

**Version 1.1**

**Date: 2018/04/09**

Abstract

**The objective of this document is to describe the purpose and functionality of the dispatch activity – post picking and the alternate processing of goods to be delivered such as supplier / customer returns etc.**

*Despatching Processing*

*Describe the process post picking completion*

**Table of Contents**

Document approval and distribution list 2

1. Introduction 3

2. Audience 3

3. Objectives 4

4. Dependencies 4

5. Risks and mitigation 4

6. Requirements overview 4

7. Acceptance 6

# Document approval and distribution list

|  |  |  |  |
| --- | --- | --- | --- |
| **Description** | **Name / Title** | **Signature** | **Date** |
| **Document Type / purpose** | | | |
| Prepared by | Graham Smith |  | 2018/10/08 |
| Reviewed by |  |  |  |
| Approved by |  |  |  |

# Introduction

Part of the despatch of goods management

Goods are primarily despatched to customers based on advised requirements using the sales order process and after the picking process has been confirmed as accurate and complete

However, there are instances where goods are delivered outside of the standard sales order process; some of which are:

* Returns to Vendor (RTV)
* Returns to Customers (RFC)
* Zero invoices for collections
* ….

# Audience

Despatch

Picking

Sales order process

Administration

Creditors

Technical - developers

Software developers

# Functionality Narrative general customer orders

* 1. Picking, once committed to ***Picking completed not invoiced*** status, the goods and supporting documentation is handed over to the checking and packing team
  2. Goods are ticked off against supporting documentation
     1. By design, packing slips are printed post picking completed status
     2. It was a recent management decision to change from a packing slip to printing the official ***invoice*** ahead of ***checking completed*** state on the assumption that the non-conformances, being few, would be corrected using application transactions (credit note).

The decision was based on ***cost saving*** and opinionated the saving of no more than R4,000 per month does not do justice to the notion and should be reconsidered

* 1. If all is OK, then the ***checker / packer*** prints an official:
     1. Invoice should the customer be designated so
     2. Alternatively, a delivery not is produced. This is a copy of the invoice without any monetary values
     3. The main reason for printing a delivery note is that Engineparts delivers goods to customers of our customers and will invoice their customer separately.
  2. If not OK, the goods and documentation is returned to the picking supervisor for correction:
     1. Fix the non-conformation items picked
     2. Update the non-conformant picking job sales orders where goods are short.
     3. Return to store where items have been over-supplied
     4. Where the invoice has been printed in advance, the requisite crediting transactions needs to be completed (this is considered a risk as it is relatively uncontrolled process and can lead to accidental or otherwise misappropriation)
  3. The packer packs the checked goods into one or more parcels and for each parcel requests a tracking label from the system
  4. The printed invoice / delivery is placed into the last parcel.
  5. Each consignment is generally passed to the appropriate load bay for vehicle loading and related controls
  6. The related management information around picker to despatch performance is documented in the picking sub-system

# Functionality Narrative counter customer orders

* 1. Sales at a designated sales counter are defaulted to the sales counter unless the customer is designated as a local delivery on account sale; in which case goods are routed to the local delivery service station.
  2. For purposes of this narrative, only the counter sales process is documented
  3. On release of the sales order it enters the optimised picking queue and prioritised by the operational floor manager to meet with overall business expectations.
  4. The overall counter expectation is to service customers within the shortest possible waiting time. To ensure visibility of each sales order process to customer departure time, a series of time stamp footprints are recorded during the fulfilment life cycle
     1. Sales order start time
     2. Sales order confirm time (this is also when the customer tracking ticket is printed)
     3. Sales order to picking job time
     4. Picking job complete time (wait times are available too). The time stamp is confirmed using a scanner at the drop of hatch used by the picker to scan the packing slip barcode
     5. The security staff then scan the same packing slip to confirm checked status.
     6. As soon as the security staff scan the packing slip the transaction is displayed on all the cashier terminals as ***ready to receive payment***. The full documentation around cashiering tasks is taken up in the related document
     7. For those cashiers that can receive payments then select an item top of the list using a mouse click. This initiates the server application to:
        1. Display customer order details on the display panel
        2. Using the Chrome browser capability for voice simulation, text details is announced via PC connected speakers. The voice style and its contents can be changed dynamically using text and related parameters without requiring detail technical assistance other than standard web developer skills.
     8. Should the customer announce at the indicated booth, the cashier receives payment and the packing slip status changes to payment effected ready to invoice.
     9. At the collection counter, the customer ticket directs the security staff to select and scan the correct packing slip and should the paid status be true the related invoice is printed.
     10. Goods are handed to the customer and the transaction is marked complete
  5. To measure the system and resources profile to ensure optimal functioning against the best possible service delivery, ePart has a set of management reports that provide sales order to completion stats pinpointing times of delay.
  6. The picking prioritisation process is largely manual yet configurable to limit the number of sales orders per picking job.
  7. Generally, for these, only one order per picker is assigned to provide best possible turnaround times
     1. However, when multiple sales order per picking job is assigned, it is known to cause delays at the counter where one-liner sales orders are mixed with order comprising of many items. ***In this kind of presentation, the one-liner is undesirably delayed and is a future optimisation requirement***.

