Document Attributes

| Attribute | Value |
| --- | --- |
| Application ID /  Application Name | • OPUS- MOTS ID: 18257  • OPUS Mobile- MOTS ID: 18257 |
| Owner | Madas, Arun (am6489) |
| Owner Contact Information |  |
| Other Attribute | Bogoju, Mrudula (mb009h)  Kandi, Pavan (pk5958)  Nandyala, Pani (pn0886) |

## 

Revision History

The following table lists the revision history of this document:

| Author | Date | Version # | Revision Description | |
| --- | --- | --- | --- | --- |
| am6489 | 05/11/2015 | 0.1 | Initial Draft | |
| Am6489 | 06/01/2015 | 0.2 | | Incorporated changes after the HLD walk through review from Mark Price (MP2591 - Yoda) regarding Yoda APIs. Incorporated changes after the HLD walk through review from Chris Pierce (PM) regarding date calculation from holidays. Incorporated several other changes as discussed in the walk through session. CRs etc. | |
| Am6489 | 06/21/2015 | 0.3 | | Updated for LD support HLD elements, Store flag names shortened/updated. | |
| Am6489 | 06/27/2015 | 0.4 | | Updated with review comments from Herve (hj058j) | |
| Am6489 | 07/13/2015 | 0.5 | | Updated with walk through on consolidated order. | |
| Am6489 | 08/16/2015 | 0.6 | | Updates based on last minute CR updates to project, cascading recent updates to SR. | |

Table of Contents

[Overview 3](#_Toc406166738)

[Problem Statement 3](#_Toc406166739)

[Design Decisions 3](#_Toc406166741)

[Functional Overview 6](#_Toc406166742)

[Middleware Design 7](#_Toc406166743)

[External Interfaces 8](#_Toc406166744)

[Component Diagram 8](#_Toc406166745)

[Performance Design 9](#_Toc406166746)

[Security Design 9](#_Toc406166747)

[UI Design 9](#_Toc406166748)

[UI Design - Wireframe 9](#_Toc406166749)

[UI Design - Screen Element Definition and Data Mapping 10](#_Toc406166750)

[UI Action Processing - Sequence Diagram 11](#_Toc406166751)

[Asynchronous/Batch Processing - Sequence Diagram 13](#_Toc406166752)

[Externalized Configuration 13](#_Toc406166753)

[Database Tier Design 14](#_Toc406166754)

[Logical Data Model 14](#_Toc406166755)

[Database Interfaces 14](#_Toc406166756)

[Batch Interfaces 14](#_Toc406166757)

[Replication 14](#_Toc406166758)

[Alternative Designs 14](#_Toc406166759)

[Assumptions/Risks 14](#_Toc406166760)

[Pre-Production Disaster Recovery Planning 14](#_Toc406166761)

[Other Plans and References 14](#_Toc406166762)

[Acceptance & Approvals 15](#_Toc406166763)

## Overview

The High Level Design (HLD) describes how an application will implement the architectural concept and meet the requirements allocated to that application. The High Level Design describes the functionality the application will provide, the overall design for providing that functionality and meeting the nonfunctional requirements, and the rationale for choosing that design. The High Level Design also describes how the application will cooperate and interface with other applications to provide an integrated solution that achieves the architectural concept.

The High Level Design covers these topics:

* Problem Statement
* Design Decisions
* Alternative Designs
* Assumptions/Risks
* Other Plans/References

## Problem Statement

## The purpose of this project is to implement enhancements to the Wireless Customer Agreement (WCA) flow in OPUS Mobile and OPUS. These enhancements were requested by Legal and impact the WCA acceptance and Print functionality, providing more flexibility in the flow and a better experience for the customer and user. This project also includes updated labels and minimal updates to the acceptance text. This document

