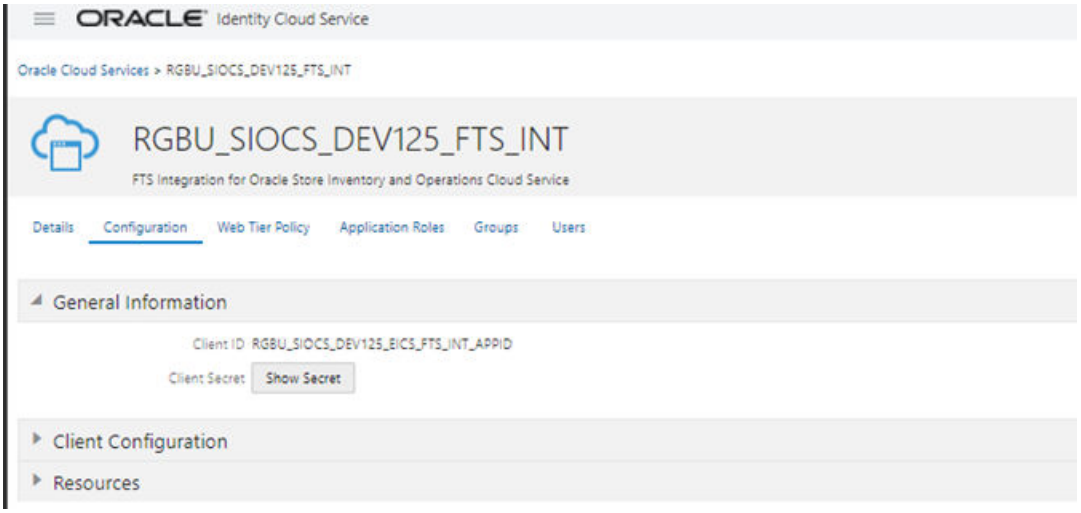
5. Click the client application, which will take you to the **Application Detail Panel**.

Figure 7-3 Application Detail Panel



6. Select the **Configuration** tab to view client Id.

Figure 7-4 Configuration Tab



7. Click **Show Secret** to see the password.

OAuth Scopes for FTS

Custom environment specific scope.

The scope pattern that is used in the FTS IDCS application creation template is rgbu:siocs:integration-{env}

For example:

rgbu:siocs:integration-STG1

IDCS OAuth2 Token URL

IDCS token URL to obtain Oauth2 token.

Example IDCS_TOKENT_URL:

https://idcs-XXXXXXXX.identity.oraclecloud.com/

Using the above URL,

IDCS_TOKENT_URL = {IDCL_BASE_URL}/oauth2/v1/token

Retrieving Access Client Token

The following is required in headers for making OAuth2.0 enabled REST Services.

- Please contact customer's IDCS administrator for FTS Client ID and Client Secret.
- An access token using the Client ID and secret from IDCS.

Example: get access Token Use Curl

```
export ACCESS_TOKEN="$(curl -u <Client ID>:<Secret> -H 'Content-Type:
application/x-www-form-urlencoded;charset=UTF-8' --request POST https://
<IDCS_BASE_URL>/oauth2/v1/token -d 'grant_type=client_credentials&scope=<Scope>'
| jq -r '.access_token')"
```

In above example, substitute the variables with proper values for your environment. See FTS Client Id and Client Secret section for obtaining Credential Client ID and Client Secret.

> **Note:**
>
> You need to have curl and jq client tool installed on your client machine for using curl for testing.

For example:

```
export ACCESS_TOKEN="$(curl -u RGBU_SIOCS_ZZZZ_EICS_FTS_INT_APPID:<secret> -H
'Content-Type: application/x-www-form-urlencoded;charset=UTF-8' --request POST
https://idcs-ZZZZ/oauth2/v1/token -d
'grant_type=client_credentials&scope=rgbu:siocs:integration-X' | jq -r
'.access_token')"
```

FTS API Call Common Headers

Each call to FTS Endpoint should contain the following Request headers:

- **Content-Type**: application/json
- **Accept**: application/json

- **Accept**: Language: en
- **Authorization**: Bearer {ACCESS_TOKEN}

Before calling FTS API, you need to get the ACCESS_TOKEN use step Retrieving Access Client Token.

How to Use FTS API to find Object Storage Prefixes

First you need to get the ACCESS_TOKEN use step Retrieving Access Client Token, then you may call the endpoint List Prefixes as below:

Sample Request:

```
curl --request GET https://rex.retail.<Region Name>.ocs.oraclecloud.com/<Customer
Subnamespace>/ /siocs-int-services/api/fts/vvvvv-siocs/listprefixes -H 'content-
type: application/json' -H 'Accept: application/json' -H 'Accept-Language: en' -H
"Authorization: Bearer ${ACCESS_TOKEN}"
```

Sample Response:

```
{"values":["archives","rejects","imports","exports"]}
```

How to Use FTS APIs to Upload Files to Object Storage

- Step1: Request upload PAR
- Step2: Use PAR to upload data files to Object Storage

Step1: Request upload PAR

First get the ACCESS_TOKEN use step Retrieving Access Client Token, then call the endpoint Request Upload PAR as below:

Sample Request:

```
curl --request POST https://rex.retail.<Region Name>.ocs.oraclecloud.com/
<Customer Subnamespace>/ /siocs-int-services/api/fts/{bucketname}/upload -H
'content-type: application/json' -H 'Accept: application/json' -H 'Accept-
Language: en' -H "Authorization: Bearer ${ACCESS_TOKEN}" -d "{\"listOfFiles\":
[{\"storagePrefix\": \"imports\",\"fileName\": \"EXTPC_1.dat\"},
{\"storagePrefix\": \"imports\",\"fileName\": \"RFID_1.dat\"}]}"
```

Sample Response:

```
{"par-List":[{"id":"zzzzzzz/:imports/
EXTPC_1.dat","name":"EXTPC_1.dat","accessUri":"https://objectstorage.us-ZZZ-
siocs/o/imports/
EXTPC_1.dat","accessType":"ObjectWrite","timeExpires":"2022-02-13T21:39:40.265Z",
"timeCreated":"2022-02-13T21:34:40.329Z","objectName":"imports/EXTPC_1.dat"},
{"id":"ZZZZ:imports/RFID_1.dat","name":"RFID_1.dat","accessUri":"https://zzzz-
siocs/o/imports/
RFID_1.dat","accessType":"ObjectWrite","timeExpires":"2022-02-13T21:39:40.411Z","
timeCreated":"2022-02-13T21:34:40.472Z","objectName":"imports/RFID_1.dat"}]}
```

Step2: Use PAR to upload data files to Object Storage

Use the accessUri returned in the get PAR response to upload the data file.

Sample Request:

```
curl https://ZZZZZ-siocs/o/imports/RFID_1.dat --upload-file C:\\temp\\RFID_1.dat
```

How to Use FTS API to List Files in Object Storage

First get the ACCESS_TOKEN using step Retrieving Access Client Token, then call the endpoint List Files as below:

Sample Request:

```
curl --request GET https://<external_load_balancer>/<cust_ env>/siocs-int-
services/api/fts//<bucketname>/listfiles?contains=RFID -H 'content-type:
application/json' -H 'Accept: application/json' -H 'Accept-Language: en' -H
"Authorization: Bearer ${ACCESS_TOKEN}"
```

Sample Response:

```
{"lim-it":999,"count":1,"offset":0,"hasMore":false,"resultSet":[{"name":"imports/
RFID_1.dat","createdDate":"2022-02-13T21:35:26Z","modifiedDate":"2022-02-13T21:35
:26Z","scanStatus":"Passed","scanDate":"2022-02-13T21:35:56.187Z","md5":"xxxx==",
"version":"xxxxx","etag":"zzzzzzz","size":75}]}
```

How to Use FTS APIs to Download Files from Object Storage

- Step1: Find what files are available for downloads
- Step2: Request Download PAR for downloading data files from Object Storage
- Step3: Download the file using the par returned from step2

Step1: Find what files are available for downloads

First get the ACCESS_TOKEN using step Retrieving Access Client Token, then call the endpoint List Files as below:

Sample Request:

```
curl --request GET https://<external_load_balancer>/<cust_ env>//siocs-int-
services/api/fts/<bucketname>/listfiles?contains=RFID -H 'content-type:
application/json' -H 'Accept: application/json' -H 'Accept-Language: en' -H
"Authorization: Bearer ${ACCESS_TOKEN}"
```

Sample Response:

```
{"lim-it":999,"count":1,"offset":0,"hasMore":false,"resultSet":[{"name":"imports/
RFID_1.dat","createdDate":"2022-02-13T21:35:26Z","modifiedDate":"2022-02-13T21:35
:26Z","scanStatus":"Passed","scanDate":"2022-02-13T21:35:56.187Z","md5":"xxxxx==",
"version":"xxxxx","etag":"ZZZZZ","size":75}]}
```

Step2: Request Download PAR for downloading data files from Object Storage

First get the ACCESS_TOKEN using step Retrieving Access Client Token, then call the endpoint Request Download PAR as below:

Sample Request:

```
curl --request POST https://ZZZZZZ-siocs/siocs-int-services/internal/fts/
rgbu_rex_cndevcorp_rgbu-rex-rgbu-dev125-siocs/download -H 'content-type:
application/json' -H 'Accept: application/json' -H 'Accept-Language: en' -H
"Authorization: Bearer ${ACCESS_TOKEN}" -d "{\"listOfFiles\":
[{\"storagePrefix\": \"imports\",\"fileName\": \"RFID_1.dat\"}]}"
```

Sample Response:


```
{"par-List":[{"id":"i91P0nFIIsgj05qrUH2ibTZ2npmbTdq1TKsGtWOerAYaE6/MYZE7840lR/
QEhaFk:imports/RFID_1.dat","name":"RFID_1.dat","accessUri":"https://
objectstorage.us-phoenix-1.oraclecloud.com/p/
ZG89KsLS_5SY7D2p7nVQt8KfJ6rLJ40FSmI97zASLRK2VrsICbvoRP0bgoQGxk3S/n/ZZZZZ-siocs/o/
imports/RFID_1.dat","accessType":"ObjectRead","timeEx-
pires":"2022-02-13T23:07:00.962Z","timeCreated":"2022-02-13T23:02:01.105Z","objec
tName":"imports/RFID_1.dat"}]}
```


Step3: Download the file using the par returned from step2

```
curl -o <destinationFileName> -X GET <PAR>
```

For example:

```
curl -o RFID_1_download.dat -X GET https://ZZZZZ-siocs/o/imports/RFID_1.dat
```

Handling Import Data Files

This section describes the general steps for an external solution application to transfer batch data files from external system to cloud application object storage.

The data to be processed can be provided as a single data file, or a zip file contains multiple data files.

The application batch imports the inbound data files from Object Storage, after the files have passed an anti-virus and malware scan. Once the files are downloaded from Object Storage, the batch process deletes the files from Object Storage to ensure it is not re-processed in next batch run. Rejected records are placed in the rejects file when applicable.

Supported Import Data Files

Table 7-2 Supported Import Data Files

File Name	Description	File Layout
Clearance File Import	The file is processed by Clearance File Import Batch. For additional details, see Batches.	Filename Format: Clearance_Tx_<YYYYMMddHHMMss>.csv See Appendix: Batch File Layout Specifications for details.
Initial Inventory Import File	The file is processed by Initial Inventory Import Batch. For additional details, see Batches.	File name prefix: EXTSTK_* See Appendix: Batch File Layout Specifications for details.