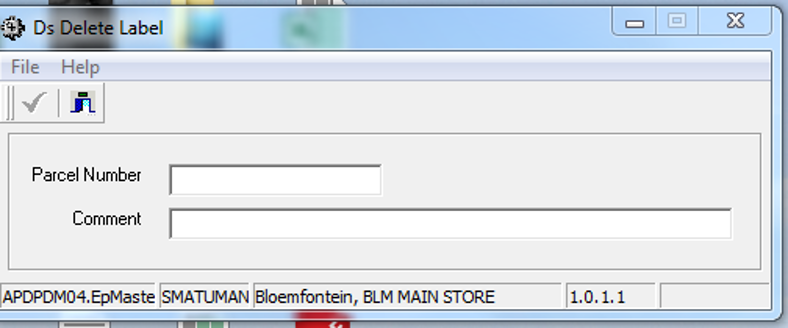
# Printing of carton / parcel labels

On packing before sealing the carton, the following is done:

* A label is printed and affixed to the specific parcel
* If it is the last parcel, the original invoice or certified copy is placed into said parcel

The parcel is sealed

If a label was incorrectly printed it can be deleted – only at authorised level using the following function:

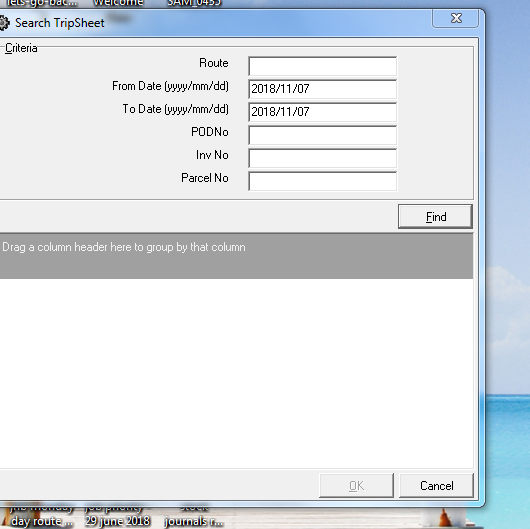


# Trip sheets

Trip sheets reflect the delivery requirements along a specific route code. The nodes for delivery are sequenced in the driver route sequence.

Each node has the delivery requirements per customer. Several attempts were made to find a suitable means to arrange the customers in sequence, but this has not been a successful exercise as drivers will optimise their sequences based on specific customers in the list per node.

The trip-sheet search facility is depicted as follows and is dominantly used to re-print at a later date and can be initiated using various search keys; often initiated using the Invoice reference



# Document samples

# Future considerations

Under current deliberations, there is thought around using barcode labels to verify the accuracy of picked goods meeting with customer expectations.

Currently this is done manually by comparing the item on the invoice / packing slip with the goods picked. Humanly speaking, this often fails as the brain often ***reads*** correctness in favour of ***likeness*** such as in the case of sized items like piston rings where the group code changes, but the serial number remains.

There is no doubt that greater efficiencies is a must consideration due to the current levels of human intervention.

There are documents with management for review exploring improved efficiencies using item barcode.

