

MD.070 Application Extensions

Technical Design

2 Degrees

Oracle E Business Suite R12

Basware Integration

PO.02.11 2D Basware PO Import

v1.0

**Dart Arellano**

06 June 2018

Document Control

Document History

|  |  |  |  |
| --- | --- | --- | --- |
| Version | Name | Date | Comment |
| Draft1A | Dart Arellano | 06-Jun-2018 | Initial version |
| 1.0 | Dart Arellano |  | Approved and released to 2 Degrees |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

Distribution

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Title/Position | Company | Date |
|  | Library Master | DXC RED ROCK | 06-June-2018 |
| Miles Harvey | Internal Systems Support Lead | 2 Degrees |  |
| John Marais | ePMO Project Manager | 2 Degrees |  |
| Matt Farrer | Project Manager, Consulting Manager |  |  |

Reviewers

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Title/Position | Review Date | Reviewed Version |
| Dirk Verbeek | Technical Consultant | 11-June-2018 | Draft1A |
| Matt Farrer | Project Manager, Consulting Manager | 11-June-2018 | Draft1A |

Approval

|  |  |  |  |
| --- | --- | --- | --- |
| Name | Title/Position | Date | Approval |
|  |  |  |  |

Table of Contents

[1 Technical Overview 5](#_Toc516744708)

[1.1 Approach 5](#_Toc516744709)

[1.2 Module List 5](#_Toc516744710)

[1.3 Prerequisite 5](#_Toc516744711)

[1.4 Document Reference 5](#_Toc516744712)

[2 2D Basware PO Import - Concurrent Program Logic 6](#_Toc516744713)

[2.1 Design Features 6](#_Toc516744714)

[2.2 2D Basware PO Import Process 7](#_Toc516744715)

[2.2.1 High Level Process Flow 7](#_Toc516744716)

[2.2.2 Program Units 7](#_Toc516744717)

[2.2.3 Referenced Standard Objects 8](#_Toc516744718)

[2.2.4 Common Interface API 10](#_Toc516744719)

[2.3 Concurrent Program Register 10](#_Toc516744720)

[2.4 Concurrent Program Calling Arguments 10](#_Toc516744721)

[2.5 Incompatibility 11](#_Toc516744722)

[2.6 Profile Option 11](#_Toc516744723)

[2.7 Input File 11](#_Toc516744724)

[2.7.1 Data Mapping 11](#_Toc516744725)

[2.8 Output File 11](#_Toc516744726)

[2.8.1 Common Interface Errors Report 11](#_Toc516744727)

[2.8.2 Common Interface Run Report 12](#_Toc516744728)

[2.9 Log File 12](#_Toc516744729)

[2.10 Related Concurrent Programs 13](#_Toc516744730)

[2.11 Special Feature 13](#_Toc516744731)

[3 Database Design 14](#_Toc516744732)

[3.1 Database Diagram 14](#_Toc516744733)

[3.2 Database Objects 14](#_Toc516744734)

[3.2.1 Sequences 14](#_Toc516744735)

[3.2.2 Tables 14](#_Toc516744736)

[3.2.3 Views 15](#_Toc516744737)

[4 Installation Instructions 16](#_Toc516744738)

[4.1 Code Release Cycle 16](#_Toc516744739)

[5 Implementation Notes 17](#_Toc516744740)

[5.1 Coding Summary 17](#_Toc516744741)

[5.2 Program Files 17](#_Toc516744742)

[5.3 Testing Summary 17](#_Toc516744743)

[6 Open and Closed Document Issues 18](#_Toc516744744)

[6.1 Open Issues 18](#_Toc516744745)

[6.2 Closed Issues 18](#_Toc516744746)

# Technical Overview

This document defines the technical components required to implement customisation . This Application Extension Technical Design document complements the Application Extension Functional Design document for the Basware PO to ERP process, consider the set to be the complete detailed design.