## Design Decisions

|  |  |  |
| --- | --- | --- |
| **Req. ID** | **Design Element** | **Trace-To** |
| 270938e\_OPUS\_OM\_HLD\_01 | HLD44 OPUS/OM shall support single ‘on/off’ project flag 270938e \_Sacred7\_Iconic\_DF enable/disable all 270938e functionality for both OPUS & OM to support the newly added functionality. Default value of the flag will be “OFF” as we deploy in OFF Mode.  OPUS/OM shall use single OM jsp’s for all new screens for both OPUS and OM while utilizing separate existing opus screens for non-new screens for updates. | 270938e.SR.OPUS.80 |
| **1 . Promise Date Handling** | | |
| 270938e\_OPUS\_OM\_HLD\_02 | HLD45 OPUS/OM shall retrieve and locally store the promise date/date range from OrderTrack (OT) via the appropriate real-time API/Atlas notification. Display the promise dates for serialized and non-serialized item display in DF Activations, DF Upgrades, SA DF Orders, DF BRE Exchange flows.  ChannelCode = OPUSDF  Atlas Queue = q/atlas/opus/expshipdatesnotification  OPUS/OM shall only consume ChannelCode = OPUSDF  Refer to [component diagram](#highlevelcomppromisedate) and [components overview](#highlevelflowcomppromisedate) | 270938e.SR.OPUS.01 |
| 270938e\_OPUS\_OM\_HLD\_03 | HLD46 When project flag is ON, Display Promise dates for Serialized and Non-serialized DF Items in impacted flows (DF Activations, DF Upgrade, SA DF Orders, DF BRE Exchange flows). Existing functionality will be replaced with new functionality when project flag is turned on for display. | 270938e.SR.OPUS.02  270938e.SR.OPUS.03 |
| 270938e\_OPUS\_OM\_HLD\_04 | HLD47 OPUS/OM will store and support display of Atlas notifications that are :   1. Date Range - Promise Dates 2. Day Range - Promise Dates   For Day Range, Translation to dates will be done on OPUS side considering National Holidays, Excluding Saturday and Sundays for business day calculations. Day range to Date range translation will start counting from the next valid business day of the day of order. we will start counting from the next business day of placing order as day 1, so if you place order on 7/17/2015 we will count 7/17 (day0), exclude weekend (7/18,7/19), 7/20 (day1), 7/21(day2), 7/22(day3), 7/23(day4), 7/24(day5) and show translation of 3-5 business days as 7/22-7/24. Orders placed during the weekend or holiday will have next business day as day1. For example : Order placed on 08/15/2015 (Saturday) with a day range of 2-3 business days applicability will translate to 08/18-08/19 (08/15 (sat) excluded, 08/16 (sun) excluded, 08/17 (day1), 08/18(day2), 08/19(day3))  Holidays are manually loaded to opus and following list is for 2015 calendar year  12/25/2015 Christmas  11/26/2015 Thanksgiving  9/7/2015 Labor Day  7/3/2015 July 4th  5/25/2015 Memorial Day  1/1/2015 New Years  If no holidays are present, they will not be used during calculations. The process of loading future holiday days into database is a manual process and data will be inserted with DB scripts into QC and then to prod or as needed BAU DB insertion manual process. | 270938e.SR.OPUS.04 |
| sample1.jpg  sample2.jpg | | |
| 270938e\_OPUS\_OM\_HLD\_05 | HLD48 OPUS/OM to reconcile promise date and associated date information as sourced from OT/Atlas (EST) against OPUS/OM order date/timestamp information to display designated or default promise date/date range. OPUS/OM to store the date time in UTC formats and do appropriate translation. OPUS receives OT/Atlas explicit dates in (EST) timezone and stores in the UTC (timezone) and displays in the store timezone in the UI. Since no explicit timestamps are showin in the UI timezones might not apply for explicit dates, but when calculating day ranges to date ranges timezone calculations will apply. | 270938e.SR.OPUS.05 |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | SKU 123 Specific : Scenario 1 : Use Matching Dates (All timestamps in UTC) | | | | | | | SKU | Order Start Date | Order End Date | ShipStart Date | ShipEnd Date | ModifiedTime | | 123 | 05/01/2015 12:00:00 AM | 05/31/2015 12:00:00 AM | 07/25/2015 | 08/02/2015 | 04/25/2015 10:20:13 AM | | SKU | Order Date | ShipStart Date | ShipEnd Date |  |  | | 123 | 05/01/2015 10:00:00 AM | 07/25/2015 | 08/02/2015 |  |  | | SKU 123 Specific : Scenario 2 : Display Single Dates (All timestamps in UTC) | | | | | | | SKU | Order Start Date | Order End Date | ShipStart Date | ShipEnd Date | ModifiedTime | | 123 | 05/01/2015 12:00:00 AM | 05/31/2015 12:00:00 AM | 07/25/2015 | 07/25/2015 | 04/26/2015 10:20:13 AM | | SKU | Order Date | ShipStart Date | ShipEnd Date |  |  | | 123 | 05/01/2015 10:00:00 AM | 07/25/2015 |  |  |  | | SKU 123 Specific : Scenario 3 : Time Based Validation for Order Date | | | | | | | SKU | Order Start Date | Order End Date | ShipStart Date | ShipEnd Date | ModifiedTime | | 123 | 05/01/2015 12:00:00 AM | 05/01/2015 3:00:00 AM | 07/25/2015 | 08/02/2015 | 04/25/2015 10:20:13 AM | | SKU | Order Date | ShipStart Date | ShipEnd Date |  |  | | 123 | 05/01/2015 1:00:00 AM | 07/25/2015 | 08/02/2015 |  |  | | SKU 123 Specific Expiry : Scenario 4 : Past Date Validity, use Channel Default | | | | | | | SKU | Order Start Date | Order End Date | ShipStart Date | ShipEnd Date | ModifiedTime | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 05/10/2015 | 05/13/2015 | 04/25/2015 10:20:13 AM | | BASE | 04/01/2015 12:00:00 AM | 04/01/2023 12:00:00 AM | 3 | 5 | 03/25/2015 12:00:00 AM | | SKU | Order Date | ShipStart Date | ShipEnd Date |  |  | | 123 | 05/14/2015 10:00:00 AM | 05/19/2015 | 05/21/2015 |  |  | | SKU 123 Specific : Scenario 5 : Use Day Ranges | | | | | | | SKU | Order Start Date | Order End Date | ShipStart Date | ShipEnd Date | ModifiedTime | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 3 | 5 | 04/25/2015 10:20:13 AM | | SKU | Order Date | ShipStart Date | ShipEnd Date |  |  | | 123 | 05/14/2015 10:00:00 AM | 05/19/2015 | 05/21/2015 |  |  | | Default : Scenario 6 : Use Default | | | | | | | SKU | Order Start Date | Order End Date | ShipStart Date | ShipEnd Date | ModifiedTime | | BASE | 04/01/2015 12:00:00 AM | 04/01/2023 12:00:00 AM | 3 | 5 | 04/25/2015 10:20:13 AM | | SKU | Order Date | ShipStart Date | ShipEnd Date |  |  | | 123 | 05/14/2015 10:00:00 AM | 05/19/2015 | 05/21/2015 |  |  | | SKU 123 Multiple Entries : Scenario 7 : Pick most latest | | | | | | | SKU | Order Start Date | Order End Date | ShipStart Date | ShipEnd Date | ModifiedTime | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 06/10/2015 | 06/13/2015 | 04/30/2015 10:20:13 AM | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 05/10/2015 | 05/13/2015 | 04/25/2015 10:20:13 AM | | SKU | Order Date | ShipStart Date | ShipEnd Date |  |  | | 123 | 05/14/2015 10:00:00 AM | 06/10/2015 | 06/13/2015 |  |  | | SKU 123 Multiple Entries : Scenario 8 : Pick most latest, mixed date range/day range | | | | | | | SKU | Order Start Date | Order End Date | ShipStart Date | ShipEnd Date | ModifiedTime | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 7 | 10 | 04/30/2015 10:20:13 AM | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 06/10/2015 | 06/13/2015 | 04/29/2015 10:20:13 AM | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 05/10/2015 | 05/13/2015 | 04/25/2015 10:20:13 AM | | SKU | Order Date | ShipStart Date | ShipEnd Date |  |  | | 123 | 05/14/2015 10:00:00 AM | 05/25/2015 | 05/28/2015 |  |  | | SKU 123 Multiple Entries : Scenario 9 : Pick most latest, day ranges | | | | | | | SKU | Order Start Date | Order End Date | ShipStart Date | ShipEnd Date | ModifiedTime | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 7 | 10 | 04/30/2015 10:20:13 AM | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 3 | 5 | 04/29/2015 10:20:13 AM | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 1 | 3 | 04/25/2015 10:20:13 AM | | SKU | Order Date | ShipStart Date | ShipEnd Date |  |  | | 123 | 05/14/2015 10:00:00 AM | 05/25/2015 | 05/28/2015 |  |  | | SKU 123 Multiple Entries : Scenario 9 : Pick most latest, time stamp based | | | | | | | SKU | Order Start Date | Order End Date | ShipStart Date | ShipEnd Date | ModifiedTime | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 7 | 10 | 04/30/2015 10:20:13 AM | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 06/10/2015 | 06/13/2015 | 04/30/2015 5:20:13 AM | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 05/10/2015 | 05/13/2015 | 04/25/2015 10:20:13 AM | | SKU | Order Date | ShipStart Date | ShipEnd Date |  |  | | 123 | 05/14/2015 10:00:00 AM | 05/25/2015 | 05/28/2015 |  |  | | Default : Scenario 10 : Pick default for expired SKU specific values | | | | | | | SKU | Order Start Date | Order End Date | ShipStart Date | ShipEnd Date | ModifiedTime | | BASE | 04/01/2015 12:00:00 AM | 04/01/2023 12:00:00 AM | 3 | 5 | 04/25/2015 10:20:13 AM | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 7 | 10 | 04/30/2015 10:20:13 AM | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 06/10/2015 | 06/13/2015 | 04/30/2015 5:20:13 AM | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 05/10/2015 | 05/13/2015 | 04/25/2015 10:20:13 AM | | SKU | Order Date | ShipStart Date | ShipEnd Date |  |  | | 123 | 05/17/2015 10:00:00 AM | 05/20/2015 | 05/22/2015 |  |  | | 123 SKU : Scenario 11 : Exclude Weekends for day range translation | | | | | | | SKU | Order Start Date | Order End Date | ShipStart Date | ShipEnd Date | ModifiedTime | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 1 | 3 | 04/30/2015 10:20:13 AM | | SKU | Order Date | ShipStart Date | ShipEnd Date |  |  | | 123 | 05/09/2015 10:00:00 AM | 05/11/2015 | 05/13/2015 |  |  | | 123 SKU : Scenario 12 : Exclude Holidays and Weekends for day range translation (05/25 - Memorial Day) | | | | | | | SKU | Order Start Date | Order End Date | ShipStart Date | ShipEnd Date | ModifiedTime | | 123 | 05/01/2015 12:00:00 AM | 05/30/2015 12:00:00 AM | 3 | 5 | 04/30/2015 10:20:13 AM | | SKU | Order Date | ShipStart Date | ShipEnd Date |  |  | | 123 | 05/23/2015 10:00:00 AM | 05/28/2015 | 06/01/2015 |  |  | | | |
| 270938e\_OPUS\_OM\_HLD\_06 | HLD49 OPUS/OM shall dynamically display change in promise data at the SKU level on the DF- Confirm Order screen if any change has occurred for the date(s) during the DF order flow (based on modified\_date parameter of Atlas event)  [Wireframe link](http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUSMobile_270938e_DFConfirmation.pdf) | 270938e.SR.OPUS.06 |
| 270938e\_OPUS\_OM\_HLD\_07 | HLD50 New Store flag “**270938e\_Default\_Promise\_Dt\_Display** “ will control the display of Default channel specific dates in UI. Default value will be used when the Flag is ON.  When flag is ON, and if there are no SKU designated Promise dates applicable, default dates will be used for display.  When flag is OFF, and if there are no SKU designated Promise dates applicable, default dates will not be used in the display. Blanks will be displayed in the UI but yoda will receive default date range computed date values.  Yoda will always receive computed ship\_start\_date and ship\_end\_date irrespective of whether or not default flag is ON/OFF.  For Accessories, OPUS/OM will never display promise dates in the UI (when explicit dates are absent for SKU) nor pass to Yoda. If there are SKU specific designated values assigned to accessory sku, OPUS/OM will then display promise dates and pass them to yoda accordingly. Default store flags will not apply to Accessory types, it will only apply to non-accessories. | 270938e.SR.OPUS.07  270938e.SR.OPUS.08 |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | 270938e\_OPUSOM\_Default\_Promise\_Date\_Display : FLAG ON: Scenario 1 : Use BASE for display | | | | | | | SKU | Order Start Date | Order End Date | ShipStart Date | ShipEnd Date | ModifiedTime | | BASE | 04/01/2015 12:00:00 AM | 04/01/2023 12:00:00 AM | 3 | 5 | 04/25/2015 10:20:13 AM | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 7 | 10 | 04/30/2015 10:20:13 AM | | SKU | Order Date | Display : ShipStart Date | Display :ShipEnd Date | Yoda : Start | Yoda : End | | 123 | 05/17/2015 10:00:00 AM | 05/20/2015 | 05/22/2015 | 05/20/2015 | 05/22/2015 | | 270938e\_OPUSOM\_Default\_Promise\_Date\_Display : FLAG OFF: Scenario 2 : Do not use BASE, blanks will be displayed but computed values will be passed (123 SKU = Device) | | | | | | | SKU | Order Start Date | Order End Date | ShipStart Date | ShipEnd Date | ModifiedTime | | BASE | 04/01/2015 12:00:00 AM | 04/01/2023 12:00:00 AM | 3 | 5 | 04/25/2015 10:20:13 AM | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 7 | 10 | 04/30/2015 10:20:13 AM | | SKU | Order Date | Display : ShipStart Date | Display :ShipEnd Date | Yoda : Start | Yoda : End | | 123 | 05/17/2015 10:00:00 AM |  |  | 05/20/2015 | 05/22/2015 | | 270938e\_OPUSOM\_Default\_Promise\_Date\_Display : FLAG ON/OFF: Scenario 3 : When SKU specific values exist use it | | | | | | | SKU | Order Start Date | Order End Date | ShipStart Date | ShipEnd Date | ModifiedTime | | BASE | 04/01/2015 12:00:00 AM | 04/01/2023 12:00:00 AM | 3 | 5 | 04/25/2015 10:20:13 AM | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 06/01/2015 | 06/30/2015 | 04/30/2015 10:20:13 AM | | SKU | Order Date | Display : ShipStart Date | Display :ShipEnd Date | Yoda : Start | Yoda : End | | 123 | 05/10/2015 10:00:00 AM | 06/01/2015 | 06/30/2015 | 06/01/2015 | 06/30/2015 | | | |
| 270938e\_OPUS\_OM\_HLD\_08 | HLD51 If OT provided promise date and associated date/timestamp period is less than or equal to the order date or determined to be invalid, and when default store flag is OFF, OPUS/OM shall systematically display a blank value and pass a blank value to YODA | 270938e.SR.OPUS.09  270938e.SR.OPUS.09.1 |
| |  |  |  |  |  |  | | --- | --- | --- | --- | --- | --- | | 270938e\_OPUSOM\_Default\_Promise\_Date\_Display : FLAG OFF: Scenario 1 : Do not use BASE, , blanks will be displayed but computed values will be passed (123 SKU = Device) | | | | | | | SKU | Order Start Date | Order End Date | ShipStart Date | ShipEnd Date | ModifiedTime | | BASE | 04/01/2015 12:00:00 AM | 04/01/2023 12:00:00 AM | 3 | 5 | 04/25/2015 10:20:13 AM | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 7 | 10 | 04/30/2015 10:20:13 AM | | SKU | Order Date | Display : ShipStart Date | Display :ShipEnd Date | Yoda : Start | Yoda : End | | 123 | 05/17/2015 10:00:00 AM |  |  | 05/20/2015 | 05/22/2015 | | 270938e\_OPUSOM\_Default\_Promise\_Date\_Display : FLAG ON : Scenario 2 : Promise Dates expired | | | | | | | SKU | Order Start Date | Order End Date | ShipStart Date | ShipEnd Date | ModifiedTime | | BASE | 04/01/2015 12:00:00 AM | 04/01/2023 12:00:00 AM | 3 | 5 | 04/25/2015 10:20:13 AM | | 123 | 05/01/2015 12:00:00 AM | 05/15/2015 12:00:00 AM | 05/14/2015 | 05/14/2015 | 04/30/2015 10:20:13 AM | | SKU | Order Date | Display : ShipStart Date | Display :ShipEnd Date | Yoda : Start | Yoda : End | | 123 | 05/15/2015 10:00:00 AM | 05/19/2015 | 05/21/2015 | 05/19/2015 | 05/21/2015 | | | |