# Dependencies

The following list of sub systems depict a dependency for despatching to function:

* Sales orders
* Pick, check and pack
* Printing / parcel of carton labels

Further sub system dependencies are as follows:

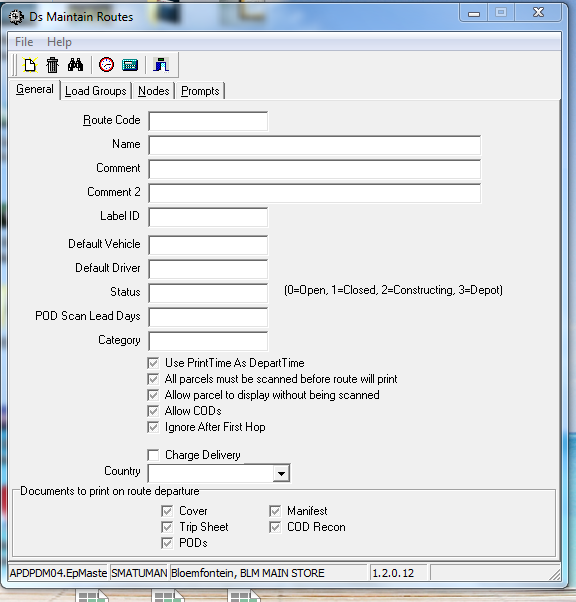
* Customer sales order would have to have received a valid delivery node allocation at the time of sales order creation / before releasing the sales order.
  + To note these node codes are verified against a route to which it is allocated
  + The sub system auto calculates the correct route allocation based of best time to departure, in the event of the node code being allocated to multiple route codes
  + It is a management responsible to ensure that routes and related node codes form an efficient / permissible combination
  + These node / route combinations are agreed to with one or more delivery service providers, including own fleet facilities.
  + ***Customer to collect*** is a valid route and node combination that depicts the collection counter
* Picking sub system. This sub system is delivery node and route allocation aware and provides the following feedback to related users:
  + Picking jobs within a parameter defined time to complete is displayed without any emphasis
  + Picking jobs within a parameter defined time period to route departure not completed are displayed in orange raising the system urgency to complete to meet with the planned departure time
  + As soon as a route departure time has almost been reached (parameter defined) these are displayed in red

# Route maintenance

It cannot be emphasised enough that this part of the sub system requires detailed planning and must be managed by a person of a responsible nature. The impact of incorrect configuration can be a root cause of customer not receiving their consignments on time.

Due to the nature of many of our work force and customers is that goods mis-directed can become part of stock shrinkage and unnecessary investigations with poor customer experiences.

**Diagram 7.1 depicts the main route maintenance screen**

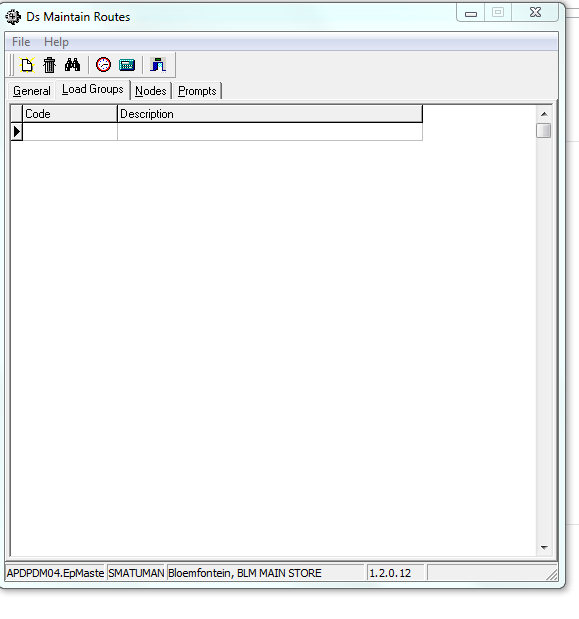


There are 4 tabs in the screen with the following functionality:

* **General** is to define a route and all related profiling details.
  + **Route Code** is an alpha numeric codeand can be anything but normally includes some characters that provide visual meaning to the users
  + **Name** describes the route
  + **Comments** is used to assist the users and also printed on the paper documentation
  + **Label Id** – is used to print a barcode on the parcel label using a specific prefix
  + **Default vehicle** – the same vehicle is used to perform deliveries for a specific route. This can be changed at time of printing the route delivery documentation and is used to assist the preparation to print process. However, some statistics can be derived from this as well
  + **Default Driver** – as a rule most times the same driver is assigned to a specific set of routes due to knowledge and experience. This can be overridden at document printing time
  + **Status** – this is the status of the route.
  + **POD Scan Lead Time** - this is used to check if the POD has been returned and processed within the designated SLA
  + **Category –** statistical use
  + **Charge delivery -** in some instances customers on a route will be charged for the delivery
  + **Country –** this is used to control compliance with export / import regulations.
  + **Documents to print –** these check box options allows optimisation of the route documentation to be printed
* **Load Groups tab**

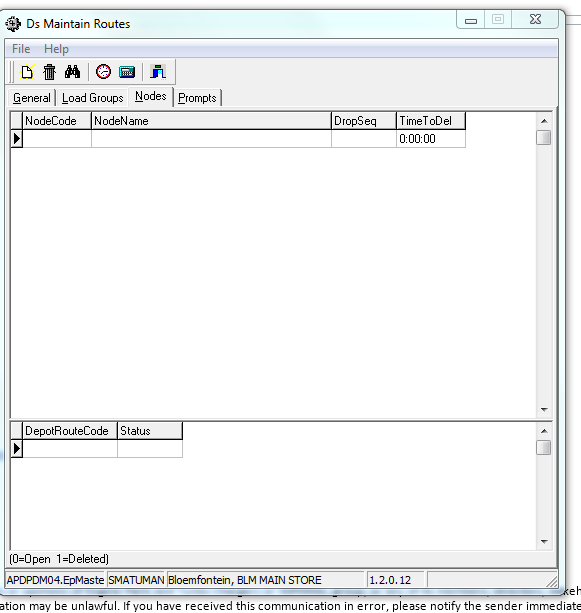
One or more routes can be grouped together for collective loading into the same vehicle.

This is in addition to the loading manifest used to load a vehicle at point of warehouse departure where several load groups are collectively loaded to one vehicle. The loading manifest attempts to order the loading sequence such that the last to deliver is loaded into the vehicle first.

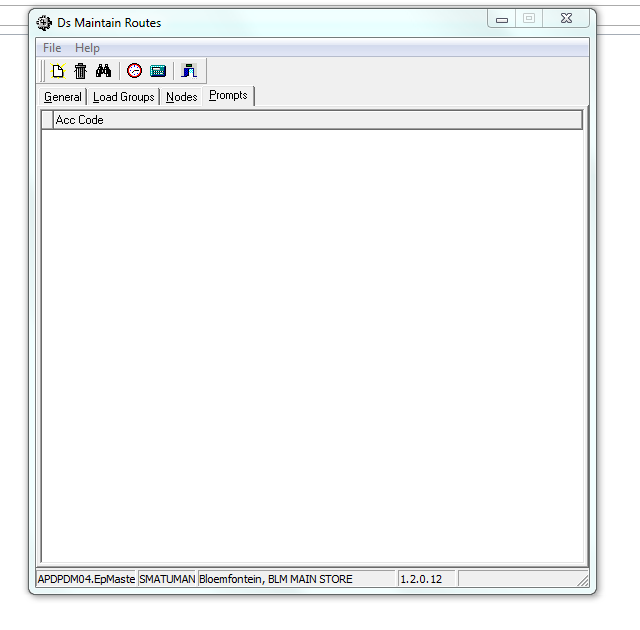


* **Nodes tab**

This a format used to define a drop point along a route and depicts the sequence in which the drop offs will be completed including the standardised time that the delivery is usually expected



* **Prompts tab**



This tab permits the association of one or more messages association with the specific route and customer to assist the driver, especially a new driver, on customer delivery expectations

# Multi-warehouse route resolution

In the original Engineparts business configuration, there were a number of business and warehouse operations country wide.

The ePart solution provides sales and self-service customers the ability to place orders on any of the warehouse locations, all for the same delivery node.

The orders are individually picked, packed and shipped from each originating warehouse.

As part of the BSc Dissertation was to algorithmically resolve the routes required to effect delivery.

By way of an example as it was at the time:

* Customer in Cape town places an order on Durban warehouse
* The order is shipped to Bloemfontein warehouse and the
* Routed to Cape Town warehouse
* At Cape Town the consignment follows the standard delivery model
* As part of the algorithm if the vehicle Durban to Bloemfontein had departed already but the vehicle Durban to Johannesburg was available then the algorithm would route via the Johannesburg branch

To ensure that consignments are managed correctly, staff at each branch would scan the shipping tracing number, and if a “via” consignment”, the following occurs:

* Receipt confirmation for location tracking created in the DB
* The parcel is allocated the appropriate (next) route
* The receiving staff would be made aware on screen and by training that the parcel(s) are to be sent to the despatch area (and not opened for inspection)
* The despatching department would scan receipt of the “via” parcel(s)
* The loading procedure would expect to receive and load for despatching
* The POD process accounts for this as well

# Current functionality

The initial systems design was around a classic ***closed distribution*** model where ONLY Engineparts freight would be participate.

Several attempts were made to include customer deliveries for a fee. However, this impacted on several business areas negatively and never was considered a successful endeavour.

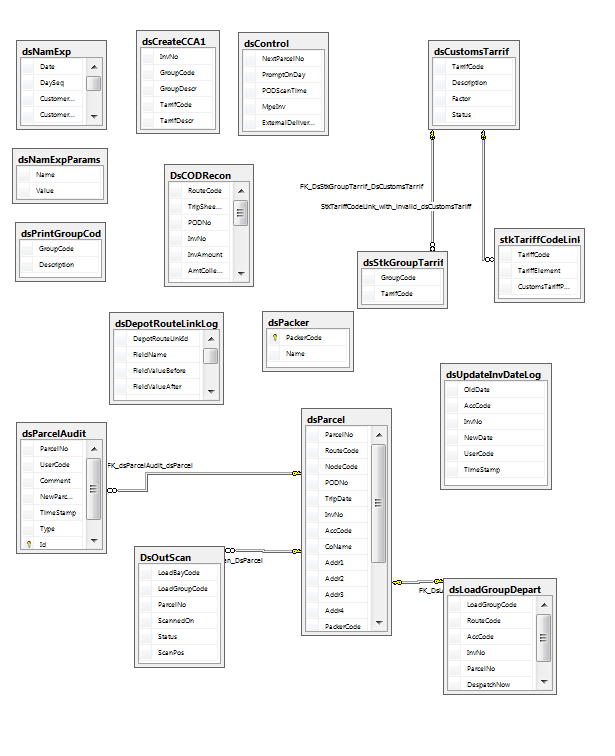
Post liquidation, the notion of free / inclusive delivery costs was changed to a charge per delivery model.

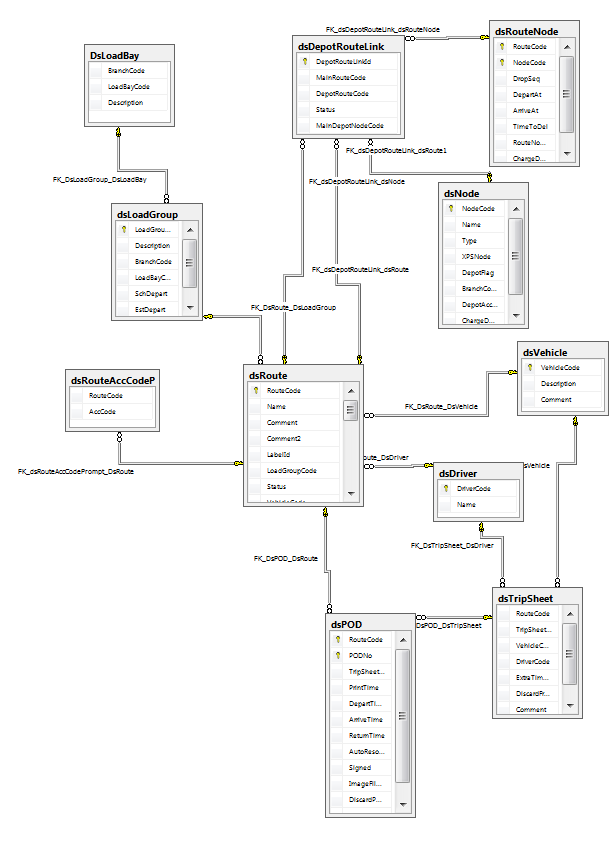
Also, multiple service providers were engaged requiring an awareness of which service provider completed which delivery request for billing and SLA controls

By virtue of the application code design, these requirements were efficiently completed in 2 weeks and implemented

The Engineparts in-house closed distribution sub system effectively parallels the service provider functionality allowing for effective and automated account reconciliations and SLA validation

# Database entities and relationships





# Programs

# MS Windows Executables

Not all programs are commented on as they are self describing

|  |  |
| --- | --- |
| **Name** | **Description** |
|  |  |
| dsAlterDeliveryNodeCode | Change the default node on delivery route |
| dsAsyExport | ASYCUDA – United Nations export / import integration |
| dsBotExp | Botswana export integration ASYCUDA variation |
| DsCreateCCA1 | Old style export / import – not used |
| DsCreateLabels |  |
| DsDeleteLabel |  |
| DsDeleteTripSheets |  |
| DsIncorrectDelivery | Stats for incorrect deliveries for parcels not dropped off |
| DsLoadGroupDepart |  |
| dsMaintainCCA1DebtorCountry | SAD500 replaces this |
| DsMaintainDrivers |  |
| dsMaintainIncorrectDeliveryType |  |
| DsMaintainLoadBay |  |
| DsMaintainLoadGroups |  |
| DsMaintainNodes |  |
| dsMaintainOutscanVerifyLocs |  |
| DsMaintainPackers |  |
| dsMaintainRouteDefVehicles | Vehicles participate in selected default routes |
| DsMaintainRoutes |  |
| dsMaintainSADInvValidation |  |
| DsMaintainTariffs | International import / export tariff codes |
| DsMaintainVehicles |  |
| dsNamAutoCreateEDIFiles | Integration with Namibia government system |
| dsNamAutoCreateSADFiles | Integration with Namibia government system |
| dsNamAutoCreateSADFilesConsol | Integration with Namibia government system |
| DsOutScan |  |
| dsPrintNamInvoice |  |
| DsProcessPODs |  |
| DsReprint |  |
| DsRerouteParcel | Parcel defaulted to wrong route & fix |
| DsRouteReturn | Show parcels on return route |
| dsSADLookup | Not used – out of date |
| DsScanCoverSheet | Trip sheet booklet summary sheet |
| DsSearchDriver |  |
| DsSearchLoadGroup |  |
| DsSearchNode |  |
| DsSearchPacker |  |
| DsSearchRoute |  |
| DsSearchTripSheet |  |
| DsSearchVehicle |  |
| DsTripAdjust | Courier contracted SLA adjustments |
| DsViewPODs |  |

# SQL Stored Procedures

|  |  |
| --- | --- |
| **Name** | **Description** |
| dsAlterDeliveryNodeCode.sql |  |
| DsAutoGenNewPODNo.sql |  |
| DsBldCODRecon.sql |  |
| dsBotPrinted.sql |  |
| DsCheckParcelBayId.sql |  |
| dsCourierDeliveryCharge.sql |  |
| dsCoverSheetOutScan.sql |  |
| DsCreateLabels.sql |  |
| dsCreateLabelsCheckSerial.sql |  |
| DsCreateNamExpFile.sql |  |
| dsDebtorCountryAssign.sql |  |
| dsDebtorCountrySearch.sql |  |
| dsDebtorCountryValidation.sql |  |
| DsDeleteLabel.sql |  |
| DsDeleteRoute.sql |  |
| DsDeleteTripSheet.sql |  |
| dsDeleteXPSCode.sql |  |
| DsDriverSchedule.sql |  |
| dsExtDailyParcels.sql |  |
| dsExtDailyParcelsFilter.sql |  |
| dsExtDailyParcelsRTT.sql |  |
| dsGetBotExp.sql |  |
| dsGetNamEdiCons.sql |  |
| dsGetNamEdiGroup.sql |  |
| dsGetNamExp.sql |  |
| dsGetNamExpEdi.sql |  |
| dsGetNamExpEdiCdvValue.sql |  |
| dsGetNamExpSAD.sql |  |
| dsGetNamExpSADConsol.sql |  |
| DsGetNextNamExp.sql |  |
| DsGetNextParcelNo.sql |  |
| DsGetNextPODNo.sql |  |
| DsGetNextTripSheetNo.sql |  |
| DsImportInvoices.sql |  |
| DsImportRoutes.sql |  |
| DsIncorrectDelivery.sql |  |
| dsInsertParcelLog.sql |  |
| DsInsertRouteNodeAVG.sql |  |
| dsMailPOD.sql |  |
| dsMaintSADInvValLineTotals.sql |  |
| dsMaintSADInvValTotals.sql |  |
| DsNamExpUpdateCustomerTin.sql |  |
| DsParcelDateReport.sql |  |
| DsParcelDateRouteReport.sql |  |
| DsPrepareCreateCCA1.sql |  |
| DsPrepareLoadGroupDepart.sql |  |
| DsProcessCreateCCA1.sql |  |
| DsProcessLoadGroupDepart.sql |  |
| DsReportLoadingManifest.sql |  |
| DsReportNodeRoute.sql |  |
| DsReportPODs.sql |  |
| DsReportRouteNode.sql |  |
| DsReportRouteNodeAVG.sql |  |
| DsReportTripSheet.sql |  |
| DsReportUnscannedParcels.sql |  |
| DsRerouteParcel.sql |  |
| DsResetLoadGroup.sql |  |
| DsResolveRoute.sql |  |
| DsResolveRoute1.sql |  |
| dsResolveRouteBase.sql |  |
| dsResolveRouteByCategory.sql |  |
| DsResolveRouteOn.sql |  |
| dsRouteDefVeh.sql |  |
| dsRouteNodeChange.sql |  |
| dsRouteSetDefVeh.sql |  |
| dsRpt3KS\_ParcelScanStatus.sql |  |
| DsRptAccCodeParcels.sql |  |
| dsRptAchievementByRoute.sql |  |
| dsRptBlankPOD.sql |  |
| dsRptBorderVAT.sql |  |
| DsRptBranchParcelCnt.sql |  |
| dsRptBWVATSchedule.sql |  |
| DsRptCCA1consol.sql |  |
| DsRptCODRecon.sql |  |
| DsRptCODs.sql |  |
| dsRptCounterSalesProfit.sql |  |
| dsRptCourierManifest.sql |  |
| dsRptCoverSheetScanByDate.sql |  |
| dsRptCustDelAddrByTown.sql |  |
| dsRptCustomersByRouteNodes.sql |  |
| dsRptCustomsReturns.sql |  |
| dsRptDeclarant.sql |  |
| DsRptDrivers.sql |  |
| dsRptExpConfirmProcess.sql |  |
| dsRptExtParcelManifest.sql |  |
| dsRptExtParcelScanComparison.sql |  |
| dsRptGetNamExpEdiManifest.sql |  |
| dsRptIncorrectDeliveries.sql |  |
| DsRptInvMessage.sql |  |
| DsRptInvNoParcels.sql |  |
| DsRptLabel.sql |  |
| DsRptLabeledParcels.sql |  |
| DsRptLabelsNotScanned.sql |  |
| DsRptLaneStatistics.sql |  |
| dsRptListOfAllNodes.sql |  |
| dsRptListViaParcelPerNode.sql |  |
| dsRptListViaParcels.sql |  |
| DsRptLoadGroupRoute.sql |  |
| dsRptLoadGroupStats.sql |  |
| DsRptLoadGroupUnscannedParcels.sql |  |
| dsRptMachineLabelStats.sql |  |
| DsRptMultiParcelInv.sql |  |
| dsRptMultipleLabels15MinDiff.sql |  |
| DsRptMultipleLabelsPerInvoice.sql |  |
| DsRptOutScanManifest.sql |  |
| dsRptParcelDeleteList.sql |  |
| dsRptParcelScanHistory.sql |  |
| dsRptParcelsScannedAndDeleted.sql |  |
| dsRptParcelWeightVariance.sql |  |
| DsRptPODsArriveLate.sql |  |
| dsRptPODsList.sql |  |
| DsRptPODsReturnLate.sql |  |
| dsRptRegAssExceptions.sql |  |
| DsRptRouteAvgDelTime.sql |  |
| DsRptRouteCharge.sql |  |
| DsRptRouteCover.sql |  |
| dsRptRouteCustomerProfit.sql |  |
| DsRptRouteCustProfit.sql |  |
| dsRptRouteDeparturesByNode.sql |  |
| DsRptRouteManifest.sql |  |
| DsRptRouteParcelCnt.sql |  |
| DsRptRouteProfit.sql |  |
| DsRptRoutesOnLoadGroup.sql |  |
| dsRptRouteStatistics.sql |  |
| dsRptRouteStatistics\_CustomerDetail.sql |  |
| dsRptRouteStatistics\_InvoiceDetail.sql |  |
| dsRptSAD500.sql |  |
| dsRptSAD500consol2.sql |  |
| dsRptSAD500ConsolBW.sql |  |
| dsRptSADConsolCustInvList.sql |  |
| dsRptScanDiscrepancies.sql |  |
| dsRptSFD\_ParcelScanStatus.sql |  |
| dsRptTripAdjustments.sql |  |
| dsRptTurnoverSummaryForRoutes.sql |  |
| DsRptUnlabeledInvoices.sql |  |
| DsRptUnreturnedPODs.sql |  |
| dsRptVAT.sql |  |
| DsRptVehicles.sql |  |
| DsRptXpsContract.sql |  |
| dsSearchDepotRoutes.sql |  |
| DsSearchDriver.sql |  |
| dsSearchGroupCodes.sql |  |
| DsSearchLoadGroup.sql |  |
| DsSearchNode.sql |  |
| DsSearchPacker.sql |  |
| DsSearchPOD.sql |  |
| DsSearchRoute.sql |  |
| dsSearchTarrifCode.sql |  |
| DsSearchTripSheet.sql |  |
| DsSearchVehicle.sql | Find in-house vehicle |
| DsSetNamExpPrinted.sql | Update printed status |
| DsUnprepareCreateCCA1.sql | Not used old format export documentation |
| DsUnprepareLoadGroupDepart.sql |  |

|  |  |
| --- | --- |
| Trigger Name | Description |
| DsControlValidateInsertDelete.sql | Check permitted combinations |
| dsCreateCCA1TarrifValidation.sql | CCA1 - Old style export documentation – not used |
| DsDebtorCountryUpdateLog.sql | Logs updates of country types |
| dsDepotRouteLinkLogs.sql | Logs changes in the depot nodes to route linking |
| DsNodeValidateType.sql | Node can be of different type. This trigger checks if of a valid type |
| DsParcelAuditValidateType.sql | Parcels can be of differing types. Validate that the type being inserted is of a valid type |
| DsParcelValidateStatus.sql | Check if parcel label (barcode) is active. Can be that label was printed and deleted (with a reason code) |
| dsRouteNodeLogs.sql | Change log |
| DsRouteValidateStatus.sql | Check if route code used is active and applicable for the node used |
|  |  |
|  |  |

# Risks and mitigation

|  |  |  |
| --- | --- | --- |
| # | Risk | Mitigation |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |
| 4 |  |  |
| 5 |  |  |
| 6 |  |  |

# Requirements overview

|  |  |  |
| --- | --- | --- |
| # | Description | Action / By whom |
| 1 |  |  |
| 2 |  |  |
| 3 |  |  |
| 4 |  |  |
| 5 |  |  |
| 6 |  |  |
| 7 |  |  |
| 8 |  |  |
| 9 |  |  |
| 10 |  |  |
| 11 |  |  |
| 12 |  |  |

# Acceptance

I hereby confirm that I have been fully informed of the documents content and, received training to understand how the detailed instructions are to be applied

Name …………………………………………………………………………….

Job Title ………………………………………………………………………….

Signed ……………………………………………………………………………

Date ………………………………………………………………………………