## Approach

* Build a process using PL/SQL procedures and functions to automate importing of Orders from Basware into Oracle Purchasing
* Adopt the Stage-Transform-Load process structure to support extensibility and efficiency
* Utilise the Oracle standard Purchasing Documents Open Interface to import data in to the Oracle Purchasing application table
* Register the process as a concurrent program to run overnight on a regular basis

## Module List

|  |  |
| --- | --- |
| Concurrent Program | 2D Basware PO Import |

## Prerequisite

The Interface Framework common API delivers a number pre-built functions and procedures regularly used across many interface requirements. Customisation uses most of the accessible functionality offered, hence installation of Interface Framework API is required.

|  |  |
| --- | --- |
| Database Packages | XXFND\_COMMON\_PKG  XXFND\_COMMON\_INT\_PKG |
| Concurrent Programs | 2D Common Interface Errors Report  2D Common Interface Run Report |

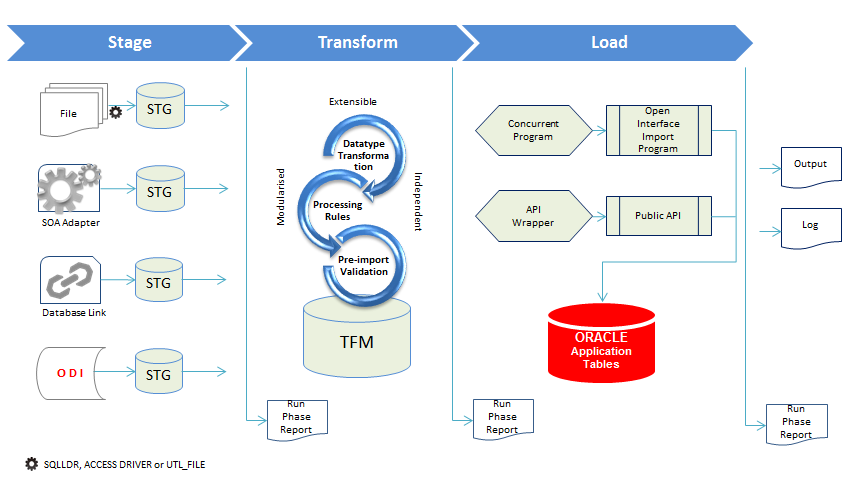
## Document Reference

| Doc ID | File Name | Date | Author |
| --- | --- | --- | --- |
|  |  |  |  |
| IDD 1.15 | Two Degrees Interface Description Document\_v1.15.pdf | 05-May-2018 | Raimo Herttua |
| MD.050 (PO to ERP) | MD 050 PO to ERP Functional Specification\_V0.3.docx | 30-May-2018 | Jennifer Pole  Vaughn Hunt |

# 2D Basware PO Import - Concurrent Program Logic

concurrent program

* Adopts the Stage-Transform-Load interface framework design
* Uses Oracle built-in database utilities for accessing and loading data files
* Implements modularised program units for reusability
* Implements extensible transformation routines for derivations, translations, from-to mapping, system parameter check (e.g. open period, operating unit, profiles, auto numbering, etc..), including pre-import validations
* Uses Oracle standard Open Interface concurrent programs for batch processing, and/or public APIs for real-time integrations that require callback or immediate response