| 270938e\_OPUS\_OM\_HLD\_09 | HLD52 OPUS/OM sends promise date information to YODA regardless of default date display flag state (on or off) for devices.  OPUS/OM shall not display or pass any date values to YODA for accessories as there will be no default dates applied to accessories.  Logic to identify accessory vs device : CUST\_INV\_PLU\_RTL\_GENERIC.SERIAL\_TYPE = N and item\_type = K (identified as serialized items) while SERIAL\_TYPE = N and item\_type != K identified as non-serialized items. | 270938e.SR.OPUS.9.2 |
| **SCOPE ITEM 1- ORDER INFO** | | |
| 270938e\_OPUS\_OM\_HLD\_10 | HLD53 OPUS/OM shall continue to capture order date and timestamp (time/time zone) for completed DF orders (inclusive of offline orders). | 270938e.SR.OPUS.12 |
| 270938e\_OPUS\_OM\_HLD\_11 | HLD54 OPUS/OM shall pass the appropriate DF order details for new DF orders upon order completion to YODA via an enhanced API. Includes Ship\_start\_date and Ship\_end\_date and order\_created\_date in UTC formats.  Yoda API = **createDFOrder**  OPUS/OM shall display single date when both ship\_start\_date and ship\_end\_date are received same from OT/Atlas but will pass them both to Yoda API as same values. | 270938e.SR.OPUS.13 |
| 270938e\_OPUS\_OM\_HLD\_12 | HLD55 OPUS/OM shall invoke new CSI API (SendInflightEquipmentOrderStatus) for all new offline DF orders placed and offline order cancellations to update Order Track. Appropriate data is passed to CSI call as needed by CSI API.  CSI API : SendInflightEquipmentOrderStatus | 270938e.SR.OPUS.14  270938e.SR.OPUS.26 |
| 270938e\_OPUS\_OM\_HLD\_13 | HLD56 OPUS/OM shall convert and pass order date/timestamp information in GMT format to YODA for new DF orders (createDFOrder() api). | 270938e.SR.OPUS.15 |
| 270938e\_OPUS\_OM\_HLD\_14 | HLD57 OPUS/OM shall support a notification of single re-try attempt for all Offline-DF Order placements/cancellations, to OT via new CSI API to meet the OT requested 2 minute SLA.  Retry attempt will be based on the first CSI call Reponse (Failure) and re-attempt is immediate upon failure identification in CSI Response. | 270938e.SR.OPUS.16  270938e.SR.OPUS.27 |
| 270938e\_OPUS\_OM\_HLD\_15 | HLD58 OPUS/OM to refresh cache at regular intervals in support of the 1-2 minute SLA for promise date display. +/- 1-2 mins variance may occur (acceptable by Business).  1508 : FE Cache is implemented using weblogic timer and checked every 2 mins for updates from regional dbs.  1510 : Cache mechanism design is similar to frontend cache, however we would ping all FEs upon atlas notification request and rely on real time data cache updates rather than DB query fetch updates. | 270938e.SR.OPUS.17 |
| **SCOPE ITEM 2- DF CANCELS** | | |
| 270938e\_OPUS\_OM\_HLD\_16 | HLD59 OPUS/OM shall natively support omni-channel order level cancellations for DF orders that have reached YODA (inclusive of pre-orders) via YODA api - **cancelDFOrder**.  Only orders with following status will be cancellable from Yoda.   |  |  | | --- | --- | | **STATUS** | **STATUS DESCRIPTION** | | O | Open (not posted, not ready for processing, order still editable) | | P | Posted (order finalized and ready for processing, nothing shipped or canceled). Inclusive of RMA created status. | | T | Partially Shipped | | L | Partially Canceled |   OPUS/OM - DF Offline orders cancellation will follow BAU but 270938e\_OPUS\_OM\_HLD\_12 and 270938e\_OPUS\_OM\_HLD\_14 will apply as new change | 270938e.SR.OPUS.18  270938e.SR.OPUS.19 |
| 270938e\_OPUS\_OM\_HLD\_17 | HLD60 Current OPUS DF- Confirm Order screen messaging shall be updated to align with BAU OM verbiage.  Note- current OPUS verbiage is inaccurate and includes ‘cannot cancel’ verbiage. | 270938e.SR.OPUS.20 |
| 270938e\_OPUS\_OM\_HLD\_18 | HLD61 Enhance the cancel order button on the DF Order Detail screen to:  • dynamically display only for cancellable statuses (see above)  • display a new screen to invoke order level cancel with comments captured in new screen for completed omni-channel DF order. | 270938e.SR.OPUS.21 |
| 270938e\_OPUS\_OM\_HLD\_19 | HLD62 The DF Cancel screen displays upon clicking the Cancel button and includes:  • ‘are you sure’ messaging  • line item details of pending/unshipped items in cancellable status to be cancelled  • cancel reason input field/section (BAU but moved from the DF Order Details screen and remains required field)  • ability to navigate back to the previous screen user navigated from  • ability to cancel the displayed items (order level).   * Cancel Order Button is disabled in the default state until cancel reason is entered. * After cancellation screen moves to search page refreshed after confirmation (BAU). * Cancel errors reported by Yoda are displayed in UI | 270938e.SR.OPUS.22  270938e.SR.OPUS.23  270938e.SR.OPUS.24 |
| 270938e\_OPUS\_OM\_HLD\_20 | HLD63 OPUS/OM shall pass the relevant DF cancel details for omni-channel DF orders upon cancel transaction to YODA via the existing API (cancelDFOrder) | 270938e.SR.OPUS.25 |
| 270938e\_OPUS\_OM\_HLD\_21 | HLD64 OPUS/OM shall print/email a receipt for DF cancel orders regardless of the originating channel (omni-channel) i.e. if an online order is viewed in OPUS/OM, in a cancellable state, and cancelled, OPUS/OM cancels the order and generates a receipt.  Receipt updates may include the following, but are not limited to:  • External Order # (regardless of originating channel)  • Total amount from cancellation of all pending/unshipped items | 270938e.SR.OPUS.28 |
| **SCOPE ITEM 5- CONSENT** | | |
| 270938e\_OPUS\_OM\_HLD\_22 | HLD65 OPUS/OM shall support at order detail level via Yoda enhanced api call **InquireOrderDetail** - with indicator **consentInd**, to identify customer consent for omni-channel orders.  If **consentInd** = “SENT”, OPUS would display a message on Order Detail page with ‘**Provide Consent’** button. Any static messaging will be displayed per the WF for the consent context. | 270938e.SR.OPUS.30  270938e.SR.OPUS.32 |
| 270938e\_OPUS\_OM\_HLD\_23 | HLD66 Clicking the Consent CTA button displays a new Consent screen to:  • obtain and submit consent at the SKU level for items of an order  • Navigate back to the DF Order Detail screen in refreshed state  • Submit designated consent(s)  When both consent and update payment is needed, the screens are displayed in sequence. | 270938e.SR.OPUS.34 |
| 270938e\_OPUS\_OM\_HLD\_24 | HLD80 The Consent screen supports the ability to:  • display instructional static messaging per WF  • view pending item(s) at the SKU level needing consent  o Order by original ship commit date or original first date in range, then by new date in cases where original is the same.  • designate consent (SKU level- checked in the default state)  • enable consent(s) submission  • cancel the order (effectively denying consent for at least 1 SKU) by navigating back to the DF Order Details screen | 270938e.SR.OPUS.35  270938e.SR.OPUS.36 |
| 270938e\_OPUS\_OM\_HLD\_25 | HLD81 Customer can choose to give consent only to some of all pending consents without resulting in order cancellation. Select-all option checkbox will be available and selected by default but can be unchecked. If some of the line item consent checkbox was unselected warning message will be thrown when submitting selections. The ‘Submit’ button is disabled in the default state and dynamically enables when: at least 1 consent is selected. | 270938e.SR.OPUS.37  270938e.SR.OPUS.38  270938e.SR.OPUS.39  270938e.SR.OPUS.41  270938e.SR.OPUS.45 |
| 270938e\_OPUS\_OM\_HLD\_26 | HLD82 Confirmation screen /error screen thrown after consent selection submission based on response. | 270938e.SR.OPUS.40 |
| 270938e\_OPUS\_OM\_HLD\_27 | HLD83 Declining a consent is not explicitly available when providing consent. Rep will have to choose cancel order button to cancel the order when denying consent. Appropriate confirmation/error screens are shown upon cancellation. | 270938e.SR.OPUS.42  270938e.SR.OPUS.43 |
| 270938e\_OPUS\_OM\_HLD\_28 | HLD84 OPUS/OM shall propagate submitted SKU level consent information to OT via a new CSI API **UpdateEquipmentOrderConsent**. Only SKU level consents are to be sent by OPUS to OT (not blank/undesignated consents).  **CSI API** **UpdateEquipmentOrderConsent** Request   | **Element** | **Description** | | --- | --- | | PartnerId | Unique for each partner | | OrderId | Order track Id | | LineId | Order Track line Id | | ExpectedFromShipDate | Expected Ship From Date | | ExpectedToShipDate | Expected Ship To Date | | ConsentAcceptedInd | If Consent is accepted | | ConsentDate | Consent date | | UserId | AT&T user id or Customer CTN. | | 270938e.SR.OPUS.44 |
| 270938e\_OPUS\_OM\_HLD\_29 | HLD85 OPUS/OM shall support a single re-try attempt in passing consent information via CSI (to OT) to meet the OT requested 2 minute SLA. | 270938e.SR.OPUS.46 |
| **UPDATE PAYMENT METHOD** | | |
| 270938e\_OPUS\_OM\_HLD\_30 | HLD86 OPUS will display ‘Update payment” notice on Order Details screen based on the response from **YODA API ~~SearchOrderShipment~~ InquireOrderDetail**.  If **subStatusInd** = “**SPP**” (Suspend Pending Payment update), OPUS will display ‘Update payment” as shown in Wireframes. Yoda Orders with pending status O/P/T/L can only have have update payment method supported option. Button and any relevant static message will be hidden otherwise. | 270938e.SR.OPUS.52  270938e.SR.OPUS.53  270938e.SR.OPUS.56 |
| 270938e\_OPUS\_OM\_HLD\_31 | HLD87 Update Payment button displays the new Update Payment Method screen to:  • update tender type (credit card only) at the order level  • Navigate back to the DF Order Detail screen in refreshed state (no updates saved)  • Submit designated consent and/or updated payment method information  OPUS/OM shall leverage order information as sourced from YODA inquireOrderDetail (remainingAmountDue, remainingAmountDueWithTax), inclusive of amount due and taxes to display in the UI for payment method update.  Note- In the event a customer needs to provide both consent AND update payment method, content/fields for both sections display sequentially. First Update payment will be done that clones the order and consent is updated on the cloned order id.  Appropriate tender details are captured. Error messaging display when tender submission fails. | 270938e.SR.OPUS.57  270938e.SR.OPUS.58  270938e.SR.OPUS.58.1  270938e.SR.OPUS.54 |
| 270938e\_OPUS\_OM\_HLD\_32 | HLD88 OPUS will update the payment method using a new **CSI API UpdateDirectFullfillmentOrder** (which in turn calls YODA CloneOrder API)  **UpdateDirectFullfillmentOrder** request  cloneAction = “UPM”  externalOrderSource (PT for OPUS)  operatorIdentifier  referenceOrderNumber - original order id being cloned (Yoda Order Id)  TenderInput - new Credit Card info  OPUS should use correct external order source for the channel of the order (ex: OPUS initiated B2B ABS orders, ***YODA\_EXTERNAL\_ORDER\_SOURCE\_FOR\_ABS = PB***).  OPUS LD update payment method call need to have updates to Yoda. Currently opus uses External order source as LDC for all LD DF. | 270938e.SR.OPUS.55 |
| 270938e\_OPUS\_OM\_HLD\_33 | HLD89 OPUS/OM shall support the usage of network signature capture device via Use Station implementation for capturing tender update card swipe details. OM shall support additional devices like MSR / All-in-One / Audio Pin. BAU screens and logic will apply similar to existing Auto-Pay/Sales Screens for capturing details/error messaging/success messaging as appropriate for the display to be in sync with existing opus/om screens. | 270938e.SR.OPUS.59  270938e.SR.OPUS.60  270938e.SR.OPUS.60.1  270938e.SR.OPUS.61 |
| 270938e\_OPUS\_OM\_HLD\_34 | HLD90 OPUS/OM shall capture, store, and pass the following data to YODA for full amount authorization of the remaining pending items of the order. OPUS/OM will not explicitly call authorizePayment() but Yoda would call internally to fetch the auth fields (code, amount etc.). If Auth fields are non-null values from OPUS/OM, yoda will throw error. | 270938e.SR.OPUS.62 |
| 270938e\_OPUS\_OM\_HLD\_35 | HLD91 OPUS/OM shall support customer signature per BAU process for update payment method. T&C does not apply at this time. Current flow does not mandate capturing customer signature during credit card swipe process flow hence this requirement will not apply as BAU process | 270938e.SR.OPUS.64 |
| 270938e\_OPUS\_OM\_HLD\_36 | HLD92 OPUS/OM shall display success/error confirmation screens upon update payment. | 270938e.SR.OPUS.65  270938e.SR.OPUS.66 |
| 270938e\_OPUS\_OM\_HLD\_37 | HLD93 OPUS/OM shall generate a receipt for a successfully completed payment method update. Details in WF. | 270938e.SR.OPUS.67 |
| **CONSENT** | | |
| 270938e\_OPUS\_OM\_HLD\_38 | HLD95 OPUS/OM shall utilize Yoda **InquireOrderDetail** **~~SearchOrderShipment~~** api to identify using **subStatusInd** = **SPP** & Pending Status && Yoda **InquireOrderDetail** **consentInd** = **SENT** to display “Complete Action” button that triggers capturing update payment flow followed by consent capturing flow. First Update payment will be done that clones the order and consent is updated on the cloned order id rather than parent order id. SKIP button will be provided for skipping Step 1 payment update if needed. | 270938e.SR.OPUS.70  270938e.SR.OPUS.71 |
| 270938e\_OPUS\_OM\_HLD\_39 | HLD96 OPUS/OM shall utilize Yoda **InquireOrderDetail** api to identify using **consentInd** = SENT and to display “Provide Consent” button that triggers capturing consent. In combined sequential flow, SKIP button is provided along with static messaging as shown in WF , to facilitate moving on to the next flow step2(Consent) if the customer is not ready with step 1 (update payment). | 270938e.SR.OPUS.70  270938e.SR.OPUS.71.1 |
| **SCOPE ITEM 5- BLANKET CONSENT** | | |
| 270938e\_OPUS\_OM\_HLD\_40 | HLD97 OPUS/OM shall display, ‘Blank Consent’ section on Create Enterprise Agreement - Company Info screen.  OPUS/OM shall propagate blanket consent indicator to ROME via **CreateAgreementAndProfiles** API. No updates can be made to blanket consent via OPUS after fan creation. WF details will be implemented. When a FAN on an account has blanket consent, if promise dates change beyond 30 days which usually trigger action for providing consent from customer, when blanket consent is present on account having FAN, Order track will update Oracle and opus will read the consent indicator from yoda which simply reads the consentflag from oracle, in this case, no action should be shown to the customer. Customers whose accounts are created with FAN having blanket consent should not have to provide consent when adjusted ship dates > 30 days while blanket consent is provided for the FAN. FAST can be used for FAN lookup of blanket consent | 270938e.SR.OPUS.51  270938e.SR.OPUS.50  270938e.SR.OPUS.47 |