Table 7-2 (Cont.) Supported Import Data Files

File Name	Description	File Layout
Price Change File Import	The file is processed by Price Change File Import Batch. For additional details, see Batches.	Filename Format: PriceChange_Tx_<YYYYMMddHHMMss>.csv See Appendix: Batch File Layout Specifications for details.
ReSA Import File	The file is processed by Retail Sale Audit Import Batch. For additional details, see Batches.	Zip Filename Format SIMT_< YYYYMMDDHH24MISS>.zip See Appendix: Batch File Layout Specifications for details.
RFID Import File	The file is processed by Third Party RFID Import Batch. For additional details, see Batches.	Zip Filename Format RFID_<YYYYMMDDHH24MISS>.zip See Appendix: Batch File Layout Specifications for details.
Store Sequence Import	The file is processed by Store Sequence Import Batch. For additional details, see Batches.	Filename Format: SSEQ date in YYYYMMDDHH24MISS format>_<loc id>.dat See Appendix: Batch File Layout Specifications for details.
Third Party Price Import File	The file is processed by Third Party Price File Import Batch. For additional details, see Batches.	Zip Filename Format EXTPC_<YYYYMMDDHH24MISS>.zip See Appendix: Batch File Layout Specifications for details.
Third Party Stock Count Import File	The file is processed by Third Party Stock Count Import Batch. For additional details, see Batches.	Zip Filename Format STK_<YYYYMMDDHH24MISS>.zip See Appendix: Batch File Layout Specifications for details.

Upload Import Data Files to Object Storage

To upload data files to object storage, the external solution application needs to perform following steps:

1. The external application gets the Oauth2 token from IDCS.
2. The external application makes an FTS request with the Oauth2 token to requesting Pre-Authentication.
3. Once the PAR is received, the external application uploads the file to object storage using the URL included within the response.
4. Files uploaded to application object storage will be processed by cloud application batches.

Handling Export Data Files

The following describes the supported export data files which are supported by cloud application. These export data files are available for external solution applications to download.

Supported Export Data Files

Table 7-3 Supported Export Data Files

Export File Name	Description	File Name Format
Inventory Extract File	The file is generated by via Inventory export batch. For additional details, see Batches.	Filename Format: PRODUCT_LOCATION_INV_* See Appendix: Batch File Layout Specifications for details.
Stock Count Export File	The stock count export file is generated when a unit and amount stock count authorization is completed.	Zip Filename Format STK_* See Appendix: Batch File Layout Specifications for details.

Steps to Download Export Data Files from Object Storage

For retailer to download the export data files from application object storage, perform the following steps:

1. The external solution application gets the Oauth2 token from IDCS.
2. The external solution application calls the FTS service with the Oauth2 token to list the available export files in Object Storage which are generated by cloud app.
3. The external solution application calls the FTS service with the Oauth2 token, requesting Pre-Authentication to download files from object storage used by cloud app.
4. Once the PAR is received, the file is downloaded using the URL included within its response. A PAR is valid for 10 minutes. A download can take longer than 10 minutes, but it must be started within 10 minutes of the PAR being received.

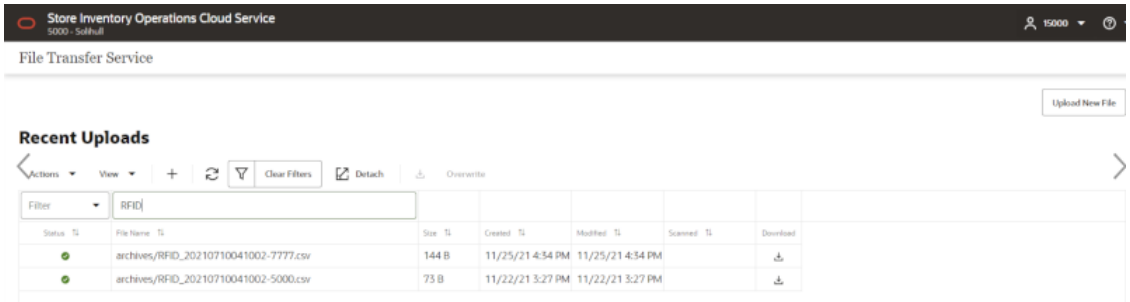
File Transfer Service UI

SIOCS provides an UI which is used to upload or download a file or view a list of files in object storage.

To access this screen, the application user needs to be assigned the **Access File Transfer Service** security permission.

The IDCS or OCI IAM application role *admin_users* is required for the user to perform the upload/download operations.

Figure 7-5 File Transfer Service UI

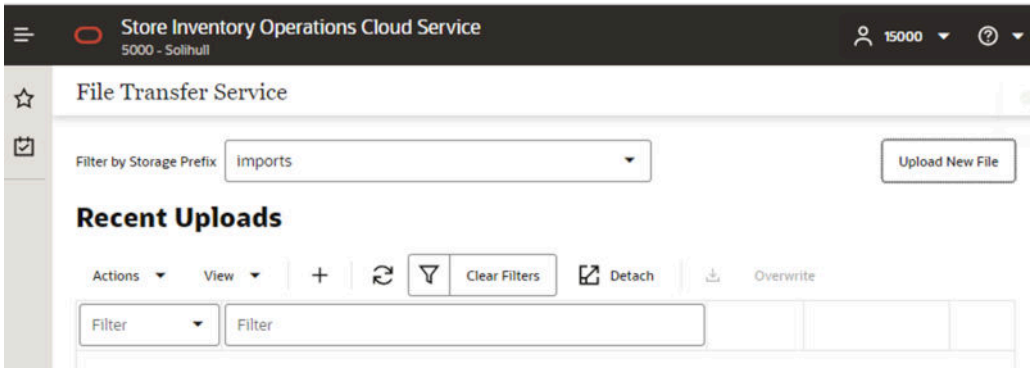


The main form lists the recently uploaded files.

Actions:

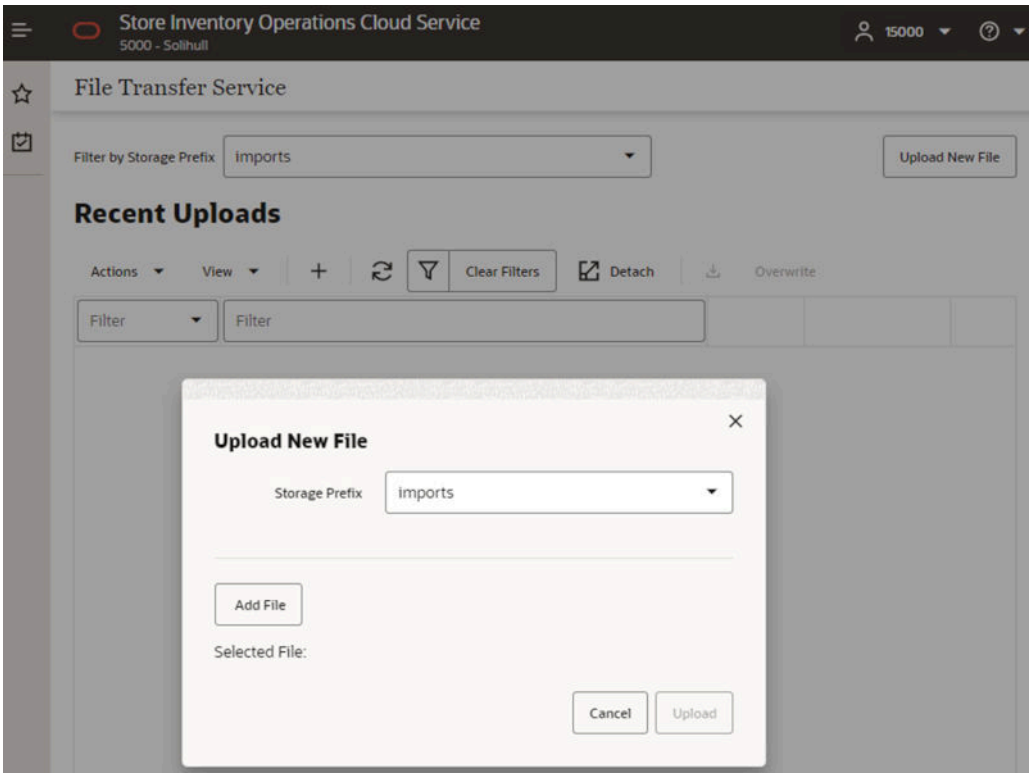
- To filter the files by store prefix, select a file storage prefix.
- To filter by file name by choosing the **Actions** choice selector on the screen.
- To upload new files, click **Upload New File** button:

Figure 7-6 Upload New File



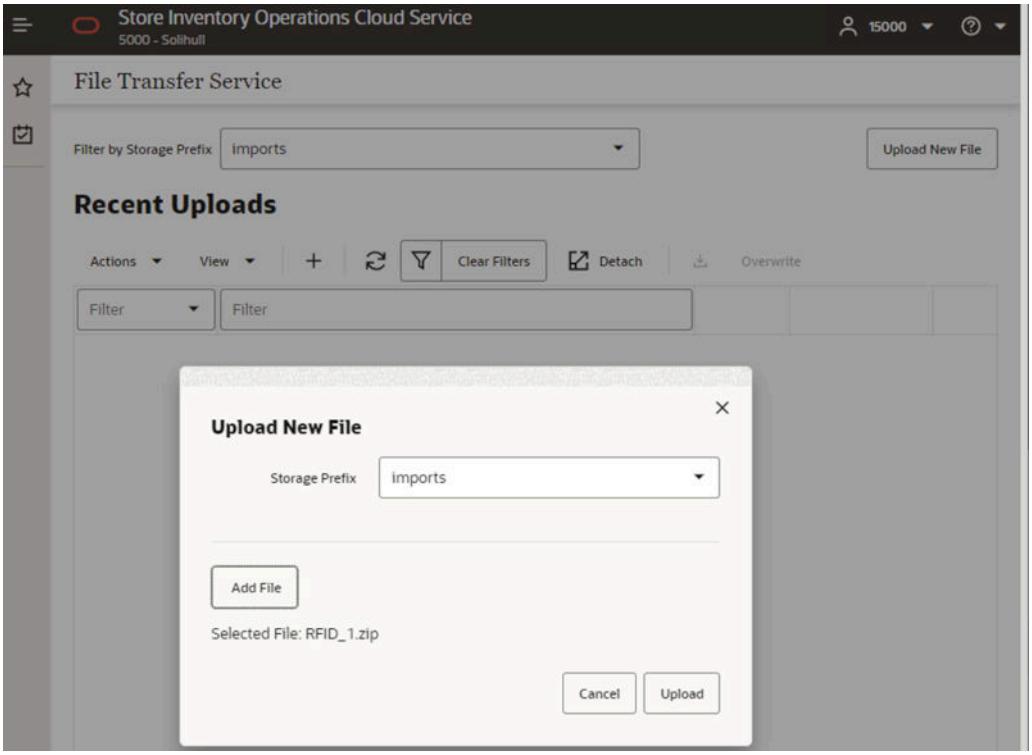
In the **Upload New File** popup dialog, choose storage prefix **Imports** and click **Add File** button.