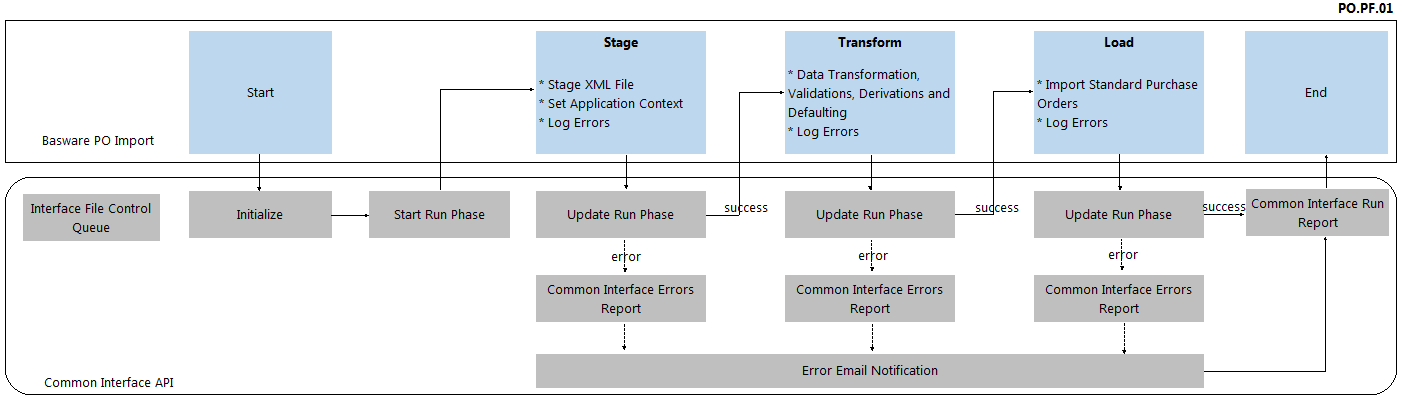
## Design Features

|  |  |
| --- | --- |
| Consistent Processing Framework | It follows the process structure where stages are defined and present to effectively capture errors where and when it occurred in the interface process. |
| Reusable Programs | All programs are written to enable future proofing. Additional changes and enhancements to the interface program should be able to be performed as easily as possible. Programs are written so that they can be generic and re-used in the future by other systems integrating to Oracle. |
| Common Reporting | Centralised reporting is an important feature of the framework design. Each interface stage will report on status and validation errors using the same method. Common reports are written in BI Publisher producing a standard and consistent error report output that can be re-used by other interface definitions. |

## 2D Basware PO Import Process

### High Level Process Flow

PO.PF.01 high level diagram below illustrates the Purchase Order import process flow in a linear view. The top and bottom separation point out Basware PO Import specific program units from the common API components, as well as show when and where they are being used.



### Program Units

| Description | Program Unit | Interface Run Phase | Scope |
| --- | --- | --- | --- |
|  |  |  |  |
| Interface File Control Queue | xxfnd\_interface\_ctl |  | Common Interface API |
| Main Program | xxpo\_purchase\_order\_pkg.run\_po\_import |  | Purchase Order Interface Wrapper |
| Initialize Interface Run Phases Stage>Transform>Load | xxfnd\_common\_int\_pkg.initialise\_run |  | Common Interface API |
| Start Run Phase | xxfnd\_common\_int\_pkg.start\_run\_phase |  | Common Interface API |
| Update Run Phase | xxfnd\_common\_int\_pkg.update\_run\_phase xxfnd\_common\_int\_pkg.end\_run\_phase |  | Common Interface API |
| Copy XML File to the Staging Directory | xxfnd\_common\_pkg.file\_copy | Stage | Common Interface API |
| Delete XML File from the Inbound Directory | xxfnd\_common\_pkg.file\_delete | Stage | Common Interface API |
| Stage XML File | xxfnd\_common\_int\_pkg.stage\_xml\_file | Stage | Common Interface API |
| Set Application Context | set\_context() | Stage | Purchase Order Interface Wrapper |
| Data Transformation, Validations, Derivations and Defaulting | validate\_open\_period() get\_record\_id() get\_agent\_id() validate\_doc\_num() get\_supplier() get\_currency\_code() get\_ap\_terms() convert\_to\_date() convert\_to\_number() assign\_location() get\_user() get\_uom\_code() calculate\_unit\_price() get\_tax\_name() get\_ccid() validate\_capex\_code() get\_category\_id() | Transform | Purchase Order Interface Wrapper |
| Import Standard Purchase Orders | POXPOPDOI | Load | Purchase Order Interface Wrapper |
| Log Error | xxfnd\_common\_int\_pkg.raise\_error | Stage, Transform, or Load | Common Interface API |
| Error Email Notification | xxfnd\_common\_pkg.send\_mail | Stage, Transform, or Load | Common Interface API |
| Common Interface Errors Report | xxfnd\_common\_int\_pkg.launch\_error\_report | Stage, Transform, or Load | Common Interface API |
| Common Interface Run Report | xxfnd\_common\_int\_pkg.launch\_run\_report |  | Common Interface API |