| **DF Order Search Enhacements** | | |
| 270938e\_OPUS\_OM\_HLD\_41 | HLD98 When Project flag is turned on, Existing DF Order Search UI will be enhanced to shuffle some fields around according the the WF. Default date range will show last 60 days (from/to) prepopulated. Location field is made optional and default value is populated blank (excluded in DF search by default). OPUS/OM will invoke Yoda api **SearchOrderShipment** api to support omni-channel DF Order retrieval. If the DF search query returns too many records (>10K records), yoda will return error translated into OPUS/OM as shown in WF. When searching in COR with Date, one of the yoda search criteria is mandatory. subStatus=SPP will mark the status in red indicating payment order is present on the order.  COR stores will have ability to perform OMNI-CHANNEL search of any df order originated from any other channel (ex: systemX, Phoenix, myAtt, OPSS, OrderHub, Permier etc.). COR will be able to search all LD orders as well as all omni-channel orders.  LD Stores will have very limited ability to perform Search. Only orders originated with in the operator allowed locations for LD can only look up those orders in OPUS LD.  For date range queries, OPUS/OM shall require at least one additional field as input. Static messaging shall display and if the additional field is not populated, OPUS/OM shall display an error message.  When Project flag is turned off, BAU screen will display and data will be fetched from OPUS database rather than yoda. | 270938e.SR.OPUS.72  270938e.SR.OPUS.73  270938e.SR.OPUS.73.1  270938e.SR.OPUS.76  270938e.SR.OPUS.77 |
| 270938e\_OPUS\_OM\_HLD\_42 | HLD99 For cloned DF orders, when using Yoda Order Id, OPUS/OM shall only display yoda resulted orders  **FLAG 270938e\_OrderMgmt\_OrderConsolidation OFF** : Consolidation OFF Flag Behavior (see 270938e\_OPUS\_OM\_HLD\_53, 270938e\_OPUS\_OM\_HLD\_54, 270938e\_OPUS\_OM\_HLD\_55): If parent yoda order is searched all children orders will be displayed, if last cloned yoda order is searched only that order will be displayed).  **FLAG 270938e\_OrderMgmt\_OrderConsolidation ON (default)** : If parent yoda order is searched all children orders will be fetched from Yoda **SearchOrderShipment** API call and consolidated into one order grouped by Receipt Number and displayed as One Order to the end user.).  The OPUS/OM DF Search results table shall be updated to support order level color-coded IVT indication | 270938e.SR.OPUS.74.1 |
| 270938e\_OPUS\_OM\_HLD\_43 | HLD100 Existing receipt number field will be enhanced to take omni-channel external order id :  • COR transaction ID (BAU)  • online order confirmation #  • call center order ID  Valid field entries include but are not limited to:  • blank (YODA orders)  • alphanumeric | 270938e.SR.OPUS.75 |
| 270938e\_OPUS\_OM\_HLD\_44 | HLD101 The Existing OPUS filter functionality shall remain BAU for COR orders only as sourced natively:  • Serialized Orders  • Ship to Store Orders  • Offline Orders | 270938e.SR.OPUS.76.1 |
| **DF Order Detail Page - Enhacements** | | |
| 270938e\_OPUS\_OM\_HLD\_45 | HLD102 OPUS/OM shall enhance the DF Order detail page to display additional fields like Home Phone number, ~~Zip Code~~, ~~Originating Order source code for Originating Channel (~~**~~CR pending~~**~~)~~ at order level. Actual and Adjusted ship dates will be displayed in the order detail page along with info icon that brings up product overlay as BAU.  ~~Suggestion : Ideal scenario is to have Yoda send externalorderdesc along with externalordersource. Alternate is to have opus connect to yoda using existing DB connection (mechid) to yoda pricing table to pull externalordersource vs externalorderdesc mapping but this is a cross-cutting concern and opus does see a risk in maintenance nightmare and implementation concern at this time. It will also have a performance hit~~. | 270938e.SR.OPUS.78 |
| 270938e\_OPUS\_OM\_HLD\_46 | HLD103 SKU level detail presentation on the DF Order Detail screen shall be grouped by:  • YODA order level status (OPUS/OM to support translation as necessary to:  o Shipped  o Pending  o Cancelled  Then by ship commit date for pending items of an order  o Order by original ship commit date or original first date in range, then by new date(s) in cases where original is the same. | 270938e.SR.OPUS.79 |
| **NFR** | | |
| 270938e\_OPUS\_OM\_HLD\_47 | HLD165 HLD104 OPUS/OM shall ensure 270938e verbiage/messaging is supported in a configurable nature for adhoc updatable items. | 270938e.SR.OPUS.81 |
| **Local Dealers** | | |
| 270938e\_OPUS\_OM\_HLD\_48 | HLD166 All functionality newly defined for 270938e shall be supported in LD stores according to the user location access permissions (location drop down (BAU) rules apply). Project level flag and Default Date display flag will be same as COR for all LD OPUS/OM new functionality. | 270938e.SR.OPUS.82 |
| 270938e\_OPUS\_OM\_HLD\_49 | HLD167 LD rep cannot access omni-channel orders search due to location constraint, orders initiated only in the user LD location can be searched. BAU logic fetches the locations based on logic defined in InventoryControlManager.getLocationsForUserFromSWIM() 🡺 InsysICServicesDAO.getLocationsForUser() 🡺 p\_retail\_location.get\_user\_location. Location drop down is prepopulated with non-blank values and pre-selected as applicable to the user location access and is mandatory in all search criteria. Promise Date, Search DF Order, DF Order details, blanket consent, Consent, Cancel DF Order, Update Payment functionality will apply only for user location specific orders. | 270938e.SR.OPUS.83 |
| 270938e\_OPUS\_OM\_HLD\_50 | HLD168 If Order cannot be found/displayed error messaging shall display per WF - No orders found based upon search criteria. Same messaging will apply for COR search as well. | 270938e.SR.OPUS.84 |
| 270938e\_OPUS\_OM\_HLD\_51 | HLD169 OPUS shall support LD update payment method functionality as defined for OPUS COR using USB ISC250 devices to capture credit card details for card authorization. OM supports devices similar to COR stores (Bluetooth, all-in-one, etc.) | 270938e.SR.OPUS.85 |
| 270938e\_OPUS\_OM\_HLD\_52 | HLD170 Since Yoda does not directly support search order shipments by retail location id, current design to search for DF orders in LD store will apply logic to :   1. Search the OPUS database with all search criteria inclusive of the selected location drop down using (p\_direct\_fulfillment.get\_df\_summary BAU) and include order id’s into memory (proceed to step 2 without displaying on UI) 2. Search Yoda with all search criteria (excluding the location id as unsupported field) and fetch results 3. Merge the steps 1 and 2 based on the yoda order id to get the status of the order 4. Display orders in search results in UI   OPUS/OM notes performance risk (although minor) in two way round-trip in this approach until yoda directly supports search orders by retail store location id. Yoda search will use the external order source criteria set by opus in request LDC/LDD etc. to restrict the search limit.  OPUS/OM COR search order UI will have location drop down field disabled by default. Location drop down will be enabled when COR only filters (requiring COR only opus db search without going to yoda) are selected for local opus db search. | 270938e.SR.OPUS.82  270938e.SR.OPUS.83  270938e.SR.OPUS.84  270938e.SR.OPUS.86 |
|  | **CR126718/CR129916- ORDER CONSOLIDATION** |  |
| 270938e\_OPUS\_OM\_HLD\_53 | **DF Search screen- COR and LD**  When order is searched in search screen by **search criteria like yoda order id**, the order id displayed in the search results section will show the order id of itself (if it has never been involved in cloning process) or display the ancestor parent of the order id as applicable according to this table.   |  |  | | --- | --- | | Results Set Contains: | UI displays: | | DF Standalone order | DF Standalone order | | DF BRE order | DF BRE order | | DF Originating Parent order | DF Originating Parent order | | DF non-BRE Child order only | DF Originating Parent order | | DF BRE Child order only | Search = Yoda order id => Immediate REF ID (not the parent),  Search = Other where it results in all children => Originating BRE Order (not parent id) | | DF Originating Parent and child | DF Originating Parent order |   OPUS/OM displays the following for cloned orders with parent order id:   * Parent YODA Order ID * Generic instructional verbiage as ‘Click Details’ or last child status (depending on search criteria) to navigate to the Order Details screen   For orders not involved with cloning process, it would show the order status accordingly.  When orders are **searched by firstname or lastname or externalorderid**, yoda will return all child clone orders, we will group orders by external order id and isolate a single parent order (parent order id) based on the clone action = UPM, we will display single order with parent id and status = ~~click details~~ last child status (and discard the others in the group for display)  If search has resulted in all child clone orders fetched, then we will pick the most recent child order substatus indicator and update status (either “Click Details” or last child status) with red color as update change payment indicator comes in. ~~If results does not get most recent child (yoda order id search), we will always show the “Click Details” in black color and clicking on the details button then would show if payment update is needed (as we try to make performance optimization).~~ Currently yoda returns all yoda child orders for any search criteria (even if you search by yoda order id, all children under that order will be returned by yoda), so opus can always pick last child status during consolidation for consolidated orders or show “Click details” if it does not apply.  If search by yoda order id results in parent and ref\_id both none, we will show status = (Explicit status) and appropriate color based on change payment indicator (parent found)    ~~We will assume that we will always get most recent child when searching by other search criteria. We will display Red font message “Click Details” Status for search results where child order is present.~~ | 270938e.SR.OPUS.87 |
| 270938e\_OPUS\_OM\_HLD\_54 | OPUS/OM shall interface with YODA (SearchOrder API) and support native logic to determine parent/child relationships | 270938e.SR.OPUS.88 |
| 270938e\_OPUS\_OM\_HLD\_55 | **Order Consolidation On/Off Flag**  OPUS/OM shall support a single on/off flag at the **store configurable** **270938e\_OrderMgmt\_OrderConsolidation** to enable/disable 270938e order consolidation functionality as necessary (on in the default state). Default value is Y (ON). | 270938e.SR.OPUS.89 |
| 270938e\_OPUS\_OM\_HLD\_56 | **Order Details- COR and LD**  OPUS/OM shall leverage the enhanced InquireOrderDetail YODA API to identify parent and child order line level details (multiple calls as needed).  If original order search was done by **yoda order id**, Use External Order ID to searchOrder API (should give us all child order ids) then we will call InquireOrderDetail for each yoda order ids.  If original order search was done using other criteria to result all children in the order, then we will not recall the searchorderapi once again when clicking the detail button. | 270938e.SR.OPUS.90 |
| 270938e\_OPUS\_OM\_HLD\_57 | **Order Details- COR and LD**  A single most recent status displays for an item across all parent and child/IVT orders categorized by status regardless of search criteria.        *Please reference wireframes for UI presentation details* | 270938e.SR.OPUS.91 |
| 270938e\_OPUS\_OM\_HLD\_58 | **Order Details- COR and LD**  For DF BREs, the original item returned shall be represented with a ‘R’ value as Qty instead of a decremented count value.  OPUS will display all items returned by Yoda.  *Please reference wireframes for UI presentation details.* | 270938e.SR.OPUS.91.1 |
| 270938e\_OPUS\_OM\_HLD\_59 | **Order Details – COR and LD**  OPUS/OM shall display the parent Order ID (Confirmation Number) in the all screens => order detail section,cancel, consent.  When performing consent operation or update payment operation, pass the most recent child yoda order id to CSI call.  When performing cancel order operation pass the most recent child yoda order id to Yoda api call cancelDFOrder.  *Please reference wireframes for UI presentation details.* | 270938e.SR.OPUS.92 |
| 270938e\_OPUS\_OM\_HLD\_60 | **Order Details- COR and LD**  In the ‘Transaction Information’ section for all orders:   * Parent order pricing details display * Parent receipt date displays * Most recent Order Level status as consumed from YODA displays   *Please reference wireframes for UI presentation details.* | 270938e.SR.OPUS.93 |
|  | **TBD New CR : Cancel Order, Update Payment, Consent, Integrated Update Payment and Consent flow, Receipts for Consolidated Orders.**  **Cancel Order** -- Will pass child yoda order id to cancel the cloned last child. As only last order can be in Cancellable state. Cancel Order button shall be displayed based on most recent child cancellable state.  **Update Payment** -- will only happen on the last cloned latest child yoda order id? – No According to Mark. Need to handle scenarios for multiple update payments pending in the same parent->child hierarchy (may be out of scope currently). Current design can support update payment on most recent child order.  **Consent** -- will only happen on a line SKU for last cloned latest child yoda order id? According to Oracle, consent can be updated on any given yoda order id. Since opus only supports cloned order display as single order, we may not be able to support parent->child (consent update on level higher than most recent child) with current design.  **Update Payment + Consent** – Integrated flow. Supported only if most recent child is in this state.  Receipt for Consolidated order operations :  Cancel Order, Update Payment operations.  Re-print Receipt button –  Only works for opus orders. What if opus order is cloned. |  |

