

**Version 1.1**

**Date: 2019/02/25**

Abstract

**Engineparts customer, after purchasing goods may return goods as long as they are in good order and within a stipulated time frame.**

**In some instances goods are returned as faulty and may be filtered through the claims system.**

Return for Credit

Control Debtors expectations vs Actual

**Table of Contents**

Document approval and distribution list 2

1. Introduction 3

2. Audience 3

3. Objectives 3

4. Database entities and relationships 4

5. Programs 4

5.1 MS Windows Executables 4

5.2 SQL Stored Procedures 4

6. Acceptance 6

# Document approval and distribution list

|  |  |  |  |
| --- | --- | --- | --- |
| **Description** | **Name / Title** | **Signature** | **Date** |
| **Document Type / purpose** | | | |
| Prepared by |  |  |  |
| Reviewed by |  |  |  |
| Approved by |  |  |  |

# Introduction

Engineparts customer, after purchasing goods may return goods on the provision that the goods are in good order and within a stipulated time frame.

In some instances, goods are returned as faulty and may require filtering through the claims sub-system.

# Audience

* Debtors
* Warehouse RFC
* Warehouse administration
* Despatching

# Objectives

The objective is, in context, to maintain a good customer experience by having an almost unrestricted returns policy subject to the state of the returned goods and returns to be made within a defined time limit

Reports are available in ePart to highlight customer with higher than ***normal*** returns. These are often reviewed directly with the customer by the sales representative

Another report is available to indicate specific parts on which there are very high returns incidents. These could indicate:

* Faulty recording in the catalogue sub system supporting the sales process and thus supplying the wrong part. This is usually followed up by the cataloguing staff for correction if applicable
* It could be that some items could be fragile and break during the transportation process.

All parts received back from customers are inspected before confirming the transaction.

* Parts received in a faulty state, without prior notification from the customer, is returned
* Goods received later than the expiry date, is returned directly to the customer. This should change and also form part of the claims system for full tracking until resolved.
* Received returns with faulty prior notification is recorded in the claims system for investigation and resolution.
* Goods qualifying for GRV are captured and posted. At this time the following happens:
  + Debtors entry is posted reflecting the credit on the debtors account.
  + A binning report is generated that the warehouse staff need to collect with the goods and to place it in the indicated bin location.
  + As soon as the binning task is confirmed complete, the received stock becomes available for selling purposes, unless the indicated bin is a non-sale bin.
* Financial governance requires that the returned goods be associated with a sales invoice. From this association, the original cost and sales value is derived, and the supporting financial transactions generated.
* Additionally, the returned goods are booked back into stock at the original sales transaction cost value and a new AUC calculated.

# image.pngDatabase entities and relationships

# Programs

# MS Windows Executables

|  |  |
| --- | --- |
| **Name** | **Description** |
| RfcPosting | Detail of RFC captured and operator requests captured data be applied |
| rfcReprintRFC | Reprint supporting documentation |
| rfcRFCView | Search and view |
| rfcMaintainQtyChangeReasons | Maintain standardised reason codes |

# SQL Stored Procedures

|  |  |
| --- | --- |
| **Name** | **Description** |
|  |  |
| rfcAddLine.sql | Add a returned item line to document |
| rfcCollectOrderAutoRelease.sql | Release to-collect instruction via despatching sub system |
| rfcCreateCollectOrder.sql | Create sales order as collection instruction |
| rfcDeleteLine.sql | Delete line pre-posted state |
| rfcFindSurchargeVal.sql | Calculate delivery surcharge |
| rfcGetCtrlRef.sql | Get next unique tracking no |
| rfcGetRFCList.sql | Get a list of RFC’s by status |
| rfcPostingEMail.sql | Notification e-mail |
| RfcPostingLoad.sql | Load unposted document |
| RfcPostingPost.sql | Apply captured detail and seal against any further changes |
| RfcPostingReprintSearch.sql | Search for RFC to re-print |
| RfcPostingSearch.sql |  |
| RfcPostingSurcharge.sql | Calc and apply surcharge |
| rfcRptCreditNoteRFC.sql | Reporting support |
| rfcRptIncompleteRFCs.sql | Unposted RFC |
| rfcRptMultipleRFCs.sql | Group and print |
| rfcRptOutstandingCollections.sql | RFC requiring collection not completed |
| rfcRptPostingActivityByUser.sql |  |
| rfcRptRFCbySalesman.sql |  |
| rfcRptRFCByUserByDateRange.sql |  |
| rfcRptRFCforPeriod.sql |  |
| rfcRptRFCsOnOnlineSales.sql |  |
| rfcRptRFCSuspense.sql | RFC placed in suspense mode – not WIP |
| rfcRptRFCsWithNonReturnItems.sql | List RFC with items that may not be returned as part of governance |
| rfcRptTopRfcUsers.sql |  |
| rfcRptTopRfcValueByUser.sql |  |
| rfcSearchTable.sql |  |
| rfcViewUnallocatedRFCs.sql |  |
| rfcViewUnpostedJob.sql |  |
| rfcViewUnpostedJobs.sql |  |
| stkGetRFCPrint |  |
| stkGetRFCPrint |  |
| rfcGetCtrlRef | Get unique transactin tracking reference |
| stkRFCGetInvoiceDate |  |
| coRptLog | Log reporting event |
| coUserPermission | Check user permissions |
| stkRFCDetQtyLog | Log actual qty posted for control purposes |
| rfcPostingEMail | Confirm posting via e-mail |
|  |  |

# 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 ………………………………………………………………………………