### Referenced Standard Objects

| Owner | Object | Type |
| --- | --- | --- |
|  |  |  |
| APPS | ap\_lookup\_codes | VIEW |
| APPS | ap\_supplier\_sites | TABLE |
| APPS | ap\_supplier\_sites\_all | TABLE |
| APPS | ap\_suppliers | TABLE |
| APPS | ap\_terms | TABLE |
| APPS | ap\_terms\_tl | TABLE |
| APPS | financials\_system\_parameters | TABLE |
| APPS | fnd\_application | TABLE |
| APPS | fnd\_concurrent | PACKAGE |
| APPS | fnd\_concurrent\_programs\_tl | TABLE |
| APPS | fnd\_concurrent\_requests | TABLE |
| APPS | fnd\_currencies | TABLE |
| APPS | fnd\_date | PACKAGE |
| APPS | fnd\_file | PACKAGE |
| APPS | fnd\_flex\_ext | PACKAGE |
| APPS | fnd\_flex\_value\_sets | TABLE |
| APPS | fnd\_flex\_values\_vl | VIEW |
| APPS | fnd\_global | PACKAGE |
| APPS | fnd\_id\_flex\_structures\_vl | VIEW |
| APPS | fnd\_lookup\_values | TABLE |
| APPS | fnd\_lookup\_values\_vl | VIEW |
| APPS | fnd\_profile | PACKAGE |
| APPS | fnd\_profile\_option\_values | TABLE |
| APPS | fnd\_profile\_options | TABLE |
| APPS | fnd\_request | PACKAGE |
| APPS | fnd\_responsibility | TABLE |
| APPS | fnd\_user | TABLE |
| APPS | fnd\_user\_resp\_groups\_all | VIEW |
| APPS | gl\_ledgers | TABLE |
| APPS | gl\_period\_statuses | TABLE |
| APPS | gl\_sets\_of\_books | VIEW |
| APPS | hr\_operating\_units | VIEW |
| APPS | mo\_global | PACKAGE |
| APPS | mtl\_categories\_b | TABLE |
| APPS | po\_buyers\_v | VIEW |
| APPS | po\_distributions\_all | TABLE |
| APPS | po\_distributions\_interface | TABLE |
| APPS | po\_distributions\_interface\_s | SEQUENCE |
| APPS | po\_headers\_all | TABLE |
| APPS | po\_headers\_interface | TABLE |
| APPS | po\_headers\_interface\_s | SEQUENCE |
| APPS | po\_interface\_errors | TABLE |
| APPS | po\_line\_locations\_all | TABLE |
| APPS | po\_lines\_all | TABLE |
| APPS | po\_lines\_interface | TABLE |
| APPS | po\_lines\_interface\_s | SEQUENCE |
| APPS | po\_units\_of\_measure\_val\_v | VIEW |
| APPS | rcv\_transactions | TABLE |
| APPS | zx\_accounts | TABLE |
| APPS | zx\_rates\_vl | VIEW |

### Common Interface API

Part of the 2D Basware PO Import concurrent program are the Interface Framework common APIs involved in storing and managing information for an interface and its related process transactions –

* Object Type (Order, Invoice, Supplier, Chart of Accounts, Response, etc.)
* Source (Basware, Oracle)
* Purpose (Data Mastering, Transaction Transfer)
* Interface Direction (Inbound or Outbound)
* Transitions (Stage-Transform-Load)

The Interface Framework API provides a common error report available to all defined interface members. It specifies the error details and at which phase the error has occurred. Separately, a summary report is generated to display additional information relating to the interface run such as record count, success count, error count, hash total, and the source tables that were used.

It includes filesystem functions for performing copy and delete commands, and for moving files from the source directory to the target directory.

* Inbound: Basware SFTP OUT 🡪 Staging Directory 🡪 Archive
* Outbound: Application Common Out 🡪 Basware SFTP IN

The Interface Framework also provides an FYI notification to inform the System Administrator via email of interface concurrent jobs completing with error status.