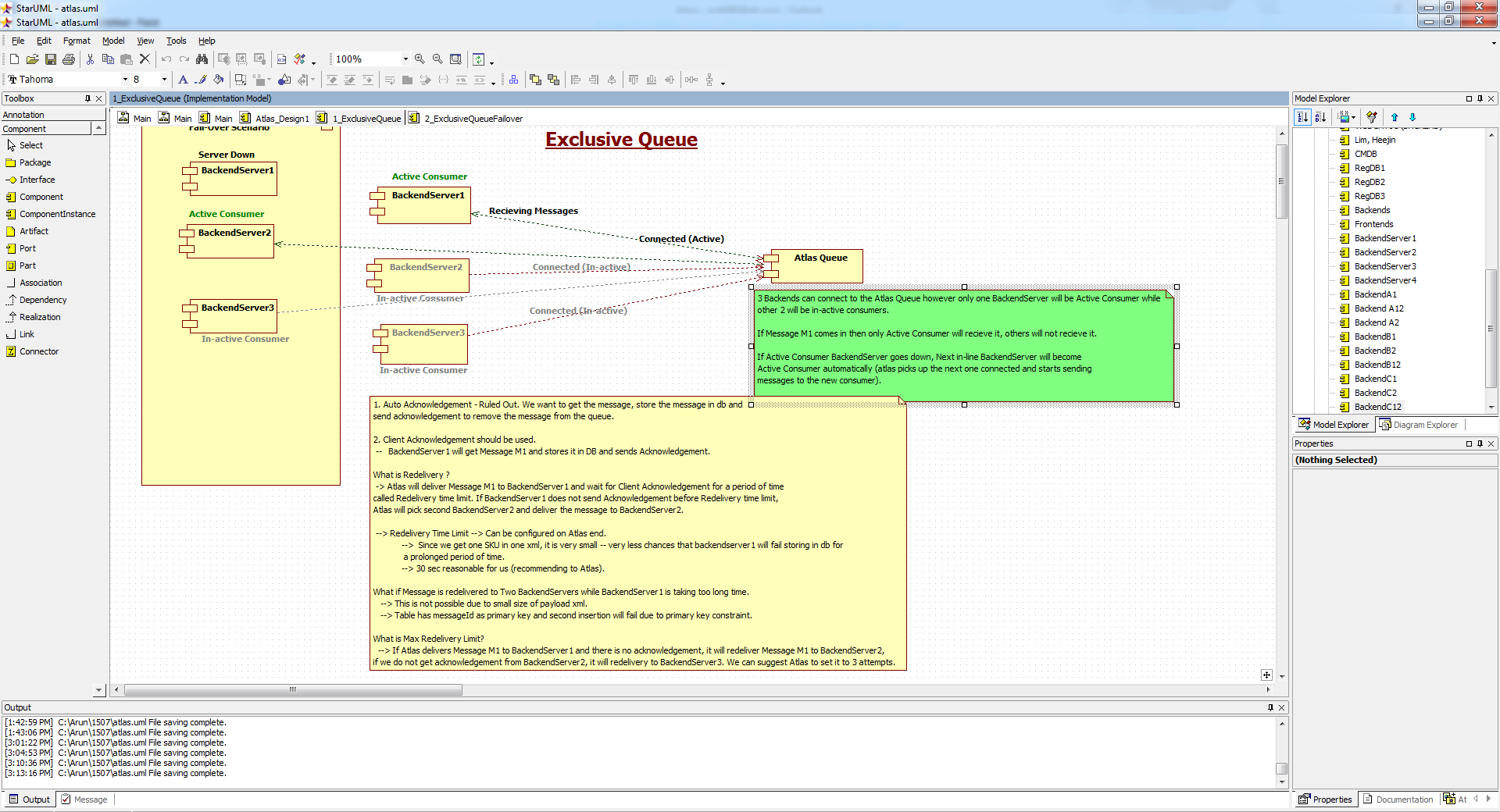
The [SRT\_RM\_ReqPro\_Auto\_Tagging](https://cps.web.att.com/CPSWorkplace/getContent?id=current&vsId=%7B67070CE6-CDED-4932-8F1D-F6073BA40DF8%7D&objectStoreName=IT-Architecture.__.Planning.__.and.__.Integration&objectType=document&guestId=servicesguest) Reference Document provides information on how to use auto-tagging within SRT ReqPro.