Figure 7-7 Upload New File Dialog



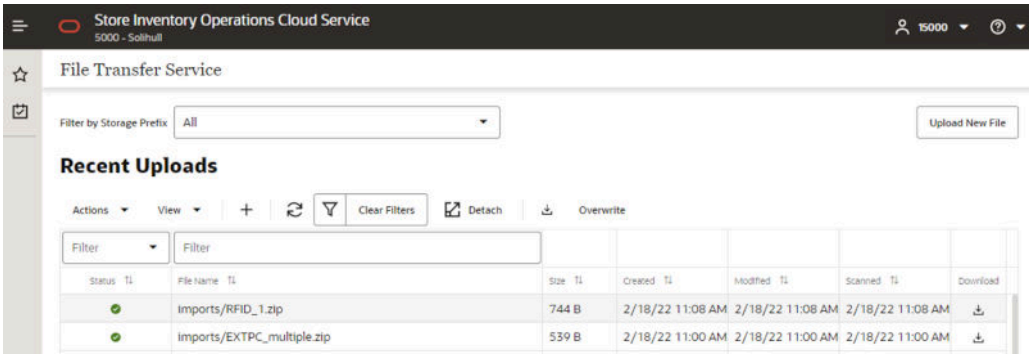
Next, choose files from your client machine, then click **Upload**:

Figure 7-8 File Added



Once the uploaded file has passed a virus scan, the file is ready for a cloud application batch to import the file from object storage into the application.

Figure 7-9 Recent Uploads



Note:

The uploaded import data files will be processed by scheduled batch import job. You may run an adhoc import batch job for testing purpose, if choose so, make sure to run the adhoc job outside of job schedule window for the select batch (or disable the job schedule for the selected batch. Once the adhoc job is completed, you will need to re-enable the batch schedule for the batch).

FTS API Specifications

This section describes FTS API specifications.

- Ping
- List Prefixes
- List Files
- Move Files
- Delete Files
- Request Upload PAR
- Request Download PAR

Ping

Returns the status of the service and provides an external health-check.

Method	GET
Endpoint	{Service Base URL}/fts/ping
HTTP Header	See Common Request Headers in making FTS API Call Common Headers.
Parameters	[{ "name": "pingMessage", "description": "Optional value to be included in the ping response.", "in": "query", "required": false, "schema": { "type": "string" } }],
Request Body	None

Response	"200": {
	"description": "OK - The service operation produced a successful response."
	},
	"400": {
	"description": "Bad Request - The path params or query params or body was not valid for this operation."
	}

List Prefixes

Returns a list of the known storage prefixes. These are analogous to directories and are restricted to predefined choices per service. SIOCS has list of pre-defined storage prefixes: import, exports, rejects and archives.

Method	GET
Endpoint	{Service Base URL}/fts/{FTS Bucket Name}/listprefixes
HTTP Header	See Common Request Headers in making FTS API Call Common Headers.

Method	GET

Parameters	[
	{
	"name": "bucketName",
	"description": "Bucket identifier.",
	"in": "path",
	"required": true,
	"schema": {
	"type": "string"
	}
	}
	],

Request Body	None

Response	A JSON array of strings containing the known prefixes.
	{
	"200": {
	"description": "OK - The service operation produced a successful response."
	},
	"400": {
	"description": "Bad Request - The path params or query params or body was not valid for this operation."
	}
	}

List Files

Returns a list of the files within a given storage prefix.

Method	GET
Endpoint	{Service Base URL}/fts/{FTS Bucket Name}/listfiles
HTTP Header	See Common Request Headers in making FTS API Call Common Headers.

Method	GET
Parameters	{ "name": "bucketName", "description": "Bucket identifier.", "in": "path", "required": true, "schema": { "type": "string" } }, { "name": "prefix", "description": "The object filter in object storage.", "in": "query", "required": false, "schema": { "type": "string" } }, { "name": "contains", "description": "The object filter in object storage.", "in": "query", "required": false, "schema": { "type": "string" } }, { "name": "scanStatus", "description": "The object filter in object storage.", "in": "query", "required": false, "schema": { "type": "string" } },

Method	GET

	{
	"name": "offset",
	"description": "The object filter in object storage.",
	"in": "query",
	"required": false,
	"schema": {
	"type": "integer"
	}
	},
	{
	"name": "limit",
	"description": "The object filter in object storage.",
	"in": "query",
	"required": false,
	"schema": {
	"type": "integer"
	}
	},
	{
	"name": "sort",
	"description": "The object filter in object storage.",
	"in": "query",
	"required": false,
	"schema": {
	"type": "string"
	}
	}
	],

Request Body	None

Response	A JSON resultSet containing array of files. For each file, there is metadata including: name, size, created and modified dates, scan status and date, scan output message.
	{
	"200": {
	"description": "OK - The service operation produced a successful response."
	},
	"400": {
	"description": "Bad Request - The path params or query params or body was not valid for this operation."
	}

Move Files

Moves one or more files between storage prefixes, while additionally allowing the name to be modified.

Method	POST
Endpoint	{Service Base URL}/fts/{FTS Bucket Name}/movefiles
HTTP Header	See Common Request Headers in making FTS API Call Common Headers.

Parameters	[
	{
	"name": "bucketName",
	"description": "Bucket identifier.",
	"in": "path",
	"required": true,
	"schema": {
	"type": "string"
	}
	}
	]

Request Body	{"listOfFiles": [{"currentPath": { "storagePrefix": "string", "fileName": "string"}, "newPath": { "storagePrefix": "string", "fileName": "string" } } }}

Delete Files

Deletes one or more files.

Method	POST
Endpoint	{Service Base URL}/fts/{FTS Bucket Name}/delete
HTTP Header	See Common Request Headers in making FTS API Call Common Headers.

Parameters	[
	{
	"name": "bucketName",
	"description": "Bucket identifier.",
	"in": "path",
	"required": true,
	"schema": {
	"type": "string"
	}
	}
	]

Request Body	A JSON array of files to be deleted. One or more pairs of storagePrefix and filename elements can be specified within the array. Required: true{ " {"listOfFiles": [[{ " {"storagePrefix": "string", "fileName": "string" }]}

Response	A JSON array of each file deletion attempted and the result.
	{
	"200": {
	"description": "OK - The service operation produced a successful response."
	},
	"400": {
	"description": "Bad Request - The path params or query params or body was not valid for this operation."

Request Upload PAR

Request PAR for uploading one or more files.

Method	POST
Endpoint	{Service Base URL}/fts/{FTS Bucket Name}/upload
HTTP Header	See Common Request Headers in making FTS API Call Common Headers.

Parameters	[
	{
	"name": "bucketName",
	"description": "Bucket identifier.",
	"in": "path",
	"required": true,
	"schema": {
	"type": "string"
	}
	}]

Request Body	A JSON array of files to be uploaded. One or more pairs of storagePrefix and filename elements can be specified within the array.
	Required: true
	{ "listOfFiles":
	[
	{
	"storagePrefix": "string",
	"fileName": "string"
	}
	]
	}

Method	POST

Response	A parList containing an array containing elements corresponding to the request including the PAR accessUri and name of file.
	{
	"parList": [
	{
	"id": "string",
	"name": "string",
	"accessUri": "string",
	"objectName": "string",
	"accessType": "string",
	"timeExpires": "2021-09-07T16:35:27.390Z",
	"timeCreated": "2021-09-07T16:35:27.390Z"
	}
	]
	}
	Response Status:
	{
	"200": {
	"description": "OK - The service operation produced a successful response."
	},
	"400": {
	"description": "Bad Request - The path params or query params or body was not valid for this operation."
	}

Request Download PAR

Request PAR for downloading one or more files.

Method	POST
Endpoint	{Service Base URL}/fts/{Bucket Name}/download
HTTP Header	See Common Request Headers in making FTS API Call Common Headers.

Parameters	[
	{
	"name": "bucketName",
	"description": "Bucket identifier.",
	"in": "path",
	"required": true,
	"schema": {
	"type": "string"
	}
	}
	]

Request Body	A JSON array of files to be downloaded. One or more pairs of storagePrefix and filenames can be specified within the array. Required: true
	{
	"listOfFiles": [
	{
	"storagePrefix": "string",
	"fileName": "string"
	}
	]
	}

Response	A parList containing an array containing elements corresponding
	to the request including the PAR accessUri and name of file.
	"parList": [
	{
	"id": "string",
	"name": "string",
	"accessUri": "string",
	"objectName": "string",
	"accessType": "string",
	"timeExpires": "2021-09-07T16:35:27.390Z",
	"timeCreated": "2021-09-07T16:35:27.390Z"
	}
	]
	}
	Response Status:
	{
	"200": {
	"description": "OK - The service operation produced a successful response.
	"

File Transfer Service Troubleshooting

These troubleshooting topics covers common file transfer service issues and possible solutions.

Troubleshooting File Transfer Service Internal Server Error

1. Try to connect to File Transfer Ping endpoint. If you can connect ping endpoints, continue to step2.
2. Try to invoke List Files endpoint, if get response status 200, continue to step3.
3. Verify the bucket name. The bucket name should have value like rgbu_rex_cnprod_<cust_env>
4. Make sure the bucket name in service request matches the configuration value set for 'File Transfer Service Bucket Name' in the system configuration screen.

   If the above steps do not resolve the internal server error, you may raise a Service Request on My Oracle Support.

Test FTS API using Postman

- Step 1: Get Client Access Token
- Step 2: Call FTS Endpoints

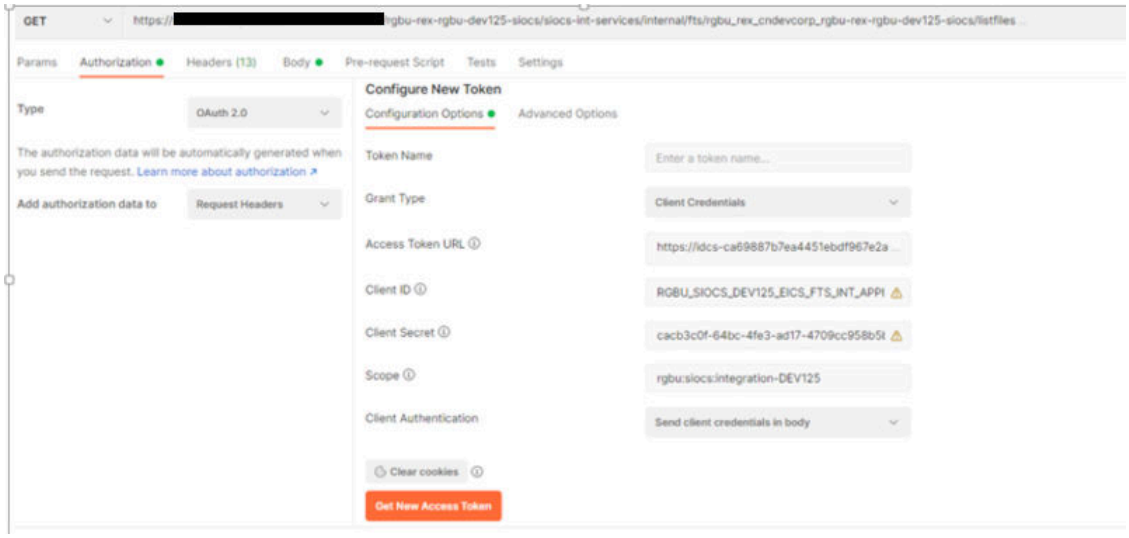
Step 1: Get Client Access Token

OAuth tokens can also be obtained by REST client tools like postman for testing purposes.