## Concurrent Program Register

| Field | Value |
| --- | --- |
|  |  |
| Name | 2D Basware PO Import |
| Short Name | XXPOIMPTM |
| Application | XXNZCL Custom Application |
| Executable | XXPOIMPTM |
| Method | PL/SQL Stored Procedure |
| Executable File Name | xxpo\_purchase\_order\_pkg.run\_po\_import\_all |

## Concurrent Program Calling Arguments

| Argument | Prompt | Value Set | Default Value |
| --- | --- | --- | --- |
|  |  |  |  |
| p\_buyer | Default Buyer | PO\_SRS\_BUYER\_NAME |  |
| p\_debug\_flag | Debug On | Yes\_No | Constant: Yes |

## Incompatibility

| Application | Name | Scope | Type |
| --- | --- | --- | --- |
|  |  |  |  |
| XXNZCL Custom Application | 2D Basware PO Import | Program | Domain |

## Profile Option

| Name | Level | Value Set | Default Value |
| --- | --- | --- | --- |
|  |  |  |  |
| 2D Inbound Directory for Basware | Site | n/a | /basware/inbound |
| 2D Notification Email Address for BW Interfaces | Site | n/a | email@email.com |

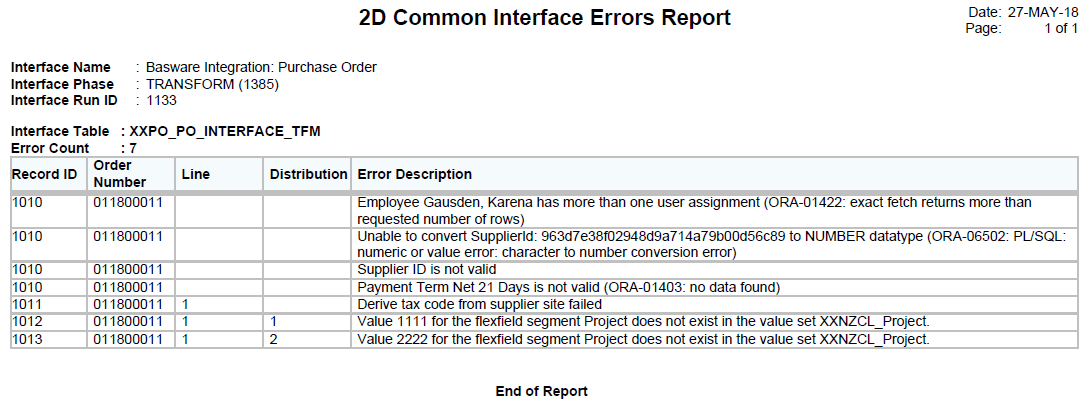
## Input File

### Data Mapping

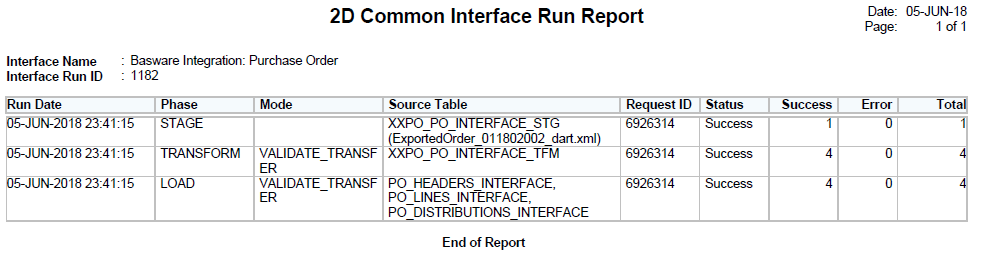
Please refer to Application Extension Functional Design document MD 050 PO to ERP Functional Specification\_V0.3.docx for the detailed data mapping.

## Output File

### Common Interface Errors Report



### Common Interface Run Report



## Log File

Following is a sample log file generated by the concurrent program with parameter Debug On set to Yes