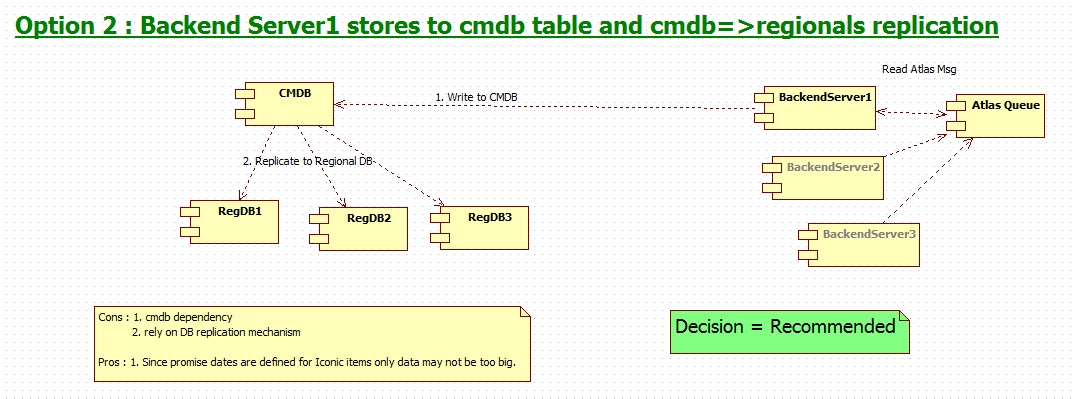
If the table is not used to capture requirements, please remove it from document.