When using Postman testing, fill in the required details:

- **Authorization**: OAuth 2.0
- **Access Token URL**: https://{IDCS_BASE_URL}/oauth2/v1/token
- **Client ID**: Client if of the OAuth
- **Client Secret**: Client secret of OAuth Client app
- **Grant Type**: client_credentials
- **Scope**: The scope pattern that is used in the FTS IDCS app creation template is rgbu:siocs:integration-{env}{env index}

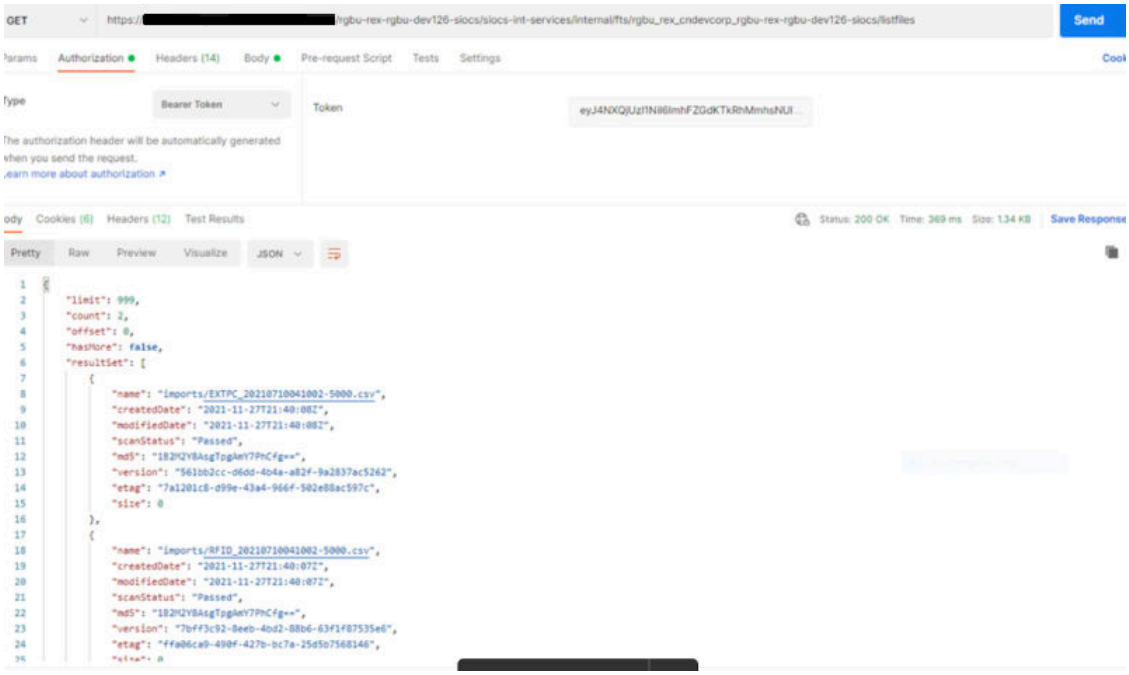
Figure 7-10 Get Client Access Token



Step 2: Call FTS Endpoints

Change **Authorization Type** to **Bearer Token**, use the access token returned from step1 as the **Token Value** as below:

Figure 7-11 Call FTS Endpoints



8

SOAP Web Services

EICS provides a large range of web services to manage the processing of information that is controlled within EICS. Each web service covers a topical area of functionality within EICS and contains numerous operations within to accomplish this functionality. This document is only meant as an outline or summary into using EICS web services and assumes the user has access to the fully documented APIs through the publishing of the web services themselves.

- Security Considerations
- Functionality
- Available Web Services
- Web Services Basic Design Principles
- Internally Managed vs Externally Managed
- Web Service Operation Basic Design Standards
- Interpreting Validation Errors

> **Note:**
>
> The WSDL files are available to download from My Oracle Support (MOS) Document 2614551.1.

Security Considerations

The SOAP web services provided by EICS are secured by Policy A using Oracle WebLogic WS-Policy configurations defined in the xml files included in Oracle WebLogic:

- Policy A
  - **Description**: Message must be sent over SSL and requires authentication of a plain text UsernameToken.
  - **Configuration**: Wssp1.2-2007-Https-UsernameToken-Plain.xml

Customers should create IDCS or OCI IAM user and the user should be assigned integration_users IDCS or OCI IAM application role to access the web-service endpoints.

See *Oracle Retail Enterprise Inventory Cloud Service Security Guide* and *Oracle Retail Enterprise Inventory Cloud Service User Guide* - Security chapter.

For REST web service security see REST WEB Services Security Considerations later in this guide.

Functionality

This document is intended to be used by someone who has read and understands all the functional areas and business functionality described in the *Oracle Retail EICS User Guide* and *Oracle Retail EICS Administration Guide.*

Available Web Services

The following list contains a summary of the web services available in EICS.

Web Service	Description
ActivityLock	This service is used to manage the locking of data within EICS. Data needs to be locked to be updated securely.
FulfillmentOrderDelivery	This service is used to manage fulfillment order deliveries (outgoing shipment to customers). It allows the creation, cancellation, and dispatch of deliveries.
FulfillmentOrderPick	This service is used to manage fulfillment order picking within EICS. It allows the creation, deletion, and confirmation of a pick to complete a fulfillment order.
FulfillmentOrderReversePick	This service is used to manage fulfillment order reverse picking within EICS. It allows the creation, update, deletion, and confirmation of a reverse pick.
InventoryAdjustment	This service is used to manage inventory adjustments within EICS. It allows the creation, update, cancellation, and confirmation of inventory adjustments.
ItemBasket	This service is used to manage item baskets within EICS. It allows the creation, update, and removal of item baskets.
OrderRequest	This service is used to create, read, update, approve, cancel and lookup store orders.
POSTransaction	This service processes external point-of-sale transactions updating the inventory accordingly. A point-of-sale is considered an externally managed transaction (internally and externally managed transaction are covered later in this document).
ProductGroup	This service is used to create or update a product group.
ProductGroupSchedule	This service is used to create, update, or cancel a product group schedule.
ReplenishmentGap	This service is used to create, update, or delete a replenishment gap.
RfidInventory	This service is used to create, update, or delete a RFID facility zone. It is also used to refresh inventory and to process RFID events.
ShelfAdjustment	This service is used to create, update, cancel or confirm a shelf adjustment.
ShelfReplenishment	This service is used to create, update, cancel or confirm a shelf replenishment.
StockCount	This service is used to retrieve the details of a stock count or a stock count child (section of stock count).
Store	This service is used to retrieve information about stores such as store detail, associated stores, or transfer zones.
StoreFulfillmentOrder	This service is used to manage fulfillment orders within EICS. It allows for the cancellation and rejection of orders and items.

Web Service	Description
StoreInventory	This service is used to lookup information about inventory positions and has several different operations to do so.
StoreInventoryISN	This service is used to create, update, or delete ISN data in EICS.
StoreInventoryUIN	This service is used to create, update, generate or read a UINs.
StoreItem	This service is used to lookup various information about an item within the store.
StoreItemPrice	This service is used to lookup prices about items within a store.
StoreNotification	This service is used to create new notifications within the system.
StoreShipmentManifest	This service is used to close documents based on shipped container information.
StoreShipmentReason	This service is used to retrieve shipment reasons codes to use when creating shipments.
StoreTicket	This service is used to create tickets and lookup ticket formats.
StoreTransfer	This service is used to create, update, and request a transfer, which describes the intent to ship items to another store or to a warehouse. It is also used to approve or reject that request. It can be used to directly create, update, approve, cancel, or close an actual transfer.
TransferDelivery	This service is used to update, receive, or confirm a transfer delivery (delivery arriving from another store or warehouse). It is also used to create, update, receive, cancel, or confirm the containers on that delivery.
TransferShipment	This service is used to create, update, submit, or dispatch a transfer shipment (shipment going out to another store or warehouse). It is also used to create, update, cancel, or confirm the containers on that shipment.
VendorDelivery	This service is used to update, receive, reject, or confirm a vendor delivery (delivery arriving from a supplier). It is also used to create, update, cancel, or confirm the containers on that delivery.
VendorReturn	This service is used to create, update, approve, cancel, or close a vendor return document, which describes the intent to ship items to a supplier.
VendorShipment	This service is used to create, update, open, submit, cancel submit or dispatch a vendor shipment (outgoing shipment to a supplier). It is also used to create, update, cancel, submit, or confirm the containers on that shipment.

Web Services Basic Design Principles

Empty Response

In the cast that a web service does not return any information (an empty list), the external system needs to understand that this is a valid response that indicates no item, transaction or queried information was found or retrieved. For example, performing a lookup in which the search criteria entered matched no input.

Error Return Key

Errors returned through a web service will be in the form of a key. This key should be translated into correct language and verbiage by the external system. EICS will not do this translation or provide English verbiage for the encountered web service error.

Boolean Data Type

If a Boolean is the data type on the interface to EICS, and no value is provided, EICS will default the value to False.

Configured System Options in EICS

Web services apply system configurations to the request that are coming in through the web service but assumes that all input validation that requires user interaction to confirm has been completed by the consumer of the web service (the third party system). This system configuration user-interaction option will be assumed to have been confirmed during the web service processing. In case the system option is a fixed restriction that does not require user interaction, and the input fails this restriction, the web service will return an error. For example:

- Shipping inventory when inventory is less than 0 can be allowed by the user of EICS. The web service assumes that the third party application did prompt the user or that their business always allows the user to do this activity.
- Adding a non-ranged item requires both a system configuration option to be enabled and the user to confirm the process. If the system configuration does not allow it, the web service will block the transaction and return an error. If the system configuration does allow adding non-ranged items, it is automatically assumed that a user confirmed its addition, and the web service adds the item.
- Allowing Receiver Unit Adjustments are dependent on a period of time. If a receiver unit adjustment were to come into EICS after that period, it would automatically be rejected, and the web service would return an error regardless of presentation or confirmation of user done by the external system.

Internally Managed vs Externally Managed

Internally Initiated

Internally initiated indicates the EICS was responsible for the original creation of the transaction being processed. A web service that creates a new transaction within EICS to be managed creates an internally initiated transaction.

Externally Initiated

Externally initiated indicates that another system created the transaction, has information about it, and notifies EICS of its creation through a notification system, not by requesting EICS create new information. EICS might manage the data after the notification but did not create the data.

Internally Managed

Internally managed data is information in which EICS is responsible for tracking its state and processing its life cycle. Our deliveries and shipments are primary examples of this. They may be externally initiated or internally initiated, but either way, they are internally managed. EICS is responsible for approving, picking, packing, manifesting, and dispatching the system and internally manages that process.

Externally Managed

Externally managed data is information that EICS does not process or track and is simply informed about after the externally managed data is complete. Point-of-sale transactions are a perfect example of this. We do not manage the sale, but once it is complete, EICS is notified and adjusts the inventory accordingly.

Web Services

EICS web services are intended for integration to allow a system using those services to control the flow and processing within EICS. Our web services are primarily designed (almost all of them) to internally manage the information. The services are intended to be used real time with the steps such as approving, picking, and dispatching occurring with real time access to EICS web services while the process is happening.

EICS web services are not designed for externally managed information. If a system is controlling the state managements itself and not informing EICS until later, this will produce out-of-sync inventory. For example, if you create a shipment, pack the shipment, and send it out and then a day later use the web service, to create, update, and dispatch the shipment, all dates and processing of inventory movements will be tagged with the later date as if they occurred real time when the web service is used.

The point-of-sale service is an externally managed service, where the timestamp on the service can be any date and EICS handles the logic of dating things according to that timestamp. Inventory Adjustment also has an "adjustment date" which represents the time the adjustment took place and so the movement of inventory can be controlled externally.

Web Service Operation Basic Design Standards

This section discusses the general approach and design standards for naming and intent regarding operations within a web service.