\*\*Starts\*\*05-JUN-2018 23:41:15

\*\*Ends\*\*05-JUN-2018 23:41:25

+---------------------------------------------------------------------------+

Start of log messages from FND\_FILE

+---------------------------------------------------------------------------+

DEBUG: object\_type=ORDER

DEBUG: user\_id=3333

DEBUG: resp\_id=50563

DEBUG: org\_id=82

DEBUG: user\_name=REDROCK1

DEBUG: fetch purchase order xml file from queue

DEBUG: purchase order file ExportedOrder\_011802002\_dart.xml

DEBUG: initialize interface run ids

DEBUG: start STAGE

DEBUG: run\_id=1182

DEBUG: run\_phase\_id=1531

DEBUG: inbound\_file=/basware/inbound/ExportedOrder\_011802002\_dart.xml

DEBUG: inbound\_directory=/basware/inbound/

DEBUG: file\_name=ExportedOrder\_011802002\_dart.xml

DEBUG: stage\_xml\_file.p\_status=S

DEBUG: stage\_xml\_file.p\_message=NULL

DEBUG: set\_of\_books\_id=2021

DEBUG: l\_record\_count=1

DEBUG: l\_success\_count=1

DEBUG: l\_error\_count=0

DEBUG: update run phase STAGE

DEBUG: end STAGE

DEBUG: start TRANSFORM

DEBUG: run\_id=1182

DEBUG: run\_phase\_id=1532

DEBUG: validating order header, line and distribution ids

DEBUG: reference key check completed

DEBUG: transform header record

DEBUG: transform line record(s)

DEBUG: order line 1

DEBUG: tax\_code=GST-12.5%

DEBUG: vendor\_site\_id=278197

DEBUG: transform distribution record(s)

DEBUG: order distribution 1.1

DEBUG: order distribution 1.2

DEBUG: update run phase TRANSFORM

DEBUG: end TRANSFORM

DEBUG: start LOAD

DEBUG: run\_id=1182

DEBUG: run\_phase\_id=1533

DEBUG: populate po interface tables

DEBUG: commit interface data

DEBUG: submit standard po import program

DEBUG: po import program request id 6926315

DEBUG: wait for request to complete

DEBUG: update transformation tables PROCESSED

DEBUG: reconcile with document reference

DEBUG: purging PDOI tables

DEBUG: update control queue table

DEBUG: update run phase LOAD

DEBUG: end LOAD

+---------------------------------------------------------------------------+

End of log messages from FND\_FILE

+---------------------------------------------------------------------------+

## Related Concurrent Programs

* 2D Basware Interface File Queue
* 2D Basware PO Import (sub-request)
* 2D Common Interface Errors Report
* 2D Common Interface Run Report

## Special Feature

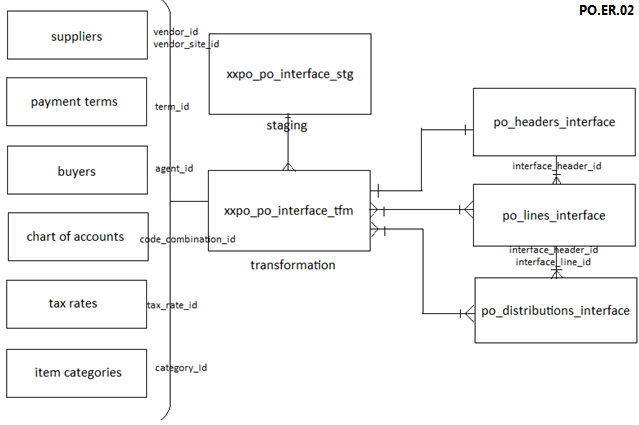
concurrent program has a context switching function. This feature is enabled so that the program can dynamically switch application context when required. Reason being is that, the correct Operating Unit (org\_id) to use can only be determined at the time the file is loaded in to the staging table. If the concurrent program is submitted or scheduled using **Two Degrees Mobile Limited** (org\_id 82) responsibility which corresponds to 01 company code, it can still process incoming orders for company codes 06 and 07 by performing an automatic context switching to **Two Degrees New Zealand Ltd** (org\_id 452) or **Two Degrees Networks Limited** (org\_id 453) respectively.

# Database Design

This section summarizes the new and changed database objects and data required to implement .

## Database Diagram

Below is a high level diagram PO.ER.02 showing the relationship between custom staging and transformation to the target open interface tables. It also points which entity uses the reference (master data) and lookup tables.



## Database Objects

### Sequences

Oracle Sequences definition to support customisation

* XXFND\_INTERFACE\_CTL\_S
* XXFND\_INTERFACE\_STG\_REC\_ID\_S
* XXPO\_PO\_INTERFACE\_TFM\_S

### Tables

Custom Tables definition to support customisation

* XXFND\_INTERFACE\_CTL
* XXFND\_INTERFACE\_STG
* XXFND\_INT\_RUNS
* XXFND\_INT\_RUN\_PHASES
* XXFND\_INT\_RUN\_PHASE\_ERRORS
* XXPO\_DOCUMENT\_REFERENCES
* XXPO\_DOCUMENT\_REFERENCES\_STG
* XXPO\_PO\_INTERFACE\_TFM

### Views

Custom View definition to support customisation

* XXPO\_PO\_INTERFACE\_STG (XXFND\_INTERFACE\_STG)

# Installation Instructions

An automated installation script is provided to migrate customisation to the target environment.

Step 1: Place the TAR file -R-[release number].tar in $XXNZCL\_TOP directory of the target environment (binary mode)

Step 2: Extract the contents of the TAR file

> cd $XXNZCL\_TOP

> tar -xvf -R-[release number].tar

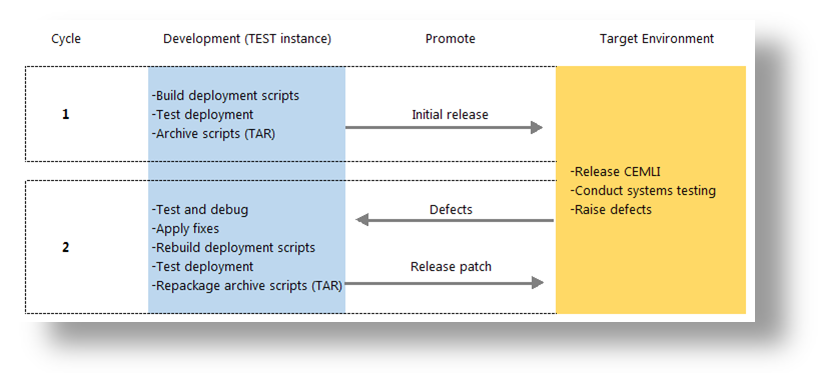
Step 3: Run the install script

> cd install

> chmod +x \_install.sh

> ./\_install.sh [db password] | tee \_install.[release number].log

## Code Release Cycle



# Implementation Notes

This document describes exactly how was developed and implemented at 2 Degrees.

## Coding Summary

Development was done on Linux 2.6.32-431.el6.i686 server, host name snzclakl025.nzc.co.nz, to work with Release 12.1.3 of Oracle Applications. All coding follows the standards defined in Oracle E Business Suite Developer’s Guide Part No. E12897-4.

## Program Files

| Type | Name | Location |
| --- | --- | --- |
|  |  |  |
| DDL | XXPO\_PO\_INTERFACE\_STG\_DDL.sql | $XXNZCL\_TOP/install/sql |
| DDL | XXPO\_PO\_INTERFACE\_TFM\_DDL.sql | $XXNZCL\_TOP/install/sql |
| DDL | XXPO\_DOCUMENT\_REFERENCES\_STG\_DDL.sql | $XXNZCL\_TOP/install/sql |
| DDL | XXPO\_DOCUMENT\_REFERENCES\_DDL.sql | $XXNZCL\_TOP/install/sql |
| Database Package | XXPO\_PURCHASE\_ORDER\_PKG.pks | $XXNZCL\_TOP/install/sql |
| Database Package | XXPO\_PURCHASE\_ORDER\_PKG.pkb | $XXNZCL\_TOP/install/sql |
| Loader | XXPOREQGRP.ldt | $XXNZCL\_TOP/import |
| Loader | XXPOIMPTM.ldt | $XXNZCL\_TOP/import |
| Loader | XXPOIMPTBAS.ldt | $XXNZCL\_TOP/import |
| Install Script | PO.02.11\_install.sh | $XXNZCL\_TOP/install |

## Testing Summary

The customizations were tested in 2 Degrees' snzclakl025.nzc.co.nz TEST environment before promoting to the next target environment. See the related functional test scripts for more information.

# Open and Closed Document Issues

## Open Issues

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Issue | Resolution | Responsibility | Target Date | Impact Date |
| 1. |  |  |  |  |  |
| 2. |  |  |  |  |  |

## Closed Issues

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| ID | Issue | Resolution | Responsibility | Target Date | Impact Date |
| 1. |  |  |  |  |  |
| 2. |  |  |  |  |  |
| 3. |  |  |  |  |  |
| 4. |  |  |  |  |  |