## Functional Overview

|  |  |  |
| --- | --- | --- |
| **Module Name** | **New/Existing** | **Functional description** |
|  |  |  |

## Middleware Design

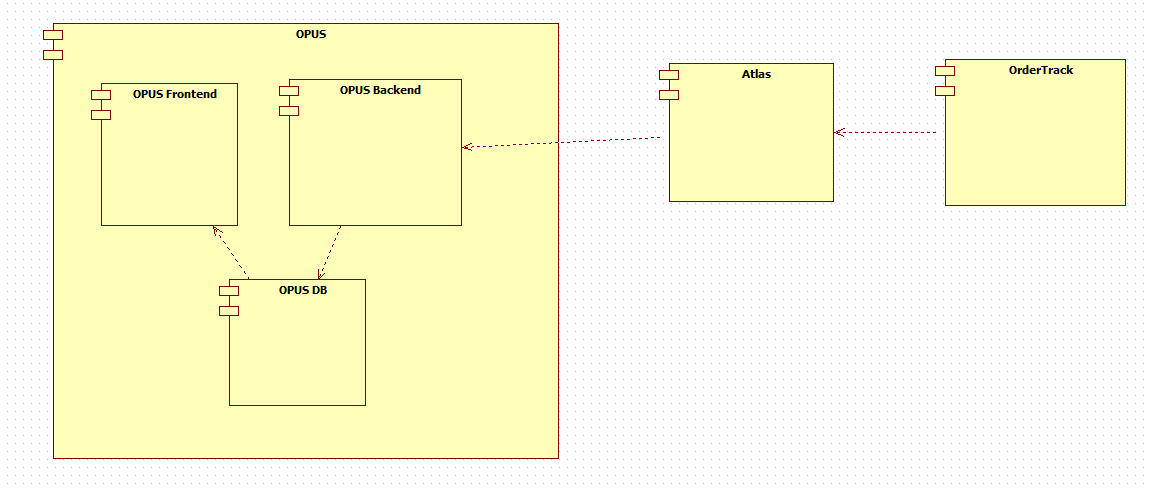


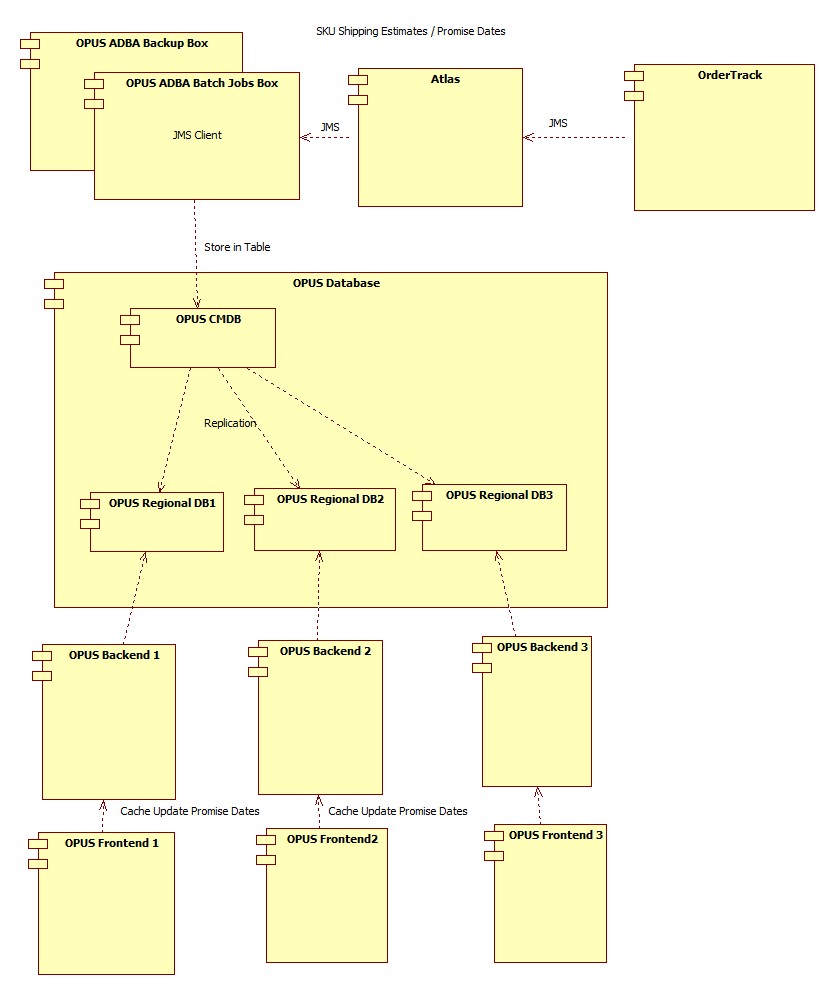


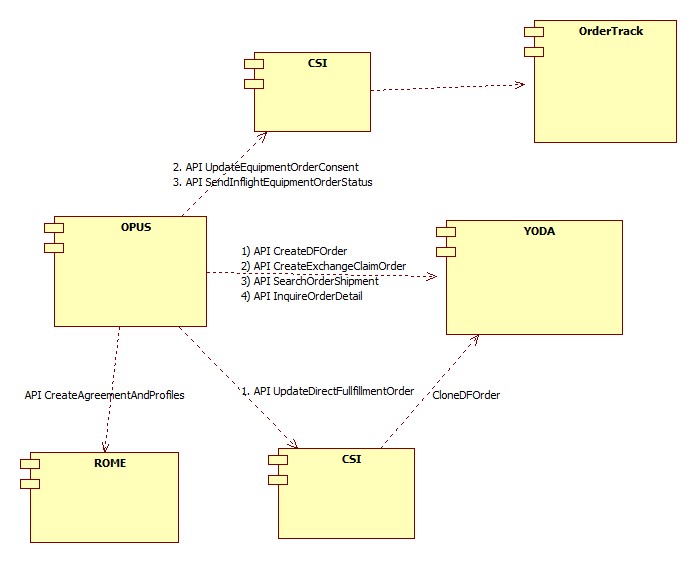
**External Interfaces**

### Component Diagram :

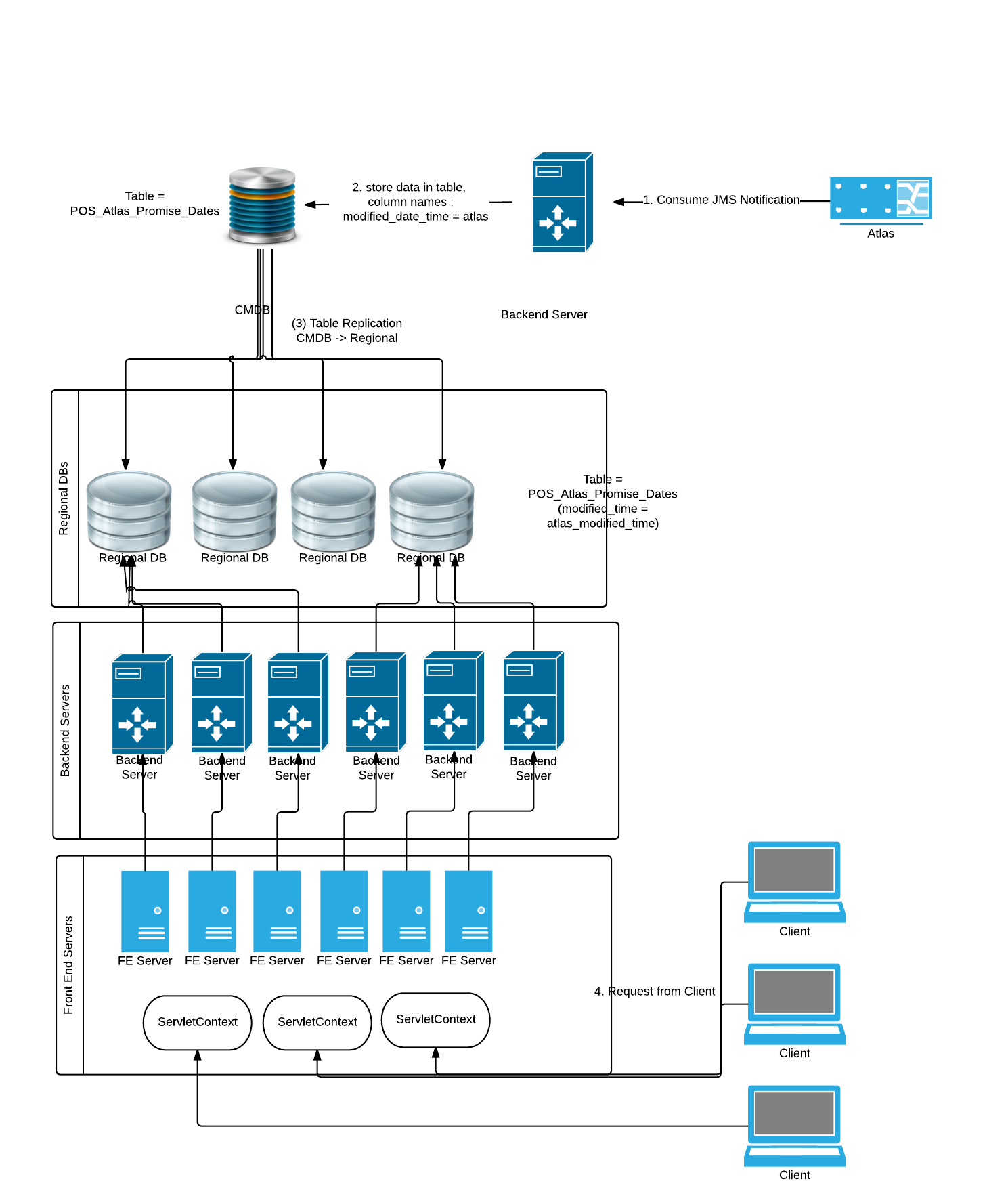
High Level - Component Diagram for Promise Date Handling:

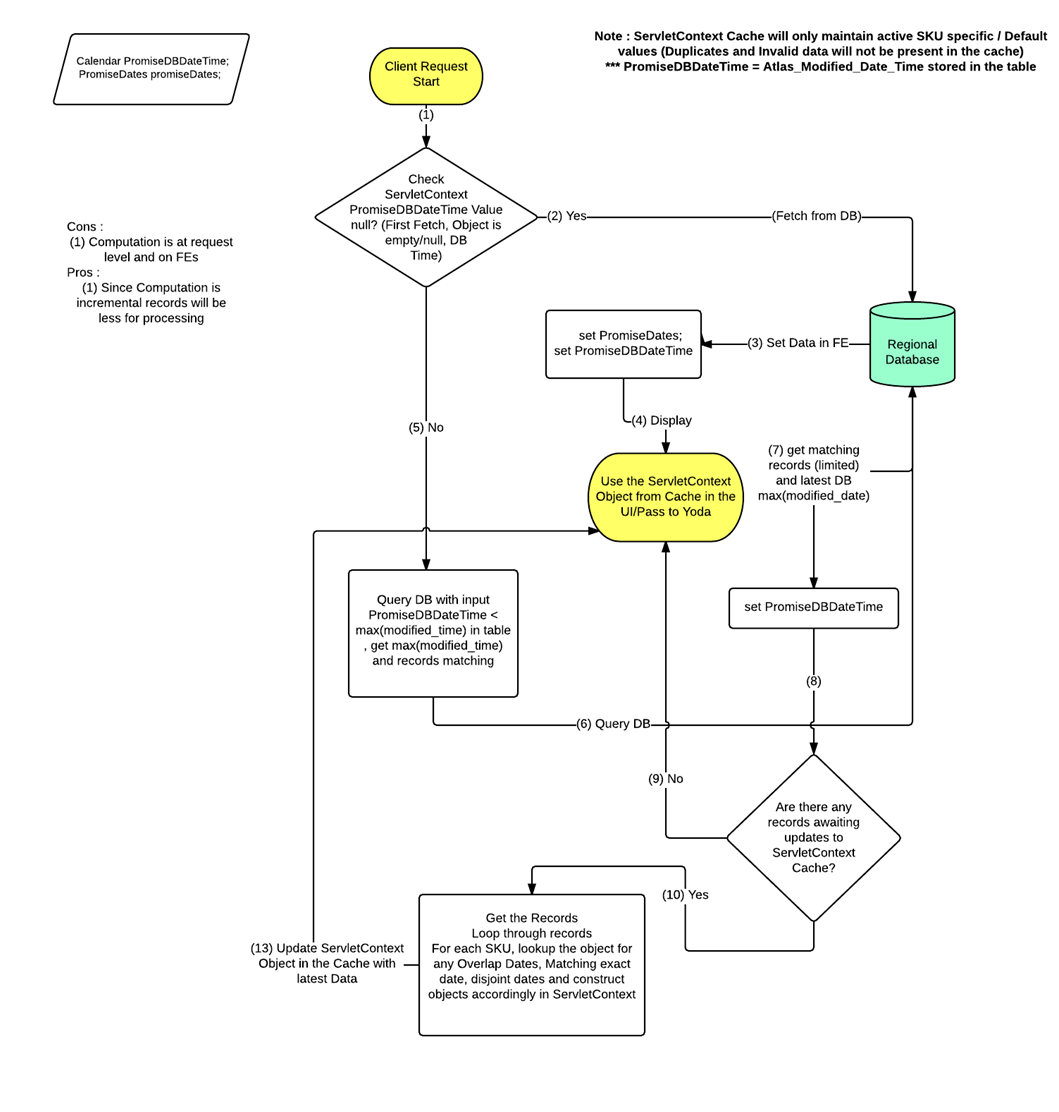






High level - Component flow for promise dates:

**



**Performance Design**

NA

### 

### Security Design

*Create an inventory of all SPI/PCI data elements that will require special handling with encryption and watermarking at each architecture layer.*

* **NA**

|  |  |  |
| --- | --- | --- |
| **SPI/PCI Data Element Inventory** | | |
| **Screen Elements** | | |
| **#** | **Element Name** | **Handling** |
| *1* | *NA* | *NA* |
|  |  |  |
| **External Interface Elements (Service/EJB calls)** | | |
| **#** | **Element Name** | **Handling** |
| 1 | *NA* | *NA* |
|  |  |  |
| **Database Elements** | | |
| **#** | **Element Name** | **Handling** |
| *1* | *NA* | *NA* |
|  |  |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Watermarking Inventory** | | | |
| **#** | **SPI/PCI Entity/Attribute Name** | **Logging Injection Point** | **Watermarking utility** |
| 1 | NA | NA | NA |

### UI Design

### UI Design - Wireframe

<http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUSMobile_270938e_ServiceandDevice.pdf>

<http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUSMobile_270938e_CartDF.pdf>

<http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUSMobile_270938e_DFConfirmation.pdf>

<http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUSMobile_270938e_Direct_Fulfillment_Standalone_iPhones.pdf>

<http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUSMobile_270938e_Direct_Fulfillment_Standalone_Accessories.pdf>