Lookup

Lookup operations take either an identifier of a set of criteria and find all the relevant records associated to it. A thin or light view of the data being asked for is returned giving reference to information you can do further interrogation on.

Read

Read operations take an identifier and return all relevant information to it. It may only be one level, however. For example, reading a transfer shipment returns only all the information at the shipment level and does not read information at the container or item level. Usually, the entity that contains items will also retrieve the items. Reading a container will return the container information and the item information within.

Create

Create usually inserts and generates something new and returns an identifier, reference, or handle to that information. Create normally does not take a great deal of information, such as items or anything, but rather gives you a set of IDs that then lets you update the transaction with that reference.

Save or Update

Save or update is used to modify the data usually without changing state on the transaction. The save or update operation is used to add items, remove items, edit attributes, change quantities and all the other tasks one does during a process.

Approve, Cancel, Confirm or Dispatch

Activities that change state take in a simple identifier and then process that state change. To dispatch a shipment, you pass in a reference only to the shipment and it becomes shipped,

updating the inventory. This means all changes are done through the save operations prior to making the state change.

Interpreting Validation Errors

If some data could not be processed, the web service will return a fault or a validation fault. The general form that a fault will take is to be a series of problem detail nodes containing a key and value that describes the fault. The first problem detail node will have the key ERROR and the value will be a description of the error type such as INVALID_INPUT. This will be followed by a series of nodes where the KEY is an object class name (ex: Transfer) and the value is its identifier (ex: 123) describing the hierarchy of data the error took place in. For example, a transfer container fault would have two nodes (Transfer:123) and then (TransferCarton:456). If a specific attribute is known, the final node in any problem detail series, it will have the key ATTRIBUTE and the value will be the name of the attribute of the error (ex: ITEM_ID:A5X).

Problem Detail Name	Value
ERROR	This describes the error (for example: INVALID_INPUT)
ATTRIBUTE	Identifies the specific attribute that had an error.

EICS follows the same business rules when processing information from a web service as it does from any of its clients, so the same business rules and functionality that exist in the User's Guide also exists for the web service. Understanding the basic functionality will help interpret why the validation or processing error occurred.

Common Error Codes

The following codes are paired as values to the ERROR Key:

Error Code	Description
ACTIVITY_LOCK_NOT_GRANTED	Indicates that a requested activity lock on a piece of data was not granted.
DUPLICATE_INPUT	Indicates the service would create a duplication of input that should be unique.
INVALID_DATE_RANGE	Indicates the end date of a date range is prior to the start date.
INVALID_INPUT	Indicates that the input is invalid. This error is usually followed by object and attribute information.
INVALID_ITEM	Indicates the item does not exist in the system.
INVALID_STATE_FOR_UPDATE	Indicates the transaction or data specified is not in a state that allows it to be updated (such as canceled).
INPUT_MISMATCH	Indicates the input to the web service has been altered incorrectly when compared to existing data. For example, the store identifier is different on the web service request than the currently existing transaction.
INPUT_TOO_LARGE	Indicates the input in the web service is larger than is allowed in the transaction date.
ITEM_NOT_RANGED	Indicates the item has not been activated in the location for which the request is made.
MULTIPLE_STORE	Indicates a batch of input data (such as a point-of-sale transaction) was for more than one store in a single web service call.
TIMEZONE_NOT_GMT	Indicates the time input of the web services was not in GMT.

Error Code	Description
UOM_MISMATCH	Indicates a mismatch of unit of measure information between the input and currently existing data that does not allow the information to be accurately merged.

Validation Error (Fault Example)

<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">

<S:Body>

<ns0:Fault xmlns:ns0="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ns1="http://www.w3.org/2003/05/soap-envelope">

<faultcode>ns0:Server</faultcode>

<faultstring>VALIDATION_ERROR</faultstring>

<detail>

<ns0:ValidationWSFaultException xmlns:ns0="http://www.oracle.com/retail/integration/services/exception/v1">

<ns0:shortErrorMessage>VALIDATION_ERROR</ns0:shortErrorMessage>

<ns0:BusinessProblemDetail>

<ns0:problemDescription>VALIDATION_ERROR</ns0:problemDescription>

<ns0:ProblemDetailEntry>

<ns0:name>ERROR</ns0:name>

<ns0:value>INVALID_INPUT</ns0:value>

</ns0:ProblemDetailEntry>

<ns0:ProblemDetailEntry>

<ns0:name>ShlfAdjRef</ns0:name>

<ns0:value>1</ns0:value>

</ns0:ProblemDetailEntry>

<ns0:ProblemDetailEntry>

<ns0:name>ATTRIBUTE</ns0:name>

<ns0:value>shelfAdjustmentId</ns0:value>

</ns0:ProblemDetailEntry>

</ns0:BusinessProblemDetail>

</ns0:ValidationWSFaultException>

</detail>

</ns0:Fault>

</S:Body>

```
</S:Envelope>
```

Business Error (Fault Example)

```
<S:Envelope xmlns:S="http://schemas.xmlsoap.org/soap/envelope/">
<S:Body>
<ns0:Fault xmlns:ns0="http://schemas.xmlsoap.org/soap/envelope/" xmlns:ns1="http://www.w3.org/2003/05/soap-envelope">
<faultcode>ns0:Server</faultcode>
<faultstring>BUSINESS_ERROR</faultstring>
<detail>
<ns0:ValidationWSFaultException xmlns:ns0="http://www.oracle.com/retail/integration/services/exception/v1">
<ns0:shortErrorMessage>BUSINESS_ERROR</ns0:shortErrorMessage>
<ns0:BusinessProblemDetail>
<ns0:problemDescription>BUSINESS_ERROR</ns0:problemDescription>
<ns0:ProblemDetailEntry>
<ns0:name>ERROR CODE</ns0:name>
<ns0:value>ADJUSTMENT_NOT_FOUND</ns0:value>
</ns0:ProblemDetailEntry>
</ns0:BusinessProblemDetail>
</ns0:ValidationWSFaultException>
</detail>
</ns0:Fault>
</S:Body>
</S:Envelope>
```

Web Services

Web services available in EICS:

ActivityLock

The following operations are available within the ActivityLock web service.

Operation	Description
lookupActivityLock	Retrieves information about one or more activity locks that match the input criteria.
readActivityLock	Retrieves detailed information about a single lock using its identifying reference.

Operation	Description
createActivityLock	Created an activity lock on a transaction.
deleteActivityLock	Deletes an activity lock thereby releasing processing on a transaction.

Standard Usage

An activity lock is a record indicating the user, time, and a piece of information (a transaction) that should be considered "locked". All server processing validates that the accessing user has a lock on the information before updating, notifying the current user if someone else has modified the information while they were locked and preventing the stale update.

Developers should create locks on information prior to performing update calls and delete locks when the update if finished. For example, create a lock on inventory adjustment with ID 123 with the ActivityLock service, then use `saveInventoryAdjustment` in the Inventory Adjustment service with Adjustment 123, and then delete the activity lock using the ActivityLock service. If you do not gain the lock, you will receive an error when attempting to save an inventory adjustment.

FulfillmentOrderDelivery

The following operations are available within the FulfillmentOrderDelivery web service.

Operation	Description
lookupFulfillmentOrderDeliveryHeaders	Retrieves summary information for fulfillment order deliveries that match the search criteria input.
readFulfillmentOrderDeliveryDetail	Reads the complete detailed information about a fulfillment order including items and quantities.
createFulfillmentOrderDelivery	Creates a new fulfillment order delivery including items and quantities in an in-progress status to be further worked on.
cancelFulfillmentOrderDeliverySubmission	Cancels the fulfillment order review and moves it back into in-progress status for further work.
dispatchFulfillmentOrderDelivery	Dispatches the fulfilment order delivery completing the delivery and updating the inventory.
submitFulfillmentOrderDelivery	Submits the fulfillment order delivery for review prior to dispatching.
updateFulfillmentOrderDelivery	Updates a fulfillment order delivery including items and quantities. This operation requires an activity lock.
lookupCustomAttributeAdmins	Retrieves the custom attribute administration information that describes what customized attributes are available on the fulfillment order delivery.

Standard Usage

A user can create a delivery by using `createFulfillmentOrderDelivery` references the fulfillment order to make a delivery for. The user can then use `updateFulfillmentOrderDelivery` to fill in all the quantities that are going to be shipped and finally use `dispatchFullfillmentOrderDelivery` to indicate that the order has been shipped out, which moves the inventory appropriately.

FulfillmentOrderPick

The following operations are available within the FulfillmentOrderPick web service.

Operation	Description
lookupFulfillmentOrderPickHeaders	Retrieves summary information for fulfillment order picks that match the search criteria input.
readFulfillmentOrderPick	Reads the complete detailed information about a fulfillment order pick including items and quantities.
confirmFulfillmentOrderPick	Confirm the fulfillment order pick which allows it to move on to the delivery cycle.
deleteFulfillmentOrderPick	Deletes a fulfillment order pick.
createFulfillmentOrderPickByFulfillmentOrder	Generate a pick based on the information in a fulfillment order.
createFulfillmentOrderPickByBin	Generate a pick based on a number of bins selecting orders as needed to fill the bins.
updateFulfillmentOrderPick	Update the item and quantity information about a pick. This operation requires an activity lock.
lookupCustomAttributeAdmins	Retrieves the custom attribute administration information that describes what customized attributes are available on the fulfillment order pick.

Standard Usage

Picking is used to reserve or set aside quantities for a later delivery. The user can create a pick for an order using `createFulfillmentOrderPickByFulfillmentOrder` or create a bin to places multiple orders in with `createFulfillmentOrderPickByBin`. The picked quantities can be updated through the `updateFullfillmentOrderPick` operation and when the pick is finished, it can be finalized with `confirmFulfillmentOrderPick` which sets assigned the goods as reserved in inventory.

FulfillmentOrderReversePick

The following operations are available within the FulfillmentOrderReversePick web service.

Operation	Description
lookupReversePickHeaders	Retrieves summary information for fulfillment order reverse picks that match the search criteria input.
lreadReversePickDetail	Reads the complete detailed information about a fulfillment order reverse pick including items and quantities.
createReversePick	Creates a new fulfillment order reverse pick for the specified fulfillment order.
deleteReversePick	Deletes a fulfillment order reverse pick.
updateFulfillmentOrderReversePick	Updates the items and quantities on a fulfillment order reverse pick. This operation requests an activity lock.
confirmReversePick	Confirms the fulfillment order reverse pick completing the process and assigning the inventory back to a location within the store system.
lookupCustomAttributeAdmins	Retrieves the custom attribute administration information that describes what customized attributes are available on the fulfillment order reverse pick.

Standard Usage

Reverse Picking is used to take reserved quantities and place them back into available inventory. The user can create a reverse pick with `createReversePick`. The quantities to return

can be updated through the `updateFulfillmentOrderReversePick` operation and when the reverse pick is ready, it can be finalized with `confirmReversePick` which moves reserved inventory back into available inventory.

InventoryAdjustment

The following operations are available within the InventoryAdjustment web service.

Operation	Description
lookupInventoryAdjustmentReason	Retrieve a complete list of adjustment reasons that can be used when updating or saving an inventory adjustment. Reason codes are attached to each line item.
lookupNonSellableQuantityType	Retrieve a complete list of non-sellable quantity types. These codes indicate the reason that unavailable inventory in unavailable.
lookupInventoryAdjustmentHeader	Retrieve summary information about inventory adjustment transactions based on the search criteria sent.
readInventoryAdjustmentDetail	Retrieve the complete detailed information about an inventory adjustment, including its item information, based on a unique reference/id.
saveInventoryAdjustment	Creates or updates the information about an inventory adjustment in the data store. You can alter information about items and quantities using this operation. This operation requires having an activity lock.
confirmInventoryAdjustment	Confirms the inventory adjustment, updating all the inventory positions, and closing the adjustment.
saveAndConfirmInventoryAdjustment	Performs the functionality of saveInventoryAdjustment and immediately thereafter performs the confirmInventoryAdjustment functionality. See those operations.
cancelInventoryAdjustment	Cancel an inventory adjustment. This can only be done prior to the inventory adjustment being confirmed.
lookupCustomAttributeAdmins	Retrieves the custom attribute administration information that describes what customized attributes are available on the inventory adjustment.

Standard Usage

A new inventory adjustment can be created using the `saveInventoryAdjustment` operation. Alternatively, the user can `lookupInventoryAdjustmentHeader` to find a specific inventory adjustment to work on. Either way, `saveInventoryAdjustment` can be used to update the information on an open adjustment. The `lookupInventoryAdjustmentReasons` will retrieve the reasons codes that need to be assigned to items when you update an adjustment. When the adjustment contains all the information you need, the `confirmInventoryAdjustment` operation will finalize the inventory adjustment and shift the inventory appropriately.

ItemBasket

The following operations are available within the Item Basket web service.

Operation	Description
lookupItemBasketHeaders	Retrieve a list of item basket headers based on search criteria which contain summary information about the item basket.
lookupItemBasketTypes	Retrieve a complete list of item basket types to use when creating a new item basket.

Operation	Description
createItemBasket	Creates a new item basket.
readItemBasket	Retrieve the complete detailed information about an item basket based on an identifier.
deleteItemBasket	Cancels an item basket. The basket will no longer be usable and will be marked for eventual purge from the data store. This operation requires an activity lock.
saveItemBasket	Updates an item basket. This operation requires an activity lock.
copyItemBasket	Creates a new item basket with the same information as an existing item basket.
confirmItemBasket	Moves the item basket to a completed state and allows it to be used within logic throughout the system. This operation requires an activity lock.
lookupCustomAttributeAdmins	Retrieves the custom attribute administration information that describes what customized attributes are available on the item basket.

Standard Usage

A new item basket can be created using the `saveItemBasket` operation. Alternatively, the user can `lookupItemBasketHeader` and `readItemBasket` to find a specific item basket to work on. Either way, `saveItemBasket` can be used to update the information on an item basket. When the item basket contains all the information you need, the `confirmItemBasket` operation will finalize the item basket and make it available to use in other areas of the system.

OrderRequest

The following operations are available within the Order Request web service.

Operation	Description
lookupOrderRequestHeader	Retrieves store order request headers based on the query criteria.
readOrderRequest	Retrieves detailed information about a store order request.
createOrderRequest	Creates a new store order request.
updateOrderRequest	Updates an existing store order request.
approveOrderRequest	Approve a store order request.
cancelOrderRequest	Cancels a store order request.
lookupDeliveryTimeSlot	Retrieves delivery time slots.
lookupOrderContext	Retrieves contexts available for store order requests.
lookupOrderArea	Retrieves store order request areas that could be used for restriction.
lookupCustomAttributeAdmins	Retrieves all the custom attributes admins configured for store order requests.

Standard Usage

A new store order can be created using the `createOrderRequest` operation. The information about store order can be read by `readOrderRequest`. The store order can be updated using `updateOrderRequest` and can be approved using `approveOrderRequest` or can be canceled using `cancelOrderRequest`. The `lookupOrderRequestHeader` is used to find the store orders.

POSTransaction

The following operations are available within the POSTransaction web service.

Operation	Description
processPOSTransactions	Processes a point-of-sale transaction or transactions through an asynchronous process. This is designed to optimize the processing at 500 PosTrnItm (across any number of transactions).

Standard Usage

POS may integrate its transactions to EICS using this web service. The service processes point-of-sale transactions through an asynchronous process. This service has a default limit of 1000 total PosTrnItms, though they may be distributed between any number of actual PosTrn transactions. Exceeding this limit causes a web service fault to occur. However, the web service is optimized for speed at greater than 400 and less than 500 total PosTrnItms per service call. These transactions may belong to multiple store identifiers. The processing operation validates the input, parses the payload information, creates a POSTransaction object within EICS, and stores these records to be processed later. See Sales Integration for additional information.

REST Web Service

A REST web service for POSTransaction exists and is the preferred service to use in order to process point-of-sale transactions (see REST WEB Services). This SOAP based web service will be deprecated and eventually removed.

ProductGroup

The following operations are available within the ProductGroup web service.

Operation	Description
lookupProductGroupHeader	Retrieves list of summary information about a product group that match the search criteria input.
readProductGroup	Retrieves the detailed information about a single product group based on its unique reference.
saveProductGroup	Creates or updates a product group. The input contains all the detailed information about the product group. An activity lock is needed for this operation.

Standard Usage

With this web service, the user can create or update the contents of a product group, a collection of items associated with a certain type of grouping, such as stock counts. The user can find the product group with `lookupProductGroupHeader`, read in the entire product group with `readProductGroup` and then, if the group is still open, update the contents of the product group with `saveProductGroup`.

ProductGroupSchedule

The following operations are available within the ProductGroupSchedule web service.

Operation	Description
lookupProductGroupScheduleHeader	Retrieves list of summary information about a product group schedule that match the search criteria input.
readProductGroupSchedule	Retrieves the detailed information about a single product group schedule based on its unique reference.
saveProductGroupSchedule	Creates or updates a product group. The input contains all the detailed information about the product group schedule. An activity lock is needed for this operation.
cancelProductGroupSchedule	Cancels the product group schedule.

Standard Usage

With this web service, the user can create or update the contents of schedule, which uses a product group to generate activity within EICS. The user can find the schedule with `lookupProductGroupScheduleHeader`, read in the entire schedule with `readProductScheduleGroup` and then, if the schedule is still open, update the contents of the schedule with `saveProductGroupSchedule`.

ReplenishmentGap

The following operations are available within the ReplenishmentGap web service.

Operation	Description
lookupReplenishmentGapHeaders	Retrieves list of summary information about replenishment gaps that match the search criteria input
readReplenishmentGap	Retrieves the detailed information about a single replenishment gap based on its unique reference.
saveReplenishmentGap	Creates a new replenishment gap or updates the detailed information about a replenishment gap. If update, this operation requires an activity lock.
deleteReplenishmentGap	Deletes a replenishment gap.
lookupCustomAttributeAdmins	Retrieves the custom attribute administration information that describes what customized attributes are available on the replenishment gap.

Standard Usage

With this web service, the user can create or update the contents of replenishment gap list which can then be used in creation of shelf replenishment within EICS. A new replenishment gap list can be created using `saveReplenishmentGap`. The user can update existing replenishment gap list with `saveReplenishmentGap`, find replenishment gap lists with `lookupReplenishmentGapHeaders`, read in the entire replenishment gap list with `readReplenishmentGap` and delete a replenishment gap list with `deleteReplenishmentGap`.

RfidInventory

The following operations are available within the RfidInventory web service.

Operation	Description
deleteRfidZone	Deletes a zone within a facility. A zone cannot be deleted if RFID tags still exist within the zone.

Operation	Description
lookupRfidZones	Returns details about all the zones within a particular facility.
processRfidEvents	Processes Radio-Frequency-Identification based events.
saveRfidZone	Creates or updates the details of a facility zone.

Standard Usage

With this web service, the user can create or update RFID zones within EICS. A new RFID zone can be created using `saveRfidZone`. The user can update an existing RFID zone with `saveRfidZone`, find RFID zones with `lookupRfidZones` and delete a RFID zone with `deleteRfidZone`. The user can process RFID based events using `processRfidEvents`.

ShelfAdjustment

The following operations are available within the ShelfAdjustment web service.

Operation	Description
lookupShelfAdjustmentHeaders	Retrieves list of summary information about shelf adjustments that match the search criteria input.
readShelfAdjustment	Retrieves the detailed information about a single shelf adjustment gap based on its unique reference.
saveShelfAdjustment	Creates a new shelf adjustment or updates the detailed information about a current shelf adjustment. If update, this operation requires an activity lock.
confirmShelfAdjustment	Confirms a shelf adjustment completing the workflow and moving inventory positions.
cancelShelfAdjustment	Deletes a shelf adjustment.
lookupCustomAttributeAdmins	Retrieves the custom attribute administration information that describes what customized attributes are available on the shelf adjustment.

Standard Usage

Shelf adjustments are used to adjust the shop-floor or backroom stock in case of any discrepancy. A new shelf adjustment can be created using `saveShelfAdjustment`. The user can update existing shelf adjustment with `saveShelfAdjustment`, find shelf adjustments with `lookupShelfAdjustmentHeaders`, read in the entire shelf adjustment with `readShelfAdjustment`, cancel a shelf adjustment with `cancelShelfAdjustment` and confirm a shelf adjustment with `confirmShelfAdjustment`.

ShelfReplenishment

The following operations are available within the ShelfReplenishment web service.

Operation	Description
lookupShelfReplenishmentHeaders	Retrieves list of summary information about shelf replenishments that match the search criteria input.
readShelfReplenishment	Retrieves the detailed information about a single shelf replenishment gap based on its unique reference.
createShelfReplenishment	Creates a new shelf replenishment.

Operation	Description
updateShelfReplenishment	Updates the detailed information about a current shelf replenishment. This operation requires an activity lock.
confirmShelfReplenishment	Confirms a shelf replenishment completing the workflow and moving inventory positions.
cancelShelfReplenishment	Deletes a shelf replenishment.
lookupCustomAttributeAdmins	Retrieves the custom attribute administration information that describes what customized attributes are available on the shelf replenishment.

Standard Usage

Shelf replenishment is used to replenish shop-floor stock from backroom or delivery bay. A new shelf replenishment can be created with `createShelfReplenishment`. The user can find shelf replenishments with `lookupShelfReplenishmentHeaders`, read in the entire shelf replenishment with `readShelfReplenishment`, update the shelf replenishment with `updateShelfReplenishment`, confirm the shelf replenishment with `confirmShelfReplenishment` and cancel the shelf replenishment with `cancelShelfReplenishment`.

StockCount

The following operations are available within the StockCount web service.

Operation	Description
lookupStockCountHeaders	Retrieves list of summary information about a stock count that match the search criteria input.
readStockCountDetail	Retrieves the detailed information about a single stock count based on its unique reference. This contains a list of summary information about the child counts.
readStockCountChild	Retrieves the detailed information about a single stock count child.
activateStockCount	This activates are starts the stock counting process including taking a snapshot of current inventory positions.
completeStockCountChild	Completes the counting or recounting of a stock count child, depending on which phase the stock count is in. This process will calculate discrepancies and move the child to the next phase.
updateCountQuantities	Updates the counted or recounted quantity fields for a stock count child based on the current phase of the stock count.
lookupCustomAttributeAdmins	Retrieves the custom attribute administration information that describes what customized attributes are available on the stock count.

Standard Usage

The stock count web services are design primarily to export information for third party counting. You first lookup the headers, choose your stock count, and then retrieve all the details for the stock count. These details do not contain item information but rather a list of child count references. You can use these references to grab the full details of a child count which includes items and quantities, and then update those quantities.

REST Web Service

A StockCount REST web service exists that allows for the snapshot of a stock count (see REST WEB Services).

Store

The following operations are available within the Store web service.

Operation	Description
lookupAutoReceiveStore	Retrieves all stores that allow auto-receiving of inventory from the input store.
lookupAssociatedStore	Retrieves all stores that are associated to the input store. They are sometimes called buddy stores.
lookupStoresInTransferZone	Retrieves all stores in the same transfer zone as the input store.
readStoreDetail	Retrieves the detailed information about a single store from the input unique reference.

Standard Usage

The Store web service is used to retrieve information about stores. There are no updates. They are used to determine such information as whether you can ship to certain stores (such as those in transfer zones).

StoreFulfillmentOrder

The following operations are available within the StoreFulfillmentOrder web service.

Operation	Description
lookuFulfillmentOrdersHeaders	Retrieves summary information for fulfillment orders that match the search criteria input.
readFulfillmentOrderDetail	Reads the complete detailed information about a fulfillment order including items and quantities.
createFulfillmentOrderDetail	Creates a new fulfillment order with detailed information, including items and quantities.
cancelFulfillmentOrderDetail	Cancels quantities on a fulfillment order. This may cancel the entire order or just reduce or cancel quantities for specific items.
rejectFulfillmentOrder	Rejects the fulfillment order indicating that the store will be unable to fulfill that order.
lookupCustomAttributeAdmins	Retrieves the custom attribute administration information that describes what customized attributes are available on the fulfillment order.

Standard Usage

Unlike some of the other web services, fulfillment order is not managed within EICS. Instead, EICs manages the picking and delivery, but the order itself is managed by an external order management system.

Oracle Retail Order Broker (OB) calls SIOCS for inventory availability.

Web services are supplied to find and read the details of a fulfillment order, but updates are not allowed. Instead, the external system uses `createFulfillmentOrderDetail` to notify EICS of a

new order to ship, `cancelFulfillmentOrderDetail` to reduce or cancel quantities (note that they cannot be increased) or call `rejectFulfillmentOrder` to notify EICS that the order has been rejected.

StoreInventory

The following operations are available within the StoreInventory web service.

Operation	Description
lookupAvailableInventory	Retrieves basic availability information for multiple items at multiple locations. Only transaction-levels items are processed (UPCs are not allowed) and only current inventory is returned. The service supports up to 200 items at 150 locations.
lookupAvailableInventoryAllStores	Retrieves basic availability information for a single item at all store locations. Only transaction-levels items are processed (UPCs are not allowed) and only current inventory is returned.
lookupAvailableInventoryAllWarehouses	Retrieves inventory information for a single item at multiple warehouses. Only transaction-level items are processed, and only current inventory is returned.
lookupInventoryInStore	Retrieves a broad set of inventory information for several items at several stores, broken down into various inventory groupings.
lookupInventoryInTransferZone	Retrieves a broad set of inventory information for items within the specific transfer zone, broken down into various inventory groupings.
lookupInventoryForBuddyStores	Retrieves a broad set of inventory information for associated or buddy stores, broken down into various inventory groupings.
lookupFutureInventory	Retrieves the future inventory information (such as inbound, ordered quantities and expected dates) for an item and store location.

Standard Usage

The StoreInventory is meant to retrieve inventory position information. Available inventory lookups are much smaller and quicker to respond than full inventory lookups. Future inventory is separated from current positions as it is much more time consuming to retrieve. Those who access the web services should consider the purpose before choosing which operation to use.

REST Web Service

An InventoryInquiry REST web service exists for inventory lookup and is the preferred service to use in order to retrieve inventory information (see REST WEB Services). This SOAP based web service will be deprecated and eventually removed.

StoreInventoryISN

The following operations are available within the StoreInventoryISN web service.

Operation	Description
lookupIsnTypes	Returns a complete list of Item Scan Number types.
lookupIsn	Returns details about matching Item Scan Numbers in store inventory.
createIsn	Create a new Item Scan Number without changing store inventory.
updateIsn	Updates an existing Item Scan Number without changing store inventory.

Operation	Description
deleteIsn	Deletes an Item Scan Number without changing store inventory.
lookupCustomAttributeAdmins	Retrieves all the custom attribute admins configured for ISNs.

Standard Usage

This web service is used to create, update, or delete ISN in store inventory. An item scan number is any number meant to be scanned to find an item, and potentially a Unique Identification Number, that is not already an item, UPC, UIN, VPN, or other value. Items Scan Numbers are only used to find information and are not tracked as inventory.

StoreInventoryUIN

The following operations are available within the StoreInventoryUIN web service.

Operation	Description
createUIN	Create a new UIN without changing store inventory.
generateUIN	Generate new UINs without changing store inventory.
lookupUINDetails	Returns details about all the UINs in store inventory for a particular item and store. This is limited to 1000 UINs for a particular item and store.
readUINDetail	Returns details about a UIN in store inventory. A UIN reference is not unique, so this may return detailed information for UINs across multiple items.
updateUIN	Updates an existing UIN without changing store inventory.

Standard Usage

This web service is used to create, generate, update, find, or read UINs in store inventory.

StoreItem

The following operations are available within the StoreItem web service.

Operation	Description
lookupItemHeaderByItem	Retrieves list of summary information about an item that match the item-based search criteria input.
lookupItemHeaderBySource	Retrieves list of summary information about an item that match the source or location-based search criteria input.
lookupItemHeaderByUDA	Retrieves list of summary information about an item that match the UDA (User Defined Attribute)-based search criteria input.
lookupItemHeaderByInventory	Retrieves list of summary information about an item that match the inventory-based search criteria input.
lookupItemCfa	Retrieve a list of custom flexible attributes for the specified item and store.
lookupItemUda	Retrieve a list of user defined attributes for the specified item and store.
readItemDetail	Retrieves the complete detailed information a single item based on its unique reference.

Operation	Description
lookupRelatedItem	Retrieves a list of summary information about items related to the item used as input criteria.
saveItemImage	Inserts a new display image or QR code image for the specified item. The service returns immediately, and the information is processed asynchronously.

Standard Usage

This web service is used to find items and retrieve information about items. The only exception is the ability to create new image-based information about an item.

StoreItemPrice

The following operations are available within the StoreItemPrice web service.

Operation	Description
lookupItemPriceHeader	Retrieve a summary list of item price information based on input criteria. This only retrieves information known to EICS and has no access to a pricing system.
readItemPrice	Retrieves the full details a single item price record based on its unique reference.
lookupItemPriceOnEffectiveDate	Retrieves the item price of an item for a specific date.

Standard Usage

This web service is used to retrieve information about prices that are known to EICS. Integration with pricing systems updates EICS information about item prices on a continual basis. These web services give a view into EICS information only.

StoreNotification

The following operations are available within the StoreNotification web service.

Operation	Description
createNotification	Creates a new notification within the system. These notifications are displayed in the client applications.

Standard Usage

This web service is designed for external system that handle related activities to EICS. With this web service, they can send notifications into EICS of activity that needs to take place based on something that has occurred in another system.

StoreShipmentManifest

The following operations are available within the StoreShipmentManifest web service.

Operation	Description
closeManifest	Closes the manifest shipments.

Standard Usage

This web service is designed to close manifest shipments. All manifest shipments matching the input criteria, such like carrier code, and carrier service code will be closed.

StoreShipmentReason

The following operations are available within the StoreShipmentReason web service.

Operation	Description
lookupAllShipmentReasons	Retrieves all the shipment reasons configured for store shipments.

Standard Usage

This web service exists to allow customers to retrieve information about shipment reasons that can be assigned to line items on outgoing shipments. The shipment based web services taking the code identifier and thus, you will need to read in these shipment reasons to be able to select and apply valid reason codes.

StoreTicket

The following operations are available within the StoreTicket web service.

Operation	Description
createTickets	Create a new group of up to 999 tickets to be managed and printed.
lookupTicketFormats	Retrieves available ticket formats for the criteria specified.

Standard Usage

The `createTickets` operation is used to create a new group up to 999 tickets to be managed and printed. The ticket formats can be retrieved using `lookupTicketFormats` operation based on the criteria specified.

StoreTransfer

The following operations are available within the StoreTransfer web service.

Operation	Description
lookupTransferHeader	Retrieve a summary list of transfers that matches the input criteria.
lookupTransferContext	Retrieves all the transfer context options available to assign to a transfer.
readTransfer	Retrieves the detailed information about transfer, including its items and quantities, based on a unique reference.
createTransferRequest	Creates a brand new transfer request (Location 1 requesting a transfer from Location 2).

Operation	Description
saveTransferRequest	Updates a transfer request allowing user to change items and quantities. This must be done prior to requesting it, which finalizes the transfer request. This requires an activity lock.
createTransfer	Generates a new transfer that you can add details to. The saveTransfer method must be used to update details such as items and quantities of the transfer.
saveTransfer	Updates a previously approved transfer item and quantity details. This operation requires an activity lock.
saveTransferApproval	Updates items and quantities on a transfer in requested status that is currently in the process of being approved but has not yet been approved. This operation requires an activity lock.
requestTransfer	Updates the status to Requested, finally the transfer request. This allows the opposite location to view the new request for transfer of goods. This operation requires an activity lock.
approveTransfer	Approves a transfer request converted the transfer request into an approved transfer. This operation requires having an activity lock.
rejectTransfer	Rejects a transfer in request status which prevents the transfer request from becoming a transfer. This operation requires having an activity lock.
cancelTransfer	Cancels an approved transfer. This operation requires having an activity lock.
closeTransfer	Closes a processed or partially processed transfer finalizing the state of the transfer. This operation requires having an activity lock.
lookupCustomAttributeAdmins	Retrieves the custom attribute administration information that describes what customized attributes are available on the transfer.

Standard Usage

The process is started by one store creating a transfer request from a shipping store using `createTransferRequest`. The requesting store can continue modifying the transfer request using `saveTransferRequest` until it is ready to notify the shipping store, when it then uses the `requestTransfer` to send the request to the shipping store. The shipping store can then begin picking items for the transfer and updating the transfer using the `saveTransferApproval` operation. When all the quantities the shipping store are willing to ship are determined, the shipping store uses `approveTransfer` to finalize the approval of the transfer. Alternatively, they can choose to reject the transfer using `rejectTransfer`. It is possible for a shipping store to create a transfer document without going through the request and approval process by using `createTransfer` and `saveTransfer`.

TransferDelivery

The following operations are available within the TransferDelivery web service.

Operation	Description
lookupTransferDeliveryHeaders	Retrieves basic information about one or more transfer deliveries that match the criteria specified. This operation is used to find a delivery arriving at the store.
readTransferDeliveryDetail	Retrieves the entire set of information about a transfer delivery header based on the identifier you pass to it.

Operation	Description
updateTransferDelivery	Updates the header information on a transfer delivery. This operation requires an activity lock.
receiveTransferDelivery	Receives all the currently open and active containers on a transfer delivery by defaulting quantities into all the unreceived items. This does not move inventory, only defaults quantities. This operation requires an activity lock.
confirmTransferDelivery	Confirms a transfer delivery receiving the goods into inventory and updating all the inventory positions. This moves the transfer delivery to a completed status. This operation requires an activity lock.
lookupTransferDeliveryContainerHeaders	Retrieves summary information about every container on a transfer delivery based on the unique delivery reference.
readTransferDeliveryContainerDetail	Reads the entire details of a container including items and quantities based on a unique container reference.
createTransferDeliveryContainer	Generates a new container on the transfer delivery and returns a reference to use so that items and quantity can be added later.
updateTransferDeliveryContainer	Updates the items and quantities on a transfer delivery container. This operation requires an activity lock.
receiveandConfirmTransferDeliveryContainer	It first defaults receiving quantity on the items within the container and then executes the same locking as the confirmTransferDeliveryContainer. This operation requires an activity lock.
confirmTransferDeliveryContainer	Confirms a transfer delivery container as received and updates all the inventory positions. This operation requires an activity lock.
cancelTranferDeliveryContainer	Cancels a transfer delivery container moving it to missing status. Changes cannot be made to a canceled container.
openTransferDeliveryContainer	Re-opens an already confirmed container moving it back into in-progress status.
lookupTransferDeliveryOrders	Retrieves any customer orders associated with the transfer delivery based on the delivery's unique reference.
lookupMisdirectedTransferDeliveryContainers	Retrieves summary information about containers that may have been misdirected based on a set of search criteria as input into the operation.
lookupCustomAttributeAdmins	Retrieves the custom attribute administration information that describes what customized attributes are available on the transfer delivery.

Standard Usage

After reading a transfer delivery using `lookupTransferDeliveryHeader`, you can read the header detail with `readTransferDelivery` or container list with `lookupTransferDeliveryContainers`. You can then use `updateTransferDelivery` to update header attributes and `updateTransferDeliveryContainer` to update items and quantities in the container. To quickly receive the quantities, `receiveTransferDeliveryContainer` automatically fills in quantities, and when quantities are entered `confirmTransferDeliveryContainer` finalizes the container (and if appropriate configurations and business rules apply) immediately updates the inventory. If `receiveTransferDelivery` or `confirmTransferDelivery` is used, then all containers will either be received or confirmed respectively.

TransferShipment

The following operations are available within the TransferShipment web service.

Operation	Description
lookupTransferShipmentHeader	Retrieves basic information about one or more transfer shipments that match the criteria specified. This operation is used to find a shipment.
readTransferShipmentDetail	Retrieves the entire set of information about a transfer shipment header based on a unique reference.
createTransferShipment	Creates a new and empty transfer shipment and returns a reference to the shipment.
saveTransferShipment	Updates the information on a transfer shipment header.
submitTransferShipment	Submits the transfer shipment for review before final dispatch.
cancelSubmittedTransferShipment	Cancels the submission of the transfer shipment for review.
dispatchTransferShipment	Dispatches a transfer shipment. This moves the shipment to dispatched state and updates the inventory. A transfer shipment cannot be modified after dispatch. Dispatch should occur only after all containers are confirmed.
cancelTransferShipment	Cancels a transfer shipment.
lookupTransferShipmentContainer	Finds all the containers on a specific shipment and retrieves basic identification information about each container.
readTransferShipmentContainer	Reads the specific and complete contents of a container.
createTransferShipmentContainer	Creates a new transfer shipment container on the shipment and returns a reference to it.
saveTransferShipmentContainer	Updates the information about a transfer shipment container including adding and removing items and quantities.
confirmTransferShipmentContainer	Confirms that a transfer shipment container is ready for shipment and marks the container as no longer editable.
cancelTransferShipmentContainer	Cancels a transfer shipment container on the shipment.
openTransferShipmentContainer	Re-opens a confirmed container on a shipment prior to the shipment being dispatched so that changes can be made to the container.
lookupCustomAttributeAdmins	Retrieves the custom attribute administration information that describes what customized attributes are available on the transfer shipment.

Standard Usage

To create a shipment for a transfer document, lookup the transfer shipment using `lookupTransferShipmentHeader`. If it does not exist, you may create one for the document using `createTransferShipment`. Create a container on the shipment using `createTransferShipmentContainer` and update the container with items and quantities using `saveTransferShipmentContainer`. Confirm the container using `confirmTransferShipmentContainer`. Repeat the process for each container as needed. Once all containers are confirmed, if configured to require submittal, submit the shipment using `submitTransferShipment` and finally, dispatch the shipment using `dispatchTransferShipment`. Dispatching the shipment finalizes the shipment and relieves the inventory.

VendorDelivery

The following operations are available within the VendorDelivery web service.

Operation	Description
lookupVendorDeliveryHeaders	Retrieves basic information about one or more vendor deliveries that match the criteria specified. This operation is used to find a delivery from a supplier.
lookupPurchaseOrderHeaders	Retrieves basic information about one or more purchase orders that match the criteria specified.
readVendorDeliveryDetail	Retrieves the entire set of information about a vendor delivery header based on a unique reference.
createVendorDelivery	Generate a new vendor delivery heaver and returns a referenced to the delivery.
updateVendorDelivery	Updates the information on a vendor delivery header. This does not include containers, items, or quantities. This operation requires an activity lock.
receiveVendorDelivery	Updates the quantities on a vendor delivery filling in any unreceived items within the containers of the delivery with a default value. It "receives" missing quantities, but no inventory positions are updated. This operation requires an activity lock.
confirmVendorDelivery	Confirms the vendor delivery updating inventory positions and completing the delivery. This operation requires an activity lock.
rejectVendorDelivery	Rejects the vendor delivery placing it in rejected status. This operation requires an activity lock.
cancelVendorDelivery	Cancels the vendor delivery placing it in canceled status. This operation requires an activity lock.
lookupVendorDeliveryContainerHeaders	Retrieves summary information about every container on a vendor delivery based on the unique delivery reference.
readVendorDeliveryContainerDetail	Reads the entire details of a container including items and quantities based on a unique container reference.
createVendorDeliveryContainer	Generates a new container on the vendor delivery and returns a reference to use so that items and quantity can be added later.
updateVendorDeliveryContainer	Updates the items and quantities on a vendor delivery container. This operation requires an activity lock.
confirmVendorDeliveryContainer	Confirms a vendor delivery container as received and updates all the inventory positions. This operation requires an activity lock.
cancelVendorDeliveryContainer	Cancels a vendor delivery container moving it to missing status. Changes cannot be made to a canceled container.
openVendorDeliveryContainer	Open Vendor delivery container. This will re-open a container after receipt allowing it to be received again.
lookupVendorDeliveryOrders	Retrieves any customer orders associated with the vendor delivery based on the delivery's unique reference.
lookupVendorDeliveryAdjustments	Retrieves any external receipt adjustments that exist for the delivery based on the specified unique reference.
cancelSubmitVendorDeliveryContainer	Opens a submitted container for further updates, moving the status to in-progress.
submitVendorDeliveryContainer	Moves the status of the container to submitted and prevents further updates. The container may still be confirmed. No inventory positions are updated via this operation.

Operation	Description
lookupCustomAttributeAdmins	Retrieves the custom attribute administration information that describes what customized attributes are available on the vendor delivery.

Standard Usage

After reading a vendor delivery using `lookupVendorDeliveryHeader`, you can read the header detail with `readVendorDelivery` or container list with `lookupVendorDeliveryContainers`. Use `updateVendorDelivery` to update header attributes and `updateVendorDeliveryContainer` to update items and quantities in the container. To quickly receive the quantities, `receiveVendorDeliveryContainer` automatically fills in quantities, and when quantities are complete `confirmVendorDeliveryContainer` finalizes the container and if appropriate configurations and business rules apply, immediately updates the inventory. If `receiveVendorDelivery` or `confirmVendorDelivery` is used, then all containers will either be received or confirmed respectively. Re-opening a container can be done using `openVendorDeliveryContainer`. To prevent further updates to the container without confirming it, use `submitVendorDeliveryContainer`. Submitted container can be re-opened and moved to in-progress status for further updates using `cancelSubmitVendorDeliveryContainer`.

VendorReturn

The following operations are available within the VendorReturn web service.

Operation	Description
lookupVendorReturnHeader	Retrieves basic information about one or more vendors return documents that match the criteria specified.
readVendorReturnDetail	Retrieves the entire set of information about a vendor return, including items and quantities, based on a unique reference.
saveVendorReturn	Updates the entire set of information about a vendor return, including items and quantities. This operation requires an activity lock.
approveVendorReturn	This marks an in-progress vendor return as approve for shipment. This operation requires an activity lock.
cancelVendorReturn	Cancels a vendor return indicating no further items and quantities should be shipped for the return.
closeVendorReturn	Closes a vendor return document moving it from in-progress to canceled, rejected, or complete status depending on the state of shipped quantities.
lookupCustomAttributeAdmins	Retrieves the custom attribute administration information that describes what customized attributes are available on the vendor return.

Standard Usage

The user may access `lookupVendorReturnHeader` to find vendor returns to deal with. Once the proper vendor return is found, `readVendorReturnDetail` will retrieve all the details of the vendor return including items and quantities. The `saveVendorReturn` operation is then used to update quantities that are expected to ship. Once the vendor return reaches its final state, the operation `approveVendorReturn` will approve the return and get it ready for shipment.

VendorShipment

The following operations are available within the VendorShipment web service.

Operation	Description
lookupVendorShipmentHeaders	Retrieves basic information about one or more vendor shipment headers that match the criteria specified.
lookupReturnContext	Retrieves all the context options that are available to assign to a vendor return shipment.
readVendorShipmentDetail	Retrieves the detailed information about a vendor return header based on a unique reference. It does not include information about containers or items.
saveVendorShipment	Creates a new vendor shipment header if not identifying reference is set or updates the vendor shipment header information if a unique reference is sent as part of the date. When used as an update, an activity lock is needed.
submitVendorShipment	Submits the vendor shipment for review before final dispatch.
cancelVendorShipmentSubmission	Cancels the submission of the vendor shipment for review.
cancelVendorShipment	Cancels a vendor shipment. This moves the shipment to canceled status. Changes cannot be made to a canceled shipment.
dispatchVendorShipment	Dispatches a vendor shipment. This moves the shipment to dispatched state and updates the inventory. A vendor shipment cannot be modified after dispatch. Dispatch should occur only after all containers are confirmed. This operation requires an activity lock.
closeVendorShipment	Closes a vendor shipment using business logic to determine its final state. It cancels the shipment of remaining quantities. Changes cannot be made after a shipment is closed.
lookupVendorShipmentContainerHeaders	Retrieves summary information about all containers within a vendor shipment based on the unique reference of the shipment.
readVendorShipment ContainerDetail	Reads the specific details, including items and quantities, about a container specified by its unique reference.
saveVendorShipmentContainer	Update the details of a container, including items and quantities. This operation requires an activity lock.
confirmVendorShipmentContainer	Confirms that the container is ready for shipment. A confirmed container cannot be modified. This operation requires an activity lock.
cancelVendorShipmentContainer	Cancels a container on the shipment removing it from the shipment.
openVendorShipmentContainer	Opens a confirmed container placing it back into in-progress status so that items can be added or removed from the container.
lookupCustomAttributeAdmins	Retrieves the custom attribute administration information that describes what customized attributes are available on the vendor shipment.

Standard Usage

To create a shipment for a vendor return document, lookup the vendor shipment using `lookupVendorShipmentHeader`. If it does not exist, create one using `createVendorShipment`. Next, create a container on the shipment using `createVendorShipmentContainer`. Update the container with items and quantities using `saveVendorShipmentContainer`. Confirm the container using `confirmVendorShipmentContainer`. Repeat the process for each container as needed. Once all containers are confirmed, if configured to require submit, then submit using `submitVendorShipment` or dispatch the shipment using `dispatchVendorShipment`. Dispatching the shipment finalizes the shipment and relieves the inventory.

Enterprise Documentation

Full web service API document in the form of web services description language files can be downloaded in .zip format. Version specific files are available under ‘Web Services Description Language Files’ section within MOS Document 2614551.1.