<http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUSMobile_270938e_Direct_Fulfillment_Standalone.pdf>

<http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUSMobile_270938e_DFSearchResults.pdf>

<http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUSMobile_270938e_OrderDetails_PartiallyShipped.pdf>

<http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUSMobile_270938e_CancelOrder.pdf>

<http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUSMobile_270938e_ConsentOnly.pdf>

<http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUSMobile_270938e_Consent_with_PaymentUpdate.pdf>

<http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUSMobile_270938e_BlanketConsent.pdf>

<http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUS_270938e_DF_Confirmation.pdf>

<http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUS_270938e_UpdatePayment_Receipt.pdf>

<http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUS_270938e_CancelOrder_Receipt.pdf>

<http://alhoov1uvinda01:8600/OPUS/OPUS_Launcher_DEV/pdfs/OPUS/OPUSMobile_270938e_UpdatePaymentOnly.pdf>

### UI Design - Screen Element Definition and Data Mapping

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Screen :** | | | | | | | | | |
| **Change Summary** | | | |  | | | | | |
| **Element Label** | **Required?** | **Element Type** | **Table column?** | **SPI/PCI** | **Default Value** | **Format** | **Data Source/Target** | **Tab Order** | **Init. Focus** |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Screen :** | | | | | | | | | |
| **Change Summary** | | | |  | | | | | |
| **Element Label** | **Required?** | **Element Type** | **Table column?** | **SPI/PCI** | **Default Value** | **Format** | **Data Source/Target** | **Tab Order** | **Init. Focus** |
|  |  |  |  |  |  |  |  |  |  |

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Screen :** | | | | | | | | | |
| **Change Summary** | | | |  | | | | | |
| **Element Label** | **Required?** | **Element Type** | **Table column?** | **SPI/PCI** | **Default Value** | **Format** | **Data Source/Target** | **Tab Order** | **Init. Focus** |
|  |  |  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |  |  |

### UI Action Processing - Sequence Diagram

### Asynchronous/Batch Processing - Sequence Diagram

*NA*

**Externalized Configuration**

*All application configuration parameters (table filled out with examples).*

|  |  |  |
| --- | --- | --- |
| **Application Configuration Parameters** | | |
| **Parameter Type** | **Parameter name** | **Description** |
| Store Level configuration | 270938e \_Sacred7\_Iconic\_DF | This store configurations provide ability to turn on/off project level changes for DF Sacred 7 Iconic project. **Default value = N** |
| Store Level configuration | 270938e\_Default\_Promise\_Dt\_Display | This store configurations provide ability to turn on/off usage of the channel specific BASE SKU values when designated SKU specific values are absent. This is only applicable for display. **Default value = Y** |
| Store Level configuration | 270938e\_OrderMgmt\_OrderConsolidation | Consolidated orders display when searching for cloned DF orders. **Default value = Y** |
| Store Level configuration | 270938e\_FORWARD\_ORDER\_ENHANCEMENTS | Internal to OPUS Dev only. Business did not ask for this flag. If we need to turn off the forward ordering flow changes for 270938e while keeping all of the post ordering flows intact turned on, this flag supports it. **Default value = Y** |
| Enterprise Level Configuration | ORDER\_MGMT\_BLANKET\_CONSENT\_MSG  ORDER\_MGMT\_UPDATE\_PAYMENT\_MSG  ORDER\_MGMT\_SHIPPING\_CONSENT\_MSG | Each enterprise configurable supports the messaging fetched from db and displayed in the UI. |
| Enterprise Level Configuration | ORDER\_MGMT\_DF\_SEARCH\_DEFAULT\_DAY\_RANGE | Supports the default value of 60 days in DF Order search UI. Default value = 60 |
| Enterprise Level Configuration | ORDER\_MGMT\_SHIP\_CMMIT\_DISPLAY\_LIST  ORDER\_MGMT\_BASE\_SHIP\_CMMIT\_DISPLAY\_LIST | Supports the list of device types and applicable device types to whom promise dates can be applied along with base channel level promise dates as applicable. |
| Enterprise Level Configuration | ORDER\_MGMT\_OT\_CHANNEL\_CODE | Default Channel Code to be passed to Order track during sendInFlightOrder and other CSI calls as applicable. Default Value = OPUSDF. |

## Database Tier Design

*NA*

### Logical Data Model

|  |  |  |
| --- | --- | --- |
| **Change Summary** | | |
| **Entity Name** | **Change type** | **Change Description** |
| NA | NA | NA |
|  |  |  |
|  |  |  |

*Sample diagram shown below.*

NA

### Database Interfaces

* *NA*

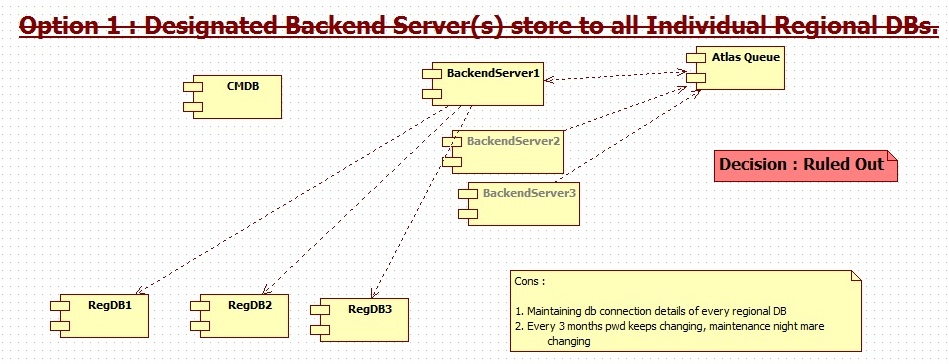
### Batch Interfaces

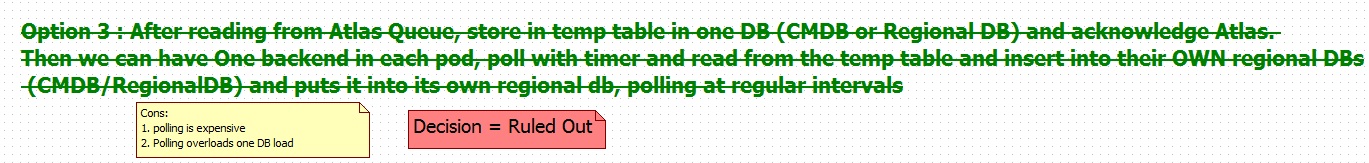
*NA*

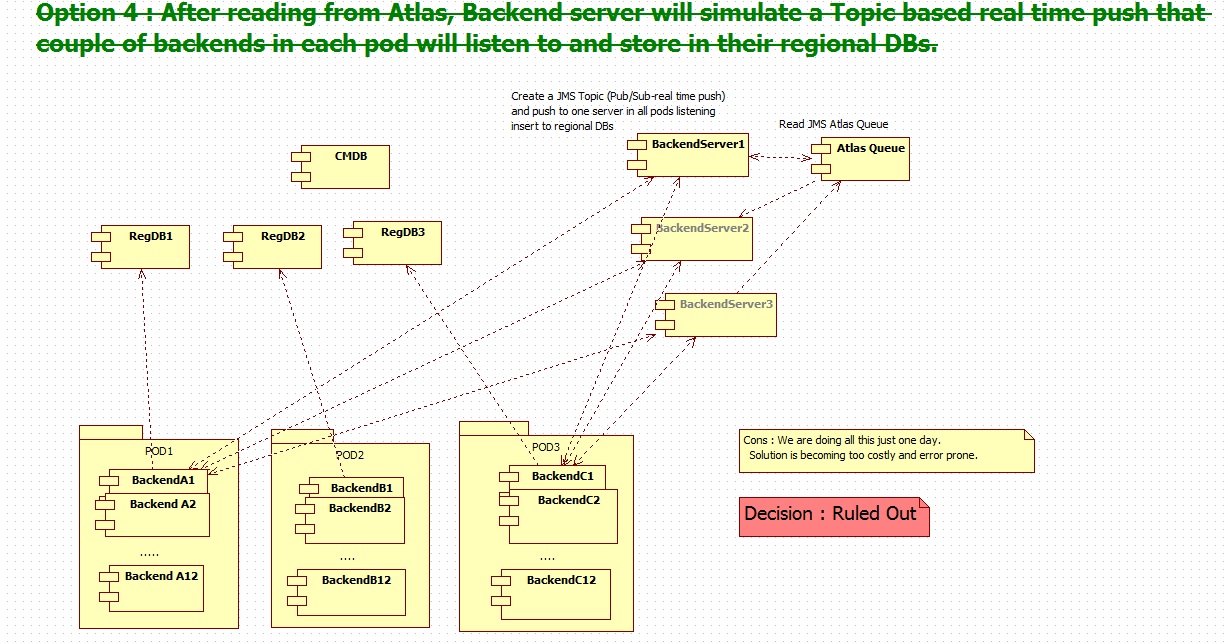
### Replication

*NA*

## Alternative Designs







## Assumptions/Risks

* Project level store flags to control the functionality implemented for this project to mitigate the risk.

## Pre-Production Disaster Recovery Planning

### NA

## Other Plans and References

NA

|  |
| --- |
|  |

## 

## Acceptance & Approvals

Overview

Use this section to capture approvals in the event that electronic approvals via the PRISM Project Workflow Module will not be used.

The Approvers of this work product agree that this document is acceptable and complete to the best of their knowledge and will be used by the project team as an official deliverable for the project. It is further agreed that this document can now be baselined and any changes to these sections from this point forward must follow the Managing Change in the Technology Development Unified Process.

Embed evidence of approval in the review table below, or use the PRISM Approval Functionality in the Project Workflow Module Workflow Template View.

Approvers

|  |  |  |  |
| --- | --- | --- | --- |
| **ATTUID and Name** | **Role** | **Group/Application** | **Version Approved, Approval Date and Approval Evidence** |
| Kandi, Pavan | PTA | OPUS / OM |  |
| Duvvada, Siva | Retail Functional Architecture | OPUS / OM |  |
| Gunna, Moapal | STD | OPUS / OM |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |