# Contents

# Contents

# Contents

# 1 Assist Help Config in Other Systems

Assist Help Configuration in Other Systems

## 1.1 CTMS

Access to the database is required.

```
SELECT * FROM ADM_SYSTEM_PARAM
WHERE PARAM_NAME LIKE '%ASSIST%'
```

Update parameter CALIDUS_ASSIST_BASE_URL:

- If set to "/calidus-assist/MTS/index.php" then uses the base URL of the system you're accessing as the start.
- You can set the whole thing with something like: "https://calidusassist.adcservices.apteancloud.com/calidus-assist/MTS/index.php"

```
UPDATE ADM_SYSTEM_PARAM
SET VALUE = 'https://calidusassist.adcservices.apteancloud.com/calidus-assist/MTS/index.php'
WHERE PARAM_NAME = 'CALIDUS_ASSIST_BASE_URL'
```

## 1.2 EPOD

Log on to the APP machine

Find the appropriate APP directory e.g. "C:\Program Files (x86)\Calidus.epod\EPOD_CTYPRD.app"

Edit the file "web.config" in Administrator mode

Find key "AssistURL" and set to the appropriate Assist system e.g.

```
<add key="AssistURL" value="https://calidusassist.adcservices.apteancloud.com/calidus-assist/EPOD/index
```

Save.

## 1.3 CTL

Log on to the APP machine

Find the appropriate APP directory e.g. "C:\Program Files (x86)\Calidus.CTL\CTYPRD"

Edit the file "web.config" in Administrator mode

Find key "HelpDocument" and set to the appropriate Assist system e.g.

```
<add key="HelpDocument" value="https://calidusassist.adcservices.apteancloud.com/calidus-assist/CTL
```

Save.

## 1.4 WMS

Log on to the system.

Go to Setup/System Registry

Go to Global/General Settings.

Set WikiURL to https://calidusassist.adcservices.apteancloud.com/calidus-assist/





Set WikiSubPath to WMS/index.php/Main_Page

Save.

**Warning**: This is not confirmed as working.

**Note:** This is not accessible from the Help menu Screen Help and Contents and Index menu items, but solely from Help/About, then clicking the Calidus Wiki button.

This functionality (to access context sensitive help) has been requested from the R&D team - this guide should be updated if/when this is complete.

## 1.5 Bay Diary

Log on to the APP machine

Find the appropriate APP directory.

Edit the file "web.config" in Administrator mode

Find (or add) key "AssistURL" and set to the appropriate Assist system e.g.

```
<add key="AssistURL" value="https://calidusassist.adcservices.apteancloud.com/calidus-assist/WMS/index.
```

Save.

## 1.6 TCM

**Warning**: This is an incomplete guide.

## 1.7 TOC

**Warning**: This is an incomplete guide.

## 1.8 Portal

**Warning**: This is an incomplete guide.



# 2 Bay Diary

This page is intended to hold technical information regarding Bay Diary for WMS.

## 2.1 Packages

DP_NET_PORTAL - main package

DP_MSG_EMAIL - sending of email from Bay Diary

## 2.2 Tables

**Note:** The majority of BAY_DIARY_* files are not in use.

### 2.2.1 CONFIGURATION

Field Config Master

```
select * from bay_diary_config t
```

Field Config per User

```
select * from bay_diary_config_users t
```

The bays per warehouse

```
select * from bay_headers t
```

### 2.2.2 OPERATIONAL

| Table | Notes |
|---|---|
| ware_diary_details | |
| ware_diary_headers | |
| ware_diary_slots | |

SQL:

```
select *
from ware_diary_details t
where company_code = 'POD'
and warehouse_id = 'RCT'
order by delivery_receipt_date desc

select *
from ware_diary_headers t
where company_code = 'JP1'
and warehouse_id = 'NFT'
order by delivery_receipt_date desc

select *
from ware_diary_slots t

Select * from CARRIERS

Select * from STK_STOCKIST
```

# 3 BI Data Extract

This page is intended to describe the configuration of the MIS/BI Extracts from a technical perspective.

This process has already been documented from a user perspective in ctms:BI_Data_Extract_Guide

## 3.1 Configuration

The files that will be created are defined in table mis_extract_header

Only active files are generated.

| COLUMN_NAME | DATA_LENGTH | NOTES |
|---|---|---|
| EXTRACT_FILENAME | 120 | Less than 25 characters only |
| EXTRACT_FILENAME_EXTENSION | 40 | csv |
| TABLE_NAME | 120 | The table being extracted. This must be unique. For more complex queries with multiple tables, consider using a view name instead.<br><br>Where the extract requires another table linked to selected the data (for example, SCH_ORD_REFERENCE), these may be listed in the table names and linked in the here clause, as long as they do not exceed 120 characters. For example table name "`sch_ord_reference, sch_ord`" and where clause "`WHERE sch_ord_reference.oms_ref = sch_ord.oms_ref AND ...`". Ensure that the table names match EXACTLY on the detail record, and be wary that field names will then need to be explicitly identified by the table you want them from.<br><br>Alternatively, use a sub-query for the select e.g. `where oms_ref in (select oms_ref ....` |
| DELIMITER | 40 | The delimiter character. This must be enclosed in PL/SQL syntax to concatenate the values. For example, use `'||'®'||'` for a unique separator character. Not relevant if using tab delimiting, as the package will instead use CHR (9). |
| EXTRACT_OUTPUT_DIRECTORY | 200 | Usually `/webint/dbname/export` or `/webint/dbname/MIS` |
| DATABASE_SOURCE | 120 | dbname |
| WHERE_CLAUSE | 4000 | Where clause. See note below. |
| RUN_FREQUENCY | 80 | EVERY_DAY |
| ACTIVE_FLAG | 4 | Y |
| TABLE_TYPE | 4 | N/A |

**Note:**

- For Where Clause:
  - Standing data tables should be always be exported. So clause should be "`WHERE 1=1`"
  - Transactional data should export any data created or updated in the last calendar day or days (depending on customer requirements). So "`WHERE ((created_date BETWEEN SYSDATE-4 AND SYSDATE) OR (updated_date BETWEEN SYSDATE-4 AND SYSDATE))`". Date column names are dependent on the table (or tables) being selected.

The columns (up to a maximum of 50) are defined in table mis_extract_detail.

| COLUMN_NAME | DATA_LENGTH | NOTES |
|---|---|---|
| TABLE_NAME | 120 | Parent key TABLE_NAME from above. |
| COLUMN_NAME | 120 | The column from the table |
| DATA_TYPE_SIZE | 120 | the data type and length e.g. `VARCHAR2(40), NUMBER(5), DATE` |
| COLUMN_POSITION | 22 | The position of this column in the extract file |
| FORMATING_REQUIREMENT | 240 | Usually used for dates e.g. `TO_CHAR(START_TIME,'RRRR-MM-DD HH24:MI')` |

**Note:**

- Up to 50 columns only.
- If more columns are required, define another extract for the additional columns.
- Note that the table name must be unique, so if multiple extracts are required from the same table, this would need to be aliased e.g. "`SCH_TRIP as st2`". Note that this will need to be less than 120 characters in length. **Warning**: This may not work.

Sample SQL

```
-- What denotes the files to be created
select * from mis_extract_header
WHERE active_flag = 'Y'
FOR UPDATE

select mih.extract_filename || '.' || mih.extract_filename_extension "File_Name"
, Table_Name
, Case WHERE_CLAUSE WHEN ' WHERE 1=1' THEN 'All Data' ELSE 'Daily Data' END "Contains"
from mis_extract_header mih
WHERE active_flag = 'Y'

select * from mis_extract_header
FOR UPDATE

select * from mis_extract_header
WHERE WHERE_CLAUSE NOT LIKE '%1=1%'
AND ACTIVE_FLAG = 'Y'

-- Set active the ones that you want
UPDATE mis_extract_header
SET active_flag = 'N'
WHERE extract_filename <> 'Z_RES_CARRIER_TYPE'

-- Set the output directory and database source
UPDATE mis_extract_header
SET EXTRACT_OUTPUT_DIRECTORY = '/webint/stsstst/export',
DATABASE_SOURCE = 'stsstst';

-- Set up the fields being exported
select * from mis_extract_detail
where table_name like 'SCH_ORD_ITEMS%'
order by table_name, column_position

select table_name, column_name, data_type_size, column_position
from mis_extract_detail
where table_name like 'SCH_ORD_ITEMS%'
order by table_name, column_position
```

## 3.2 Parameters

System parameters control the extract.

- MIS_CHAR_SET - WE8ISO8859P1
- MIS_DELETED_DAYS - 30
- MIS_DELETE_REQUIRED - Y
- MIS_FTP_DESTINATION_DIRECTORY - if a subdirectory is required, define it here.
- MIS_FTP_DESTINATION_IP_ADDRESS - an IP address or URLL for the FTP or SFTP
- MIS_FTP_DESTINATION_PASSWORD - the password for the FTP/SFTP server
- MIS_FTP_DESTINATION_PORT - the port, 22 for SFTP, 23 for FTP
- MIS_FTP_DESTINATION_USERNAME - the username for the FTP/SFTP server
- MIS_FTP_PROTOCOL - FTP or SFTP
- MIS_TAB_DELIMITER - Y if you want the files generated to be tab delimited.

If using SFTP, define also the level of logging:

- SFTP_LOG_LEVEL - 0 to 3.

Sample SQL:



```
-- Set up MIS parameters
SELECT * FROM ADM_SYSTEM_PARAM
WHERE PARAM_NAME LIKE 'MIS%'
FOR UPDATE


SELECT * FROM ADM_SYSTEM_PARAM
WHERE PARAM_NAME LIKE 'SFTP%'
FOR UPDATE
```

## 3.3 Directory access

First, create directory from PLSQL Developer, from Directories object/create new

Usually `/webint/dbname/export` or `/webint/dbname/MIS`

Then grant permissions for MTS_USER to access (EDI user)

For FTP/SFTP destinations you will need to set up the fingerprint

Sample SQL:

```
-- grant permission for MTS_USER to access (EDI user)
GRANT READ,WRITE ON DIRECTORY EXPORT TO MTS_USER;

BEGIN
  dp_sftp.open_connection( i_host => 'the host', i_trust_server => true );
  dp_sftp.close_connection;
END;

-- See the fingerprint
select * from sftp_known_hosts
```

## 3.4 Packages

The package to send MIS/BI data is DP_MIS

Sample SQL:

```
select DP_MIS.GET_PACKAGE_VERSION from dual
select SYSDATE FROM DUAL
```

## 3.5 Auditing and Troubleshooting

Sample SQL:

```
-- Audit for SFTP
select * from adm_log
--WHERE PROG_NAME IN ('DP_SFTP')
where date_created <= TO_DATE('2025-10-31 07:20:00', 'YYYY-MM-DD HH24:MI:SS')
AND PROG_NAME NOT IN ('TRM', 'PAR')
-- where err_type <> 'DEBUG' AND STMT < 2255004935 -- PROG_NAME IN ('DP_SFTP')
order by stmt desc
order by adm_log.date_created desc

-- Audit log from the MIS package
select * from   Mis_extract_run_detail
order by last_ran desc

-- Checking job
Select * from DBA_JOBS
where UPPER(what) like '%DAILY_EXTRACTS%'

-- When did it last run?
-- Check the last run date/time on the jobs table to the sysdate
select sysdate from dual;
```



```
-- Run the job now
DECLARE
g_process_name edi_process_header.process_name%TYPE;
BEGIN
g_process_name := 'DAILY_EXTRACTS';
DP_REPORTS.P_RUN_PROCESS(g_process_name);
COMMIT;
END;
-- Or just find the job number, and then use PLSQL to run the job from the Jobs list to the left.

-- Jobs not running? Check the following:
select logins from v$instance; -- if not allowed, won't run jobs
select value from dba_scheduler_global_attribute where attribute_name='SCHEDULER_DISABLED' -- if TRUE won't
select value from v$parameter where name='job_queue_processes'; -- if 0 won't run jobs (most common
alter system set job_queue_processes=20; -- Fix to above issue
```





# 4 Browserless Application Creation

**Note:** **For internal use only**

## 4.1 Introduction

All of the code to generate Browserless Oracle installers is located here: Browserless Oracle on Sharepoint

## 4.2 Folder Contents

**Note:** Folders in red below are working directories and should **not** be accessed/maintained when creating builds.

- **Builds**

  This directory contains a temporary folder for each client/version built and is generated during the browserless oracle build for each client

- **Config**

  This directory contains batch files which are used for configuring each individual client. See the Creating an installer package for details.

- **Documents**

  Documentation relating to the process

- **Installers**

  This directory contains the built executable installers for release to the client. there will be a folder for each client/version built

- **Java**

  This directory contains the java versions which will be build into the installer packages

- **Resource**

  This directory contains the scripts, sources and configuration files used in the build

- **Setup**

  This directory contains the software which must be installed on the users PC prior to running the installer creation process.
  **Note:** These are for CREATING installer packages, and are NOT required on the final users machines.

## 4.3 Creating an Installer package

- Pre-Requisites

  InnoSetup and Launch4J must be installed before an installer can be created. Run the installers from the Setup directory and accept all defaults during the process.

- Creation process

The batch script to create an installer is in the main Browserless Oracle directory:

```
createNewBuild.bat
```

Before this is executed, a Config script is required. These are stored in the Config directory.





The scripts should be named by client. To create a new script for a new client, copy an existing one, rename and then amend it.

A client script should look like this:

```
set WMSTMS=
set BUILD=
set PREFIX=
set PROD=
set TEST=
set QA=
set JAVA_VER=
REM **** USE _GENERATE GUI.bat to generate a unique GUID for each client - ONLY DO THIS ONCE PER CLIENT!!!
set GUID=
```

- WMSTMS - Should be either TMS or WMS dependent on the system
- BUILD - the build version of the installer e.g. 1.0
- PREFIX - the clients "short" code (usually the first part of the database name before tst or prd) e.g. psdt, lfst, schw etc.
- PROD - the Production URL for the client
- TEST - the Production URL for the client
- QA - the Production URL for the client. Leave blank (but do not remove the line) if no QA/UAT system exists.
- GUID - a unique identifier for the client which is used to identify the install in the registry.

  If this is a new client script, run _**GENERATE GUI.bat** to generate a new GUID:

  ```
  Generated GUI:
  bcc4b78c-11c7-4bf0-840d-275b5bc729be
  This has been pasted into the clipboard - Paste this into the "<client>".bat file on the line GUID=xxx
  Press any key to continue . . .
  ```

  Paste this value onto the GUID line

  **Note:** Once generated, these GUID values should not be changed for the client

Once the client script has been created, the **createNewBuild.bat** can be executed.

This will prompt for a client ID, which should match the name of the client script.

Once executed, the process will generate all of the files needed for the installer, sign them, and package them into an exe.

Example:

```
Client: Polar Speed
Polar Speed Browserless Oracle Build 15/02/2024 11:35:45.14
Create calidus_info.txt
launch4j
signtool - EXE
Inno Setup USER
Inno Setup ADMIN
signtool - Installer
Press any key to continue . . .
```

The above run would create 2 installers (User/Admin) in the Installers directory within Aptean Calidus-TMS or WMS and then a client/version directory:

```
Aptean Calidus-TMS-Polar Speed_ADMIN.exe
Aptean Calidus-TMS-Polar Speed_USER.exe
```

These are the signed executable installers which should be issued to the client. **Note: These installers should be tested on your own PC before issue to the client**

## 4.4 Additional Technical Details

- **Java**





New Java JRE packages should be downloaded from the Eclipse Temurin site and placed in the Java directory as jdk-xxx
**Note:** This may change if we start using Oracle Java packages.

Once this is done, you should edit **createJavaJRE.bat** and change the set JAVA=xxx to match the folder above.

Run **createJavaJRE.bat** to create a java-xxx directory which has a cut down version of Java with only the required files to run browserless oracle.

The java-xxx should then be added to the JAVA_VER parameter inside the required Config script for the client.

- **frmsal**

The frmsal jar files are located in Resource\frmsal

The build process will try to locate a frmsal file which matches the exe being produced e.g. psdttst.frmsal.jar. If it finds a specific frmsal for the build it will use this, and if not, it will use the generic frmsal.jar

- **Signing Certificate**

The certificate is located in Resource\Aptean. If updated, the following lines will need amending in Resouce\core.bat:

```
set CERT=..\..\Resource\Aptean\Aptean_Inc.pfx
set CERTPASS=A23cDez37Fx
set CEREX=29/03/2025
```

- **Additional Info File**

The file \Resource\calidus_info_generic.txt will be pulled into the calidus_info.txt file built into the installer exe.

Any **non-client** specific info which is required can be added to this file.



# 5 CALIDUS Portal TTM Interface

The setup of the interface has been documented for customers here: ctms:CALIDUS PORTAL TTM Interface and is included below by reference.

This provides a more technical guide

# 6 Portal TTM Interface Functional Guide

Customer - must have LOTS Enabled set to TTM Customer, plus tick boxes for at least the ORD message.

System Parameters must be configured

| PARAM_NAME | DESCRIPTION | CONFIG_BY |
|---|---|---|
| LOTS_ADD_DEBRIEF | LOTS Additional Debrief Details | COST_CENTRE |
| LOTS_ALWAYS_SEND_HDR_CONTACTS | Always send header contacts to LOTS in addition to order contacts | COST_CENTRE |
| LOTS_DEL_CUST_SIGNATORY | Controls if DEL messages for LOTS require an actual signatory - Y to hold messages. | COST_CENTRE |
| LOTS_DEL_MSG_TRIP_STATUS_COMPLETE | Send LOTS DEL Message if trip status set to COMPLETE (Y/N) | COST_CENTRE |
| LOTS_EXPORT_TYRE | LOTS export tyre | COST_CENTRE |
| *LOTS_FTP_DESTINATION_DIRECTORY | Final Destination directory for FTP of LOTS Files | SYSTEM |
| *LOTS_FTP_DESTINATION_IP_ADDRESS | IP for FTP of LOTS Files | SYSTEM |
| *LOTS_FTP_DESTINATION_PASSWORD | Password for FTP of LOTS Files | SYSTEM |
| *LOTS_FTP_DESTINATION_PORT | Port for FTP of LOTS Files | SYSTEM |
| *LOTS_FTP_DESTINATION_USERNAME | Username for FTP of LOTS Files | SYSTEM |
| LOTS_FTP_PUT_DIRECTORY | Mid-Send directory for FTP of LOTS Files | SYSTEM |
| LOTS_INBOUND_ARCH | LOTS inbound failures directory | SYSTEM |
| LOTS_INBOUND_FAIL | Pattern for LOTS Trip Inbound XML | SYSTEM |
| LOTS_INBOUND_IDENTIFIER | Filename for list of files in directory for LOTS XML OB1 Inbound | SYSTEM |
| LOTS_INBOUND_LISTING_NAME | LOTS inbound directory | SYSTEM |
| LOTS_INBOUND_PATH | Script name foe LOTS XML OB1 Inbound | SYSTEM |
| LOTS_INC_ORDER_HEADER_CONFIRM | Includes the ORDER_HEADER_CONFIRM section in the DEL file for LOTS. | SYSTEM |
| LOTS_INC_STOP_SIGNATURE | Indicates if the STOP_SIGNATURE section will be displayed in the DEL outbound files to LOTS. | SYSTEM |
| LOTS_INC_STOP_TYPE_DESC | Include STOP-TYPE_DESC tag in XML | SYSTEM |
| LOTS_LISTING_SCRIPT_NAME | LOTS outbound archive directory | SYSTEM |
| *LOTS_OUTBOUND_ARCH | LOTS inbound failures directory | SYSTEM |
| *LOTS_OUTBOUND_FAIL | LOTS outbound directory | SYSTEM |
| *LOTS_OUTBOUND_PATH | Maximum number of days a location can remain inactive before being set to inactive automatically. | SYSTEM |
| LOTS_PREVENT_MULTI_VLI_VUI | Prevent multiple VLI and VUI files being sent to LOTS (Y/N). | SYSTEM |
| LOTS_SEND_PLANNED | Allow TRP update message to be sent to LOTS when trip is still at status Planned | SYSTEM |
| LOTS_SEND_WHEN_VEHICLE_SCANNED | List of LOTS message types that will be held until a vehicle scan is received for the order. | COST_CENTRE |
| LOTS_VERSION_NUMBER | LOTS Poller Version | SYSTEM |
| *LOTS_XFER_EXTENSION | Sets the LOTS transfer file extension | COST_CENTRE |
| ORD_XDOCK_LOTS_DEL_MSG | Lots DEL msg is only sent on last trip for xdock orders (Y/N) | SYSTEM |
| PAR_USE_SLOTS | Use slots when exporting to Paragon (DUN format) | SYSTEM |
| *SEND_LINE_ITEM_TO_LOTS | Send Line Item To LOTS (Y/N) | SYSTEM |
| SET_MICROLISE_SLOTS | Determines if the delivery slot times are sent to Microlise instead of the planned delivery times. | SYSTEM |

Any issues should be forwarded to your Aptean support team.





# 7 Technical Information

The Job must be enabled

```
Select * from DBA_JOBS
where UPPER(what) like '%LOTS%'

        BEGIN
        INT_XML_OUT2.PROCESS_XML_OUTBOUND_LOTS_ORD;

        INT_XML_OUT2.PROCESS_XML_OUTBOUND_LOTS;
        END;
```

Possible issues

```
-- Jobs not running? Check the following:
select logins from v$instance; -- if not allowed, won't run jobs
select value from dba_scheduler_global_attribute where attribute_name='SCHEDULER_DISABLED' -- if TRUE won't
select value from v$parameter where name='job_queue_processes'; -- if 0 won't run jobs (most common
alter system set job_queue_processes=20; -- Fix to above issue
```

Messages are written to INT_XML_CONTROL, type LOTS, processed "N" or "P".

Any issues with FTP to the box will be logged to ADM_LOG

```
select * from adm_log
where prog_name like '%INT_XML%'
and err_type = 'ERROR'
-- ORI_USER = 'MTS_OWNER'
order by stmt desc
```





# 8 Checking EDI File Folders

This page will document how to check EDI file production and other related topics. CTMS will be used as an example, but this is relevant to any Oracle system.

## 8.1 Requirements

- Access to the systems
- Access to the database
- TNS Names
- FileZilla or other FTP client.
- VPN.
- For some systems, access to the jump box.

## 8.2 Process

Check the EDI for the destination folder.

- For EDI processes, check ctms:EDI Maintenance, which will tell you the destination folder.
- For BI/MIS, check the table mis_extract_header for details see BI Data Extract.

Get the details of the Oracle server from your TNS Names, usually located in `C:\oracle\app\product\11.2.0\client_1\network\admin\tnsnames.ora`. Note that version and client name can vary, depending on how it is installed on your machine.

In FileZilla:

- Create a new connection in connection manager.
- Set a meaningful name - suggested using the database name i.e. STSTPRD.
- Set the IP address to the IP address of the TNS Name
- Set the username and password to the `oracle` or your provided username and password. Consult your team for details.
- Connect.
- In the right explorer, paste in the destination directory.
- This will show the produced files.





# 9 Creating XSD Documentation

The purpose of this document is to show how XSDs are properly documented and the results stored in shared areas and source control for all to use. These files are required by customers to aid in developing interfaces into CTMS and as such are reuired parts of the development process.

## 9.1 Requirements

- An XSD to document, with valid annotation tags embedded within it.
- Notepad++ with XML Tools plugin installed.
- Chrome (or other HTML to PDF conversion tool).
- Access to the shared drives.
- Access to the XSL transformation sheet xs3p.xsl, in the shared drives.

## 9.2 Process

Once the XSD has been modified, ensure that it is in source code control.

Save the XSD to the shared drive, for exmple for TripOrder XSD, \\DGA1FS01OBS\Projects3\Product\OBS XML\C-TMS tripOrder\Schema Source

- Using Notepad++, open the XSD you have edited.
- Menu, *Plugins/XML Tools/XSL Transformation*.
- Select the XSL transformation sheet xs3p.xsl.
- Click **Transform**.
- On the newly created file, replace the titles with the version. For example, for TripOrderXML, replace all "Trip/Order XML" with "Trip/Order XML vx.yy". There should be 3.
- Click **Save**.
- Change type to HTML.
- Name as "{InterfaceName} vx.yy XML Schema Documentation.html", for example "TripOrder vx.yy XML Schema Documentation.html".
- Open the HTML file in Chrome (directly, or double click the created HTML file, or from Notepad++, Menu, *Run/Open in Chrome*).
- Right-click, *Print*
- Change Destination to *Save as PDF*
- Click **Save**
- Name as "{InterfaceName} vx.yy XML Schema Documentation.pdf", for example "TripOrder vx.yy XML Schema Documentation.pdf".
- Copy the following files to the shared drive:
    - {InterfaceName} vx.yy XML Schema Documentation.pdf
    - {InterfaceName} vx.yy XML Schema Documentation.html





# 10 CTL Customer Services Solution Overview

This is a technical overview of the CTL project as used for customer services and fleet management.

## 10.1 Projects

- CTL-TMS - UI - webforms, JQuery, etc
- TMSCustomServerControls - UI - reusable standardised controls
- TMSDatabaseClassLibrary - Database access - DAL, BAL, SQL, models
- TMSDatabaseScriptLibrary
- TMSResourceClassLibrary - Multi-lingual, labels, messages, etc.
- TMSUtilityClassLibrary

### 10.1.1 CTL-TMS

The main UI.

- App_Data - a list of text entries that should not be allowed. Security, prevents trojan attacks.
- App_Start - Project folder
- bin - included .NET objects. DLLs generally.
- Content - reusable aspx/aspx.cs and JS content. Think include files for commonly used things.
    - Internal - reusable items that we have written. reusable aspx/aspx.cs. Lots are to do with CTL only, but some are super-important - called out below. Usually included in teh Master page that the webform page is based on.
        - Toolbar - used to define the toolbar
        - AuditTrailGV - CTL - N/A
        - Calender - CTL - N/A
        - Clock - CTL - N/A
        - ContactsMaint
        - CustomerService specific to customer services screen
        - CustomServerControls
        - Footer - generic footer for every screen.
        - HereMap
        - LeftSlider - left pop-out pane
        - LocationAssociation - CTL - N/A
        - LookupList
        - MenuSlider
        - NoteMaint
        - OverheadDataFeed
        - ParameterDetail - CTL - N/A
        - ReportHub - specific to report hub
        - Requirements
        - RightSlider - right pop-out pane
- Controllersempty - usually used for MVC forms. There are none in the customer services CTL at the moment, several (but not many) in CTL project.
- DataFiles - a list of text entries that should not be allowed. Security, prevents trojan attacks.
- gateway - unsure - think might be to do with Customer Servies Screen processing.
- Images - images used by CSS and webforms. E.g. the Aptean logo.
- Models - empty - usually used for MVC forms. There are none in the customer services CTL at the moment, several (but not many) in CTL project.
- obj - built objects
- Properties - project properties
- Scripts - Javascript included in webforms - reusable browser-based code, commonly written by someone else. E.g. all the JQuery stuff is in here.
- Service References
- Skins - JQueryUI schemes, CSS
- WebForm - Where the forms front-end (.aspx) and back-end/server (.aspx.cs) code exists
- WebServices - CTL - N/A

In the top level are also the master forms. think of these as the basic template of every form that references it.

- AdminSite.Master - most screens inherit this
- Basic.Master
- Blank.Master





- EndGateway.Master
- MainOnly.Master
- Planning.Master
- Site.Mobile.Master - doesn't seem used.

There are also some basic HTTP Error forms that ae defined here:

- Errors.htm
- InvalidAccess.htm
- LoggedOut.htm
- NotAuthorised.htm
- SessionExpired.htm

Theere are also some others:

- Default.aspx - a default form for the project - N/A
- ViewSwitcher.ascx - something to do with friendly URLs and mobile views. Doesn't seem used by anything except the site mobile master, which itself seems unused.

### 10.1.2 TMSCustomServerControls

UI - reusable standardised controls

- ButtonControls - defines classes for all common buttons that might be used in the application, along with the styles that are applicable to the button in that area
    - FA - Slider Menus
    - AT - Table buttons
    - FT - Footer buttons
    - Others
- CheckBoxControls
- DropDownListControls - defines the drop-down list PLUS the data table access required to populate that drop-down list. There are dozens of these.
- GridViewControls - controls related to the main gridview
- HyperLinkControls - don't appear to be used in this project
- LabelControls - used everywhere for labels of any kind, in grid views, accordions, etc.
- LiternalControls - not used
- PageControls - reusable button click events, attached to control events, e.g. toolbar buttons
- RadioButtonControls
- TextBoxControls - All types of data entry, from generic reusable types to specific field types. You can define lookups in here and it will add the buttons for you. The lookups are defined in the LookupBAL/DAL objects. Assume that every unique field will have a custom TextBox control defined here. Most will simply reuse a core class such as
    - ..General
    - ..Filter
    - ..Alpha
    - ..Website
    - ..Numeric
    - ..Integer
    - ..Percentage
    - ..Decimal
    - ..etc
    - TextBoxProductCode builds on TextBoxAlpha, which itself builds on TextBoxGeneral, which builds on TextBox, which is the .NET basic text box control.

### 10.1.3 TMSDatabaseClassLibrary

- BAL = Business Access Layer = Business Logic
- DAL = Data Access Layer = the core components to select, update,insert on table(s) in this DAL object, including all variants.

Naming convention varies, but generally after the table or view name (and in a lot of cases, after the CTL table name).

Typically every table will need a

- Model





- A simple, usually generated
- A more complex one listing the fields and any properties and types.

- DAL
  - Again, in pairs, this defines the selects required, inserts, updates, etc.
- BAL
  - Some objects require BALs. This typically collects properties nd methods relating to this data object above and beyond the simple SQL functions. For example, the BAL may contain static lists or classes that define the data that we would use, or functions relating to the object, like calculating collection and delibery windows for an order.
  - Predominantly, BAL business logic functionality will not exist - this will remain in the database, but there may be functions to CALL these BL procedures from within .NET within the BAL.

### 10.1.4 TMSDatabaseScriptLibrary

Database Change scripts and customer configuration scripts. part of the release process. Probably superseded by Carl's CTMS release process.

### 10.1.5 TMSResourceClassLibrary

Multi-lingual, labels, messages, etc.

- AppMessages - messages
- AppTerms - labels

### 10.1.6 TMSUtilityClassLibrary

Lots of really general (and some really specific) utility classes. Useful for coding/reusability.

- WebService.cs
- Exceptions
- obj
- Properties
- Resources
- Callback.cs
- Cookies.cs
- CustomEvents.cs - HTML Error event logging - related to hosting log files
- DataProcess.cs - specific to CTL Import process - N/A
- Exception_Handler.cs - handing exceptions - seems to use the Exceptions filder, and seems to be specific to CTL
- General.cs - coding utility classes. Like how to generate a random numner.
- MCSUtilities.cs - TimeZone functions. Not used anywhere.
- Message.cs - functions to take standard messages (in the Resource classes) and insert values at placeholder positions. Includes some pre/suffixes to help identify what type of message it is and therefore the formatting. Does not seem used.
- MessageDuration.cs - used when popping up messages (Content/Internal/Toolbar)
- Security.cs - used mostly in adm01_Login
- Validation.cs - common validations. Used in CustomServerControls like TextBox, GridView, etc.





# 11 CTMS Paragon Interface

The basic Paragon interface has been documented here: ctms:Paragon Interface and is included below by reference

This guide provides more technical information





# 12 Paragon Functional Guide

**Note:** This guide covers the direct Paragon API interface. There are other more manual interfaces to Aptean Routing & Scheduling - Paragon edition, but these are not covered here.

There are 2 types of direct Paragon APIs:

- Strategic - Fixed Drop Scheduling Engine
- Tactical - creating and optimising routes before or on the day of execution

These can be configured separately, so that either or both can be in use.

## 12.1 Contents



## 12.2 Configuration

### 12.2.1 System Parameters

System parameters enable the functionality of the Paragon API.

| Name | Description | Usage |
|---|---|---|
| PAR_TENNANT_KEY1 | Tennant KEY 1 for paragon API | SYSTEM |
| PAR_ENDPOINT_URL | URL for paragon API | SYSTEM |
| PAR_TENNANT | Tennant for paragon API | SYSTEM |
| PAR_FREQUENCY | Frequency for PAR Master keys | SYSTEM |
| PAR_START_DATE | Start Date for Master Keys | SYSTEM |
| PAR_KEY_FORMAT | PAR Master key format WKXX, DDMM, DAYX | SYSTEM |
| TK PAR_USE_RPOXY | Paragon Use Proxy | SYSTEM |
| TK PAR_PROXY | Paragon Proxy Server | SYSTEM |
| PAR_GROUP_STAGING | Paragon Group Staging Level | SYSTEM |
| PAR_RUN_NUMBER | Is Paragon Planning based on Run Numbers? | SYSTEM |
| PAR_KEY_PROJECT | PAR Project name | SYSTEM |
| PAR_API | Create Control records for Paragon API | SYSTEM |
| PAR_SEND_ALL_LOCATIONS | Are locations sent out via API ('TACTICAL','STRATEGIC','BOTH') | SYSTEM |
| PAR_AUDIT | Include auditing of the import process in the STP version of the Paragon API (Y/N) | SYSTEM |
| HTTPS_WALLET_FILE | | |
| HTTPS_PASSWORD | | |
| AUTO_SCHED_INACTIVE_DEPOTS | | |
| TRM_RETAIN_EMPTY_STOPS | | |

A full list of configurable parameters is available here:

- System Parameters List

### 12.2.2 Order and Location Details sent to Paragon





The content of each message sent to Paragon is controlled through internal configuration tables. These are maintained and configured by your Aptean implementation team.

These allow configuration of the various elements that are sent from CTMS to Aptean Routing and Scheduling - Paragon Edition.

Orders

- Any direct field from tables:
  - SCH_ORD - the order.
  - SCH_ORDER_LINE - the deliverable types such as Parcels, Tyres, etc.
  - SCH_ORD_ITEMS - the individual parcels, or quantity of each specific product.
  - GEO_LOCATION GEO_TO - details of the final destination.
  - GEO_LOCATION GEO_FROM - details of the origin.
- Functions can be called for other information:
  - DP_PAR_API_STP.GET_REF - retrieve any reference against the order.
  - DP_PAR_API_STP.GET_ORDER_TYPE - retrieve the order type.
  - DP_PAR_API_STP.GET_FROM_LOC - summarised details of the origin.
  - DP_PAR_API_STP.GET_TO_LOC - summarised details of the destination.
  - DP_PAR_API_STP.OPENING_TIMES - opening times of the destination.
  - DP_PAR_API_STP.CLOSING_TIMES - closing times of the destination.
  - DP_PAR_API_STP.GET_DROP_NUMBER - the specific drop number.
  - DP_PAR_API_STP.GET_TOTAL_TYRES - the total tyres (specific to tyre delivery - use the below function for more generic systems)
  - DP_PAR_API_STP.GET_QTY_BY_DU - the total quantity of a specific deliverable unit, e.g. pallets, parcels, tyres, etc.

Locations:

- Any direct field from the following tables:
  - GEO_LOCATION - details of the supplied location.

### 12.2.3 Run Key Configuration

Aptean Routing and Scheduling - Paragon Edition controls all planning through Runs. Runs are normally associated to a schedule within CTMS, but not always. In this case, there is a Run configuration that aligns the dates of jobs within CTMS to the appropriate Paragon run key.

This is achieved through the Business Data Maintenance screen, on the *Paragon Keys* tab.

Note that the enabled in Access Control, accessible tabs, for screen "BDM" tab "PAR_KEYS".



A full list of configurable tabs and functions is available here:

- Access Control - Accessible Functionality

### 12.2.4 Turning on the interface

The individual processes for Paragon are controlled through EDI Process Configuration in the EDI Maintenance screen.

**Inbound**

Inbound processes are split into 3

- Inbound Tactical Receive per depot
- Inbound Strategic Receive per depot
- Inbound Processing for all staged receipts above

Regardless of the components of the interface that are in use, the latter process must always be running.

**Inbound Strategic Route**

These processes get the information from Paragon and stage the information on inbound tables, ready for import

This can be configured for all depots or one per depot, depending on how Paragon is configured. For example, if Paragon is configured with different plans per regional depot, then each import process should be configured separately for each depot here. Therefore this should be named appropriately e.g. the name of the depot.

Process: DP_PAR_API_STP.get_paragon_route

Parameters

- DEPOT_KEY - the RDC Location ID

Report Values

- Package PROCESS DP_PAR_API_STP.get_paragon_route
- Process p_process_name the name of the EDI process that has been configured above.

**Inbound Tactical**

These processes get the information from Paragon and stage the information on inbound tables, ready for import

This can be configured for all depots or one per depot, depending on how Paragon is configured. For example, if Paragon is configured with different plans per regional depot, then each import process should be configured separately for each depot here. Therefore this should be named appropriately e.g. the name of the depot.

Process: DP_PAR_AP_STPI.get_paragon

Parameters

- DEPOT - the EDI Process Name

Report Values

- Package PROCESS DP_PAR_API_STP.get_paragon
- Process p_process_name Get_Paragon

**Inbound Processing**

This is the general inbound processing job.



This process processes the information from the inbound tables into the CTMS database.

- Name: paragon_in
- PROCESS: DP_PAR_API_STP.READ_PARAGON_IN

Report Values

- Package PROCESS DP_PAR_API_STP.READ_PARAGON_IN
- Process p_process_name paragon_in

**Outbound**

- Name: Paragon_Outbound
- Process: DP_PAR_API_STP.process_paragon

Parameters

- AUDIT_WS Y/N
- USE_RUN_DEPOT Y/N

Report Values

- Package PROCESS DP_PAR_API_STP.process_paragon_outbound
- Process p_process_name Paragon_Outbound

## 12.3 Strategic Interface

This interface allows definition of locations onto fixed routes at specific drop numbers.

**Note:** This is applicable to Fixed Drop Scheduling engine only.

When imported, this deletes any previous configuration against locations and replaces it with the new network map.

**Note:** Bank Holiday routes will NOT be deleted - these are expected to be managed manually in CTMS.

The data that is sent is configurable, as seen in the sections above.

## 12.4 Tactical

When orders are received into CTMS, they may be planned on temporary trips using the scheduling engine - these trips should be configured to be prefixed with "TMP" so that they can be easily distinguished.

Orders and Locations are sent to Paragon for planning.

The data that is sent is configurable, as seen in the sections above.

Paragon users then optimise and plans the orders.

When these Paragon routes are frozen, these are exported back to CTMS automatically. This remove any TMP trips, creates RTE trips and sets them to TENDERED status.



# 13 Technical Information

## 13.1 Packages

- DP_PAR_API
- DP_PAR_API_STS

## 13.2 CONFIGURATION - OUTBOUND

```
-- EDI Interface - FROM EDI_IF_CONTROL EXTERNAL_SYSTEM LIKE 'PAR_API%'

SELECT * FROM EDI_PROCESS_HEADER
--WHERE FTP_USERNAME IS NOT NULL
WHERE PROCESS_NAME LIKE '%PAR%'

-- EDI Process Parameters
SELECT * FROM EDI_PROCESS_TRIGGERS
WHERE PROCESS_NAME LIKE '%PAR%'

-- Records are triggered to be written to Paragon from
-- TRG_GEO_LOCATION_UID - note that this does not use system parameter PAR_API
-- TRG_LOCATIONS_UID
-- TI_SCH_ORD_STATUS
-- TRG_SCH_ORDER_LINE_PAR
-- TRG_SCH_ORD_AUDIT_LOCS
-- TRG_SCH_ORD_CURRENT_DEPOT
-- TRG_SHA_TRUNK_TRIP
--Written to INT_XML_CONTROL - check interface type and order

-- NOTE: The FIX interface (for adding locations to a routing plan, to generate routes/drop-numbers)
-- is no longer in use on STSTPRD - commented out code, but still in TMSDEV - watch out for that
-- They manually add the addresses to Paragon now.

-- Packages that reference PAR_API
-- DP_CTMS_IMPORT
-- DP_FLEXIPOD
-- DP_FLEXIPOD_2
-- DP_PAR_API
-- DP_PAR_API_STP
-- DP_SCHEDULING_ENGINE_STAP
-- DP_SCHEDULING_ENGINE
-- TRM

-- Packages that reference PAR_API parameter
-- TI_SCH_ORD_STATUS
-- TRG_LOCATIONS_UID
-- TRG_SCH_ORDER_LINE_PAR
-- TRG_SCH_ORD_AUDIT_LOCS
-- TRG_SCH_ORD_CURRENT_DEPOT
-- TRG_SCH_ORD_REFERENCE_RUN
-- TRG_SHA_TRUNK_TRIP
-- TRM

-- Packages that write 'LOC' paramgon records (FIX interface)
-- DP_PAR_API_STP
-- TI_SCH_ORD_STATUS
-- TRG_GEO_LOCATION_UID

select * from int_xml_control
where external_system = 'PAR_API'
and event_type = 'ORD'
and OMS_REF = 846511
order by int_xml_seq desc

select * from int_xml_control
where external_system = 'PAR_API'
and event_type = 'LOC'
and OMS_REF = 846511
order by int_xml_seq desc
```



## 13.3 Parameters

Parameters - PAR_API Also

- PAR_ENDPOINT_URL
- PAR_TENNANT keys
- KEY_FORMAT

```
-- Searching for parameters
SELECT '|'||PARAM_NAME, '||'||DESCRIPTION, '||'||CONFIG_BY
FROM ADM_SYSTEM_PARAM
--SELECT * FROM ADM_SYSTEM_PARAM
WHERE PARAM_NAME LIKE '%PAR_API%'

SELECT '|'||PARAM_NAME, '||'||DESCRIPTION, '||'||CONFIG_BY
FROM ADM_SYSTEM_PARAM
WHERE PARAM_NAME LIKE 'PAR%'
FOR UPDATE
```

## 13.4 Tenant Keys

Keys matching dates/schedules to Paragon run keys

```
SELECT * FROM PAR_KEYS

SELECT * FROM PAR_KEYS_ROUTE
```

## 13.5 Outbound

```
-- Configuration of outbound processing
-- Orders to paragon
SELECT * FROM PAR_CALL_API
-- locations to paragon
SELECT * FROM PAR_CUST_API
-- Fixed routes to paragon
SELECT * FROM PAR_FIX_API


-- PICKED UP BY (EITHER/OR)
-- DP_PAR_API
-- DP_PAR_API_STS

-- PROCESS_PARAGON

-- That sends the details to Paragon.
```

## 13.6 Inbound

```
-- WARNING: You only want to get as few records as possible here - the results are massive.
-- So, use the following to identify a record you want using PAR_ID, the select * where PAR_ID = ?

-- Inbound Tactical
SELECT PAR_ID, PAR_DATE, PAR_PROCESSED, DEPOT_PLAN
FROM PAR_JSON_IN
ORDER BY PAR_DATE DESC

-- Inbound Strategic
SELECT PAR_ID, PAR_DATE, PAR_PROCESSED, RUN_NUMBER
FROM PAR_JSON_IN_ROUTE
ORDER BY PAR_DATE DESC

-- Where values extracted from JSON are stored for onward processing
-- Written one per order or record
select * from PAR_API_STAGING
select * from PAR_API_STAGING_ROUTE

-- INBOUND STRATEGIC
-- When staged, writes 1 record to INT_XML_CONTROL
```



```
select * from int_xml_control
where external_system = 'PARAGON_IN'
order by int_xml_seq desc

-- Typically runs DP_PAR_API_STP.READ_PARAGON_IN passing the process name as a named parameter

-- This process will ONLY process orders where the FROZEN flag is not 0
-- It does do SOME processing against non-frozen orders, but not a lot

-- When complete processing a record (frozen or not) the record is removed from staging.

-- For frozen orders, there is no auditing.
-- For non-frozen orders, there is auditing against the orders and the trips affected.

-- Other Auditing - only audits failures not successes.
SELECT * FROM EPOD_WEB_SERVICE_AUDIT
WHERE TRIP_ID = 'PARAGON'
AND EVENT_TYPE = 'FAIL'


-- INBOUND ROUTE (FIX) Interface
-- Calls Stage_paragon_route which stages to PAR_API_STAGING_ROUTE

-- Then creates an INT_XML_CONTROL record for PARAGON_IN as above (no idea why)

-- Then directly processes the records on PAR_API_STAGING_ROUTE in PROCESS_ROUTE_DATA
-- not triggered by a background process, so no ides why there is a control record created. No comments to
-- Simply creates Fixed route and stop, created from the PAR_FIX interface
select * from geo_route_dtls

-- Audits:
SELECT * FROM EPOD_WEB_SERVICE_AUDIT
WHERE TRIP_ID = 'PAR_ROUTE';
```





# 14 DHL Invoicing

# 15 DHL Invoicing

The invoicing is transactional so requires data extracted from each system.

# 16 DHL Invoicing - CTMS

Each server has a crontab entry to run the processing on the 1st of the month. It runs at 0421.

```
#Entry for DHL monthly invoicing

21 4 1 * * /oraapp/util/sql/dhl_invoicing >> /tmp/dhl_invoicing
```

The script calls a database process called DP_DHL_INVOICING.RUN_ME

Using the oratab entries (active systems) the script will run against each live database.

The package code is in CVS and should be maintained through the standard code change procedures.

A system parameter called DHL_INV_EMAIL which contains the list of email addresses to send the output to. Multiple entries should be separated with a semi-colon.

The processing runs slightly different iterations of the queries depending on the system. The queries were split into groups based on the agreed costing models.

There is also a listing of users.

Users:

- List of users who logged in in the last month.

Group 1:

- Trips - count of non-deleted trips in the previous month
- Bookings - count of schedule bookings in the previous month
- Scheduled Orders - count of orders' Load activities in the previous month.

Group 2:

- Scheduled Orders - count of orders' Load activities in the previous month, split down by cost centre and planning group (depot).

Databases and Groups:

| Database | Processes Run |
|---|---|
| aam | Users, Group 1, Group 2 |
| bnl | Users, Group 1, Group 2 |
| con | Users, Group 1, Group 2 |
| dun | Users, Group 1, Capped Orders, Group 2 |
| eur | Users, Group 1, Group 2 |
| hcr | Users, Group 1, Group 2, Scheduled Low Volume Orders, Scheduled Standard Orders |
| ind | Users, Group 1, Group 2 |

**Note:**

- dun - includes a list of capped orders (order count is a minimum of 30 orders per trip, or the count of orders per trip, whichever is the larger).
- hcr - includes a split of low-volume orders (volume <= 0.20) and standard orders (any order not in that list above).

Each database will send an email with an attachment which contains tab separated data. It can be pasted directly into Excel.

# 17 DHL Invoicing - LOTS

There is a stored procedure in the MySQL database called dhl_invoicing that contains the SQL.

A bat file D:\LOTS\Invoicing\invoicing.bat runs the stored procedure and emails the file created. The file is then deleted (MySQL cannot overwrite a file).

The list of email addresses is hard coded but it is just a text file that can be edited as required.

The bat file runs from the Windows task scheduler on the first of the month at 0421





# 18 Flexipod Technical Setup

**Warning**: This is an incomplete guide.

Note from Dom:

- Create collection order, adds to a trip
- Set to accepted, sent through to FleXipod.
- Added a new collection order, this is scheduled onto the same accepted trip then appeared on the device even though the trip was ACCEPTED
- BUG: Updates the order items, but does not update the attributes on the drop (or the drops). Does not update the sequence of the drops.

## 18.1 APOD Public API Guide

You can find the full APOD Public API Guide here:

- APOD API Guide - Introduction

You can find help on all of APOD here:

- APOD Online Help

# 19 Ford Orders EDI

Ford EDI is a DHL AA interface, which is a pass-through from DHL Link from SAP.

## 19.1 Content

The content is XML from SAP, containing

- Shipments - the shipment details, plus child nodes
  - Ship_unit - the pallets
  - ship_unit_row - the pallet and details
  - Shipping_line/shipping_line_row - the order line contents

Sample file:

- File:Ford EDI Sample.zip

## 19.2 Process

Package DP_FORD_EDI_IN

Process_file_orig - presumably old.

Process_file - presumably new

Basically,

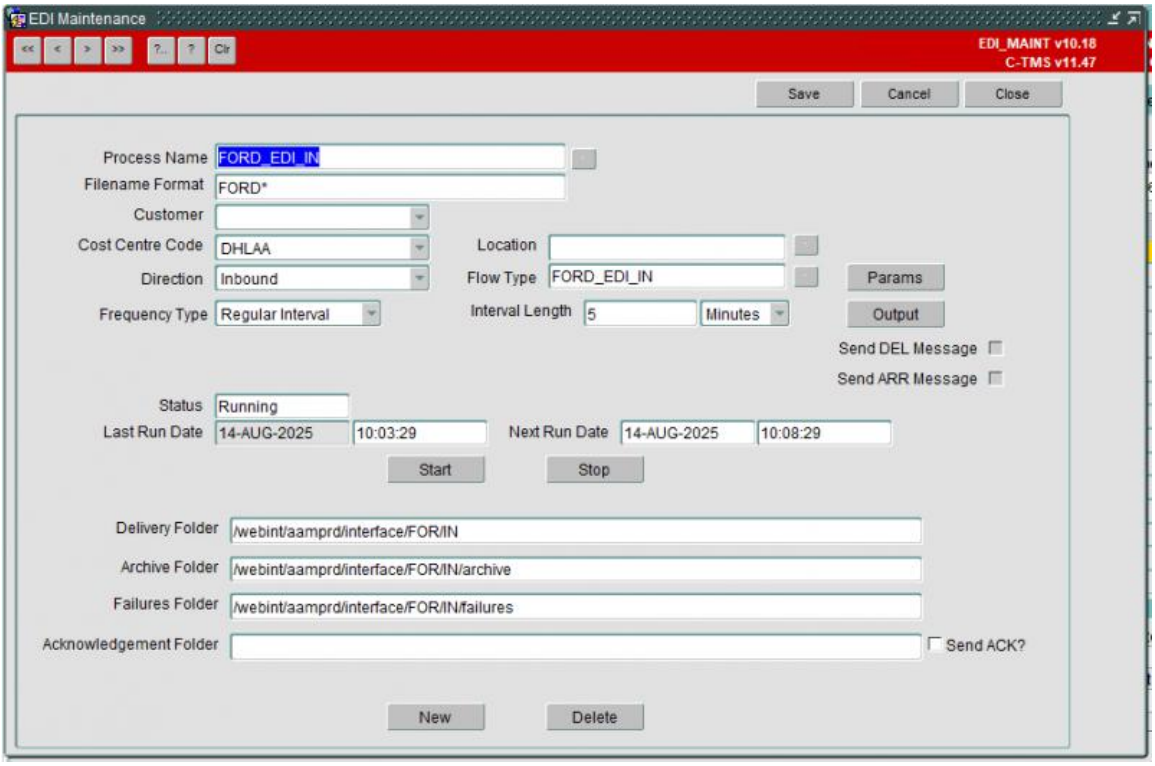
- Finds the order
- Stores the items and contents onto standard EDI tables
- Then processes them to update the order
  - Adds Items
  - Adds contents
  - Generates order lines

## 19.3 Implementation

Create an EDI








- Flow Type: FORD_EDI_IN
- Parameters - none
    - ♦ Report Values - none
    - ♦ Parameters - none

## 19.4 Management

**Warning**: Unknown

# 20 NHSBT Loginext ORDS Support Guide

LogiNext --> https://nhsbt-websvc01.calidusctms.apteancloud.com/ords/import/order/update/

(i) cloudflare ACL restricted (ie outside of ACL cloudflare returns "BLOCKED") - accessible through jump server 10.43.9.32 - a URL GET returns ORDS error page in browser - expected behaviour ORDS interaction is via POST only

(ii) apache webserver

172.23.45.227 RUNnhswebSVC1 [172.20.45.227] SPKnhswebSVC1 [172.20.43.227]

apache : nhsbt-websvc01.calidusctms.apteancloud.com 172.23.45.226 80

LOGGING : cd /usr/local/apache-2.4.46/apache2/logs cat access_log | grep nhsbt-websvc01.calidusctms.apteancloud.com

(iii) (a) apache --> (b) ORDS --> (c) database

172.23.45.226 RUNnhsprdSVC1 [172.20.45.226] SPKnhsprdSVC1 [172.20.43.226]

(a) apache : nhsbt-websvc01.calidusctms.apteancloud.com 127.0.0.1 8500

LOGGING :

cd /usr/local/apache-2.4.46/apache2/logs cat access_log | grep order

(b) ORDS

/oradb19/ords ords_nhstprd stop ords_nhstprd start

ps -ef |grep ords | grep 8500

/oradb19/ords/java/jdk-21.0.5+11-jre/bin/java -Doracle.dbtools.cmdline.home=/oradb19/ords/nhstprd -Duser.language=en -Duser.region=US - Djava.util.logging.config.file=/oradb19/ords/nhstprd/bin/logging.properties -Djava.awt.headless=true -Dnashorn.args=--no-deprecation-warning - Doracle.dbtools.cmdline.ShellCommand=ords -Duser.timezone=UTC -jar /oradb19/ords/nhstprd/ords.war serve --port 8500

NOTE : GENERATED tokens have a default 1 hour expiry

LOGINEXT APPLICATION HAS A ONEOFF 10 YEAR TOKEN

LOGGING :

note : Java logging is set to FINE

cd /tmp ls -ltr ords*

minutely count : ls -tr ords.nhstprd.log* | while read LINE; do cat $LINE; done | grep order | grep TIM | awk -F"start: " '{print($2)}' | awk -F":" '{print($1":"$2)}' | sort | uniq -c

daily count : ls -tr ords.nhstprd.log* | while read LINE; do cat $LINE; done | grep order | grep TIM | awk -F"start: " '{print($2)}' | awk -F"T" '{print($1)}' | sort | uniq -c 2085 2025-11-21 972 2025-11-22 891 2025-11-23 734 2025-11-24

raw ms ordered ls -tr ords.nhstprd.log* | while read LINE; do cat $LINE; done | grep order | grep TIM | awk -F"duration:" '{print($2)}' | awk -F"ms" '{print($1)}' | sort -n > ords.nhstprd.txt





(c) database

note : when querying the ORDS schema login as ordsimport/{password}

```
                    LOGGING :


                    select * from ords_metadata.sec_sessions

                    {Results will be displayed}


                    select * from user_ords_clients;


-----------------------------------------------------------------
ORDS PL/SQL block
-----------------------------------------------------------------


BEGIN
  ORDS.define_handler(
    p_module_name    => 'rest-json-import-order-update',
    p_pattern        => 'update/',
    p_method         => 'POST',
    p_source_type    => ORDS.source_type_plsql,
    p_source         =>
q'[
    declare
        l_response json_object_t := json_object_t();

  CHUNK_SIZE constant pls_integer := 8192;
  vOffset  pls_integer := 1;
  vData     clob := EMPTY_CLOB();
  vChunk   varchar2(CHUNK_SIZE CHAR);

begin


l_response := mts_owner.dp_loginext.process_inbound(i_message =>  json_object_t.parse(:body));

owa_util.mime_header ('application/json', true);

vData := l_response.to_clob;
  loop
    vChunk := substr (vData, vOffset, CHUNK_SIZE);
    exit when vChunk is null;
    htp.prn(vChunk);
    vOffset := vOffset + length(vChunk);
  end loop;

exception when others then

l_response :=  JSON_OBJECT_T.parse ('{"status":"exception","sqlcode":"'||sqlcode||'","sqlerrm":"'||sqlerrm|
htp.p (l_response.stringify);


end;
    ]',
    p_items_per_page => 0);

  COMMIT;

END;
/

-----------------------------------------------------------------



Postman

        https://nhsbt-websvc01.calidusctms.apteancloud.com/ords/import/oauth/token
```





```
        select * from user_ords_clients;

        Grant Type              Client Credentials
        Client ID               {from previous query}
        Client Secret           {from previous query}
        Scope                   Bearer
        Client Authentication   Send as Basic Auth header

        - will generate a token which expires in 1 hour


        or just use client 10 year token


A basic test is to just pass a dummy payload to the webservice
        {abc}

returns :
        {"status":"exception","sqlcode":"-40441","sqlerrm":"ORA-40441: JSON syntax error"}
```





# 21 Oracle ORDS Technical Guide

This is a technical guide for the setup of Oracle ORDS, used for OAUTH2 webservices through Oracle Fusion Middleware.

**Note:** The majority of the ORDS schema creation is through a WAR file that is installed when the process is initialized.

The initial build consists of a java unzip, and then using the WAR file to create schemas / deploy objects to the database (logged in as sys) - this WAR file is also the ?ORDS java? side and a minor amount of config.

This guide is intended to be a small introduction to the technical tables and requirements, for support purposes.

## 21.1 Pre-requirements

- PL/SQL Developer.
- TNS Name for the Oracle database.

#### 21.1.1 Users

- ords
- ordsimport
- mts_owner

## 21.2 Overview of Affected Tables

### 21.2.1 Settings for users when editing ORDS

```
select * from user_ords_roles;
select * from user_ords_privileges;
select * from user_ords_privilege_roles;
select * from user_ords_privilege_mappings;
```

### 21.2.2 Creation

The actual web services

```
select * from user_ords_modules;
select * from user_ords_services;
select * from user_ords_handlers;
select * from user_ords_templates;
select * from user_ords_schemas;

select * from user_ords_parameters;
```

### 21.2.3 Execution

```
select * from v$session where type = 'USER'
select * from user_ords_clients;
select * from user_ords_client_privileges;
select * from user_ords_client_roles;
```

## 21.3 Setting Up

As mts_owner:

```
create user ords identified by {password} default tablespace data temporary tablespace temp profile defa
grant ords_administrator_role to ords;
grant ords_runtime_role to ords;
grant create session to ords;
grant select any table to ords;

create user ordsimport identified by {password} default tablespace data temporary tablespace temp;

grant create session to ordsimport;
grant ords_runtime_role to ordsimport;
grant select any table to ordsimport;
```





```
grant execute on mts_owner.dp_ctms_import to ordsimport;
```

**Note:** The grant of execute on the package must be the package that the ORDS handler is going to execute within the MTS_OWNER space. If this is different (or there are several), then this must be changed and added to here.

as ords:

```
begin
  ords_metadata.ords.enable_schema(
    p_schema              => 'ordsimport',
    p_url_mapping_type    => 'BASE_PATH',
    p_url_mapping_pattern => 'import'
  );

  commit;
end;
/

declare
  l_roles_arr    owa.vc_arr;
  l_patterns_arr owa.vc_arr;
begin
  l_roles_arr(1)    := 'ordsimport_role';
  l_patterns_arr(1) := 'ordsimport_pattern/';

  ords_metadata.ords.define_privilege (
    p_privilege_name => 'ordsimport_priv',
    p_roles          => l_roles_arr,
    p_patterns       => l_patterns_arr,
    p_label          => 'test priviledge',
    p_description    => null
  );

  commit;
end;
/

select * from user_ords_privileges;
select * from user_ords_privilege_roles;
select * from user_ords_privilege_mappings

begin
  ords_metadata.oauth.create_client(
    p_name            => 'ordsimport_client',
    p_grant_type      => 'client_credentials',
    p_support_email   => 'noreply@email.com',
    p_privilege_names => 'ordsimport_priv'
  );

  commit;
end;
/

select * from user_ords_clients;
select * from user_ords_client_privileges;

begin
  ords_metadata.oauth.grant_client_role(
    p_client_name => 'ordsimport_client',
    p_role_name   => 'ordsimport_role'
  );
  commit;
end;
/

select * from user_ords_client_roles;
```

## 21.4 Examples

Example Order Create - example uses XML - REST, XML-based order create method/service.

```
BEGIN
  ORDS.define_module(
    p_module_name    => 'rest-xml-import-order-create',
```





```
    p_base_path      => 'order',
    p_items_per_page => 0);

  ORDS.define_template(
   p_module_name    => 'rest-xml-import-order-create',
   p_pattern        => 'create/');

  ORDS.define_handler(
    p_module_name    => 'rest-xml-import-order-create',
    p_pattern        => 'create/',
    p_method         => 'POST',
    p_source_type    => ORDS.source_type_plsql,
    p_source         =>
q'[
    declare
        l_response xmltype;

  CHUNK_SIZE constant pls_integer := 8192;
  vOffset  pls_integer := 1;
  DOC1     clob;
  vChunk   varchar2(CHUNK_SIZE CHAR);

    begin

        l_response := mts_owner.dp_ctms_import.import_order(ctms_ord => XMLTYPE.createXML(:body_text));
        owa_util.mime_header ('application/xml', true);

DOC1 := XMLTYPE.getClobVal(l_response);
  loop
    vChunk := substr (DOC1, vOffset, CHUNK_SIZE);
    exit when vChunk is null;
    htp.prn(vChunk);
    vOffset := vOffset + length(vChunk);
  end loop;

     end;
    ]',
    p_items_per_page => 0);

  COMMIT;
END;
/
```

Example handler for LogiNext interface - REST, JSON-based order update method/service.

```
BEGIN
  ORDS.define_module(
    p_module_name    => 'rest-json-import-order-update',
    p_base_path      => 'order',
    p_items_per_page => 0);

  ORDS.define_template(
   p_module_name    => 'rest-json-import-order-update',
   p_pattern        => 'update/');

  ORDS.define_handler(
    p_module_name    => 'rest-json-import-order-update',
    p_pattern        => 'update/',
    p_method         => 'POST',
    p_source_type    => ORDS.source_type_plsql,
    p_source         =>
q'[
    declare
        l_response json_object_t := json_object_t();

  CHUNK_SIZE constant pls_integer := 8192;
  vOffset  pls_integer := 1;
  vData     clob := EMPTY_CLOB();
  vChunk   varchar2(CHUNK_SIZE CHAR);

begin


l_response := mts_owner.dp_loginext.process_inbound(i_message =>  json_object_t.parse(:body));

owa_util.mime_header ('application/json', true);

vData := l_response.to_clob;
```





```
  loop
    vChunk := substr (vData, vOffset, CHUNK_SIZE);
    exit when vChunk is null;
    htp.prn(vChunk);
    vOffset := vOffset + length(vChunk);
  end loop;

exception when others then

l_response :=  JSON_OBJECT_T.parse('{"status":"exception","sqlcode":"'||sqlcode||'","sqlerrm":"'||sqlerr
htp.p(l_response.stringify);


end;
    ]',
    p_items_per_page => 0);

  COMMIT;
END;
/
```

## 21.5 Basic Queries

```
select * from user_ords_handlers
where UPPER(source) like '%DP_LOGINEXT%'
```





# 22 Support - Tesla EDI Inbound Guide

## 22.1 Order Template

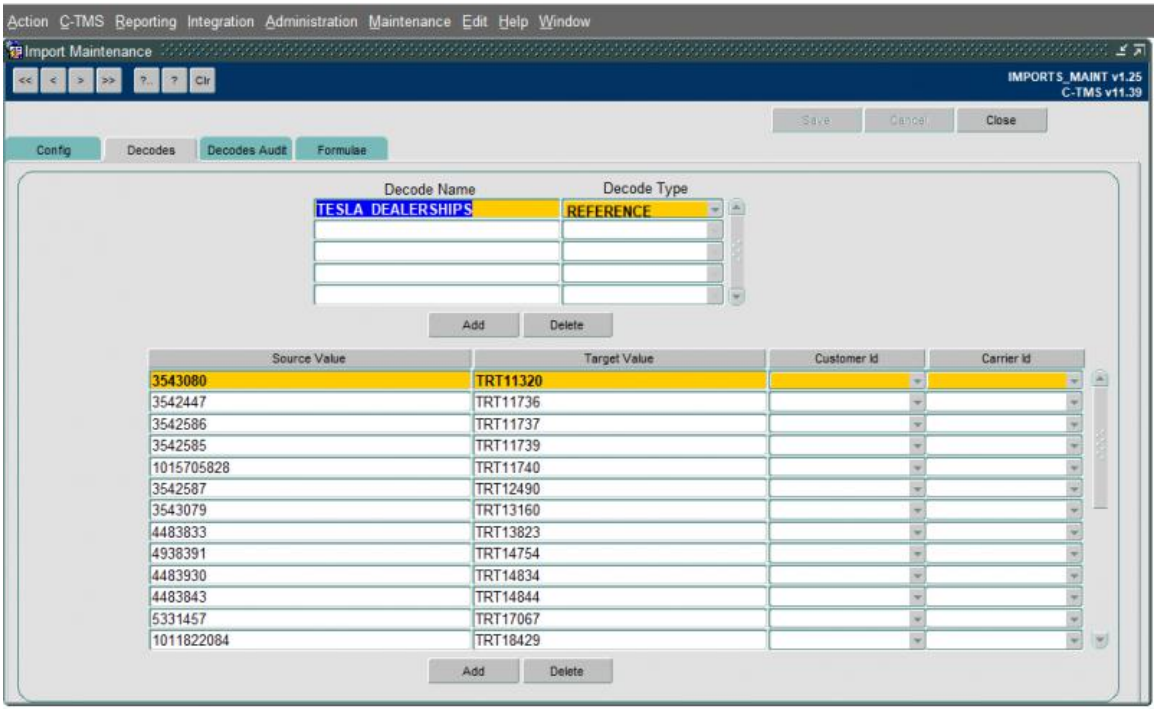
For each dealership an order will be created each day, this is a placeholder which will be added to by the import process. This is existing fixed schedules functionality which has not been modified for Tesla.

## 22.2 System parameters

ALLOW_MULTIPLE_ADD_REFS should be set to Y for the appropriate cost centre.

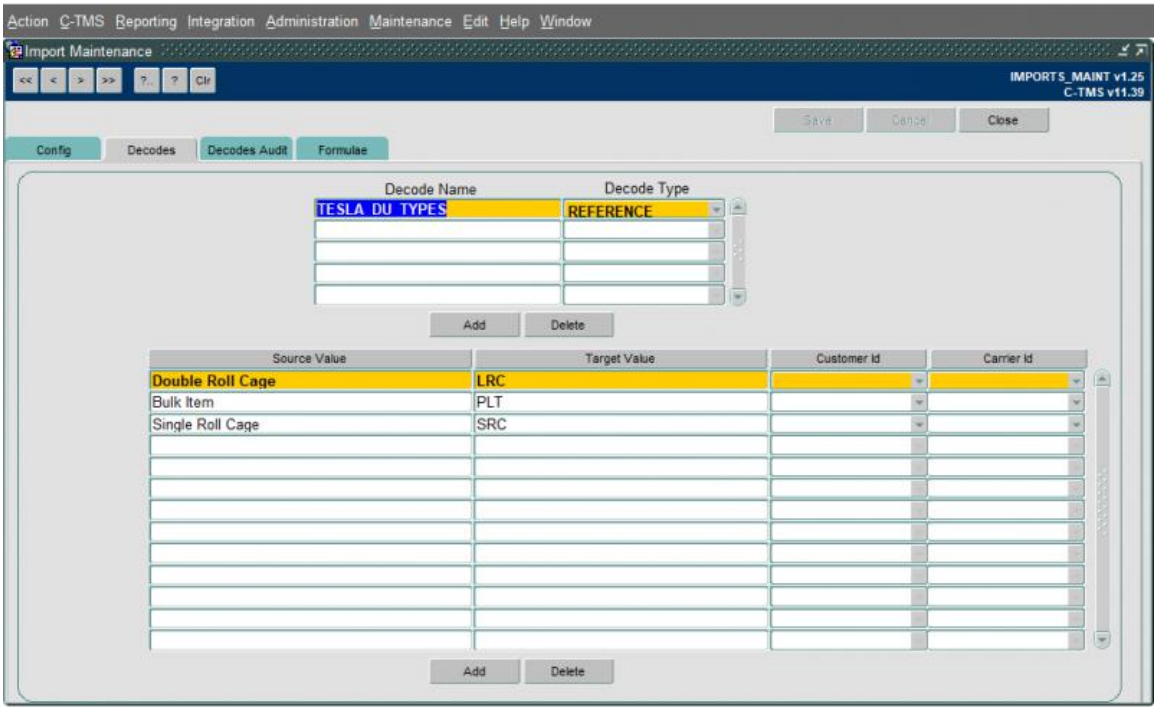
## 22.3 Dealership Decodes

The delivery locations in the inbound files are the Tesla dealership locations not CTMS locations and as such must be decoded, scripts have been created and released during the normal release process to create these decodes




## 22.4 Du Type Decodes

The du types in the inbound files are the Tesla du types not du types and as such must be decoded, scripts have been created and released during the normal release process to create these decodes




## 22.5 Inbound Files

An example inbound file is included below

## 22.6 EDI Tables

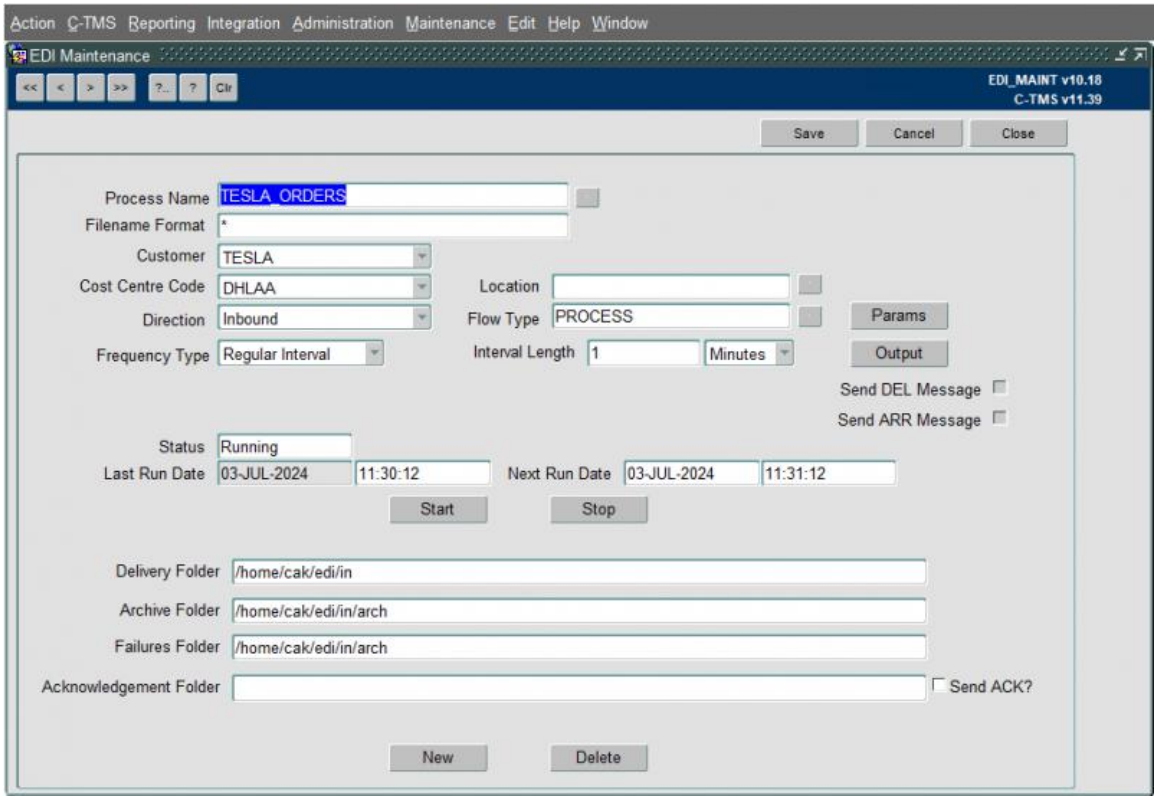
Two new tables have been created to hold the details of the Inbound files

- TESLA_EDI_ORDER_HEADER
- TESLA_EDI_ORDER_DETAILS

Each table has an associated sequence, and the detail is linked to the header by the header ID.

## 22.7 Inbound EDI

A new EDI flow will be present for the DHLAA cost centre and the Tesla customer. The import processes will be completed within the DP_TESLA_EDI package using the IMPORT_ORDERS procedure.

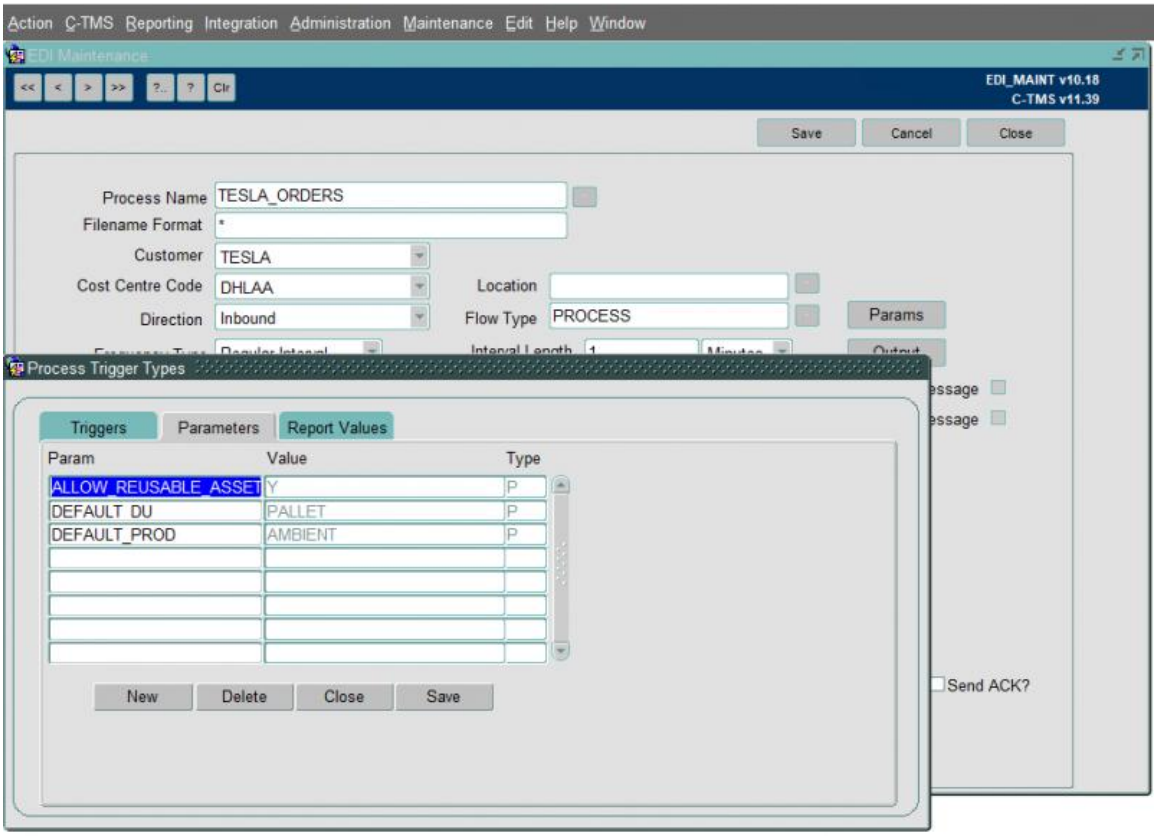



## 22.8 File Format

The test files provide simply have numeric a file name, so the file pattern is not specified on the EDI flow. An example file is included above.

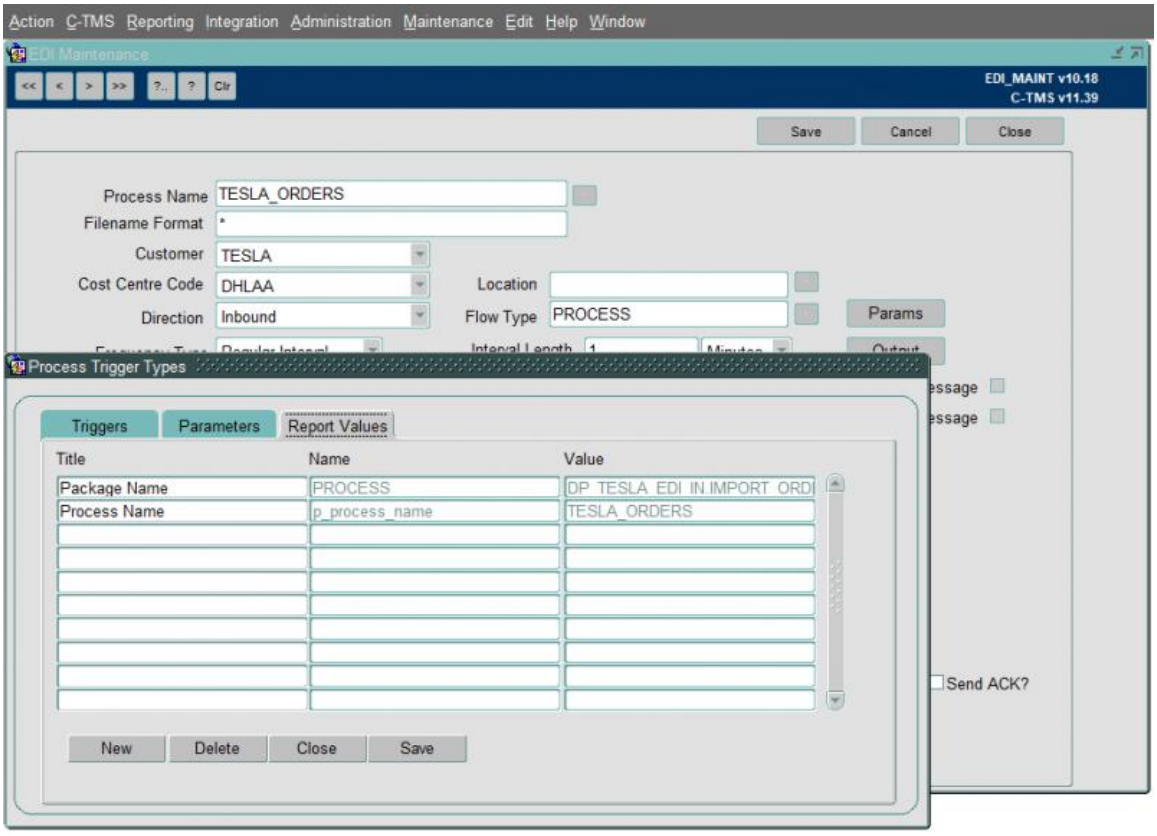
## 22.9 EDI Parameters

The following Parameters are required




## 22.10 EDI Report Values

The following Report values are required

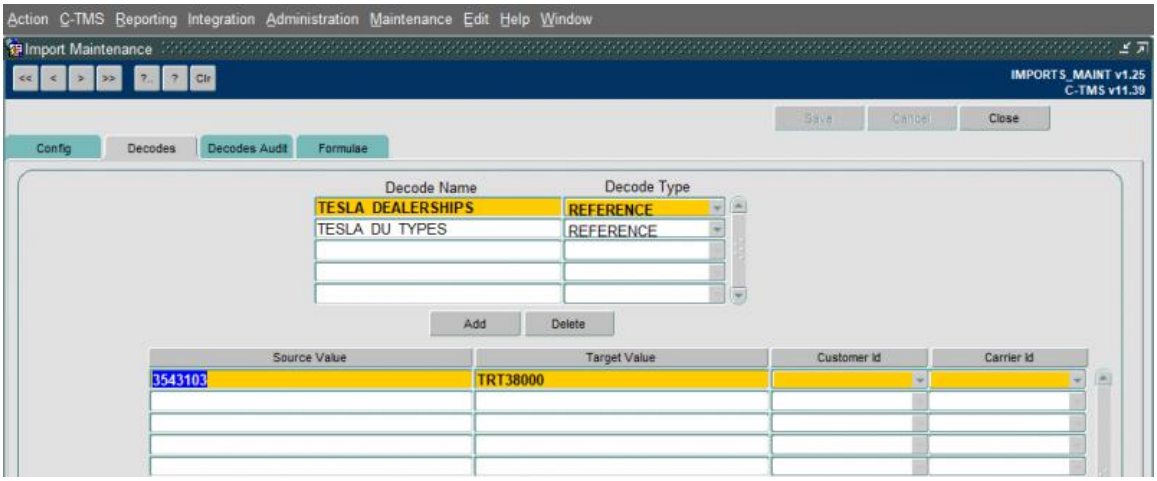


## 22.11 Order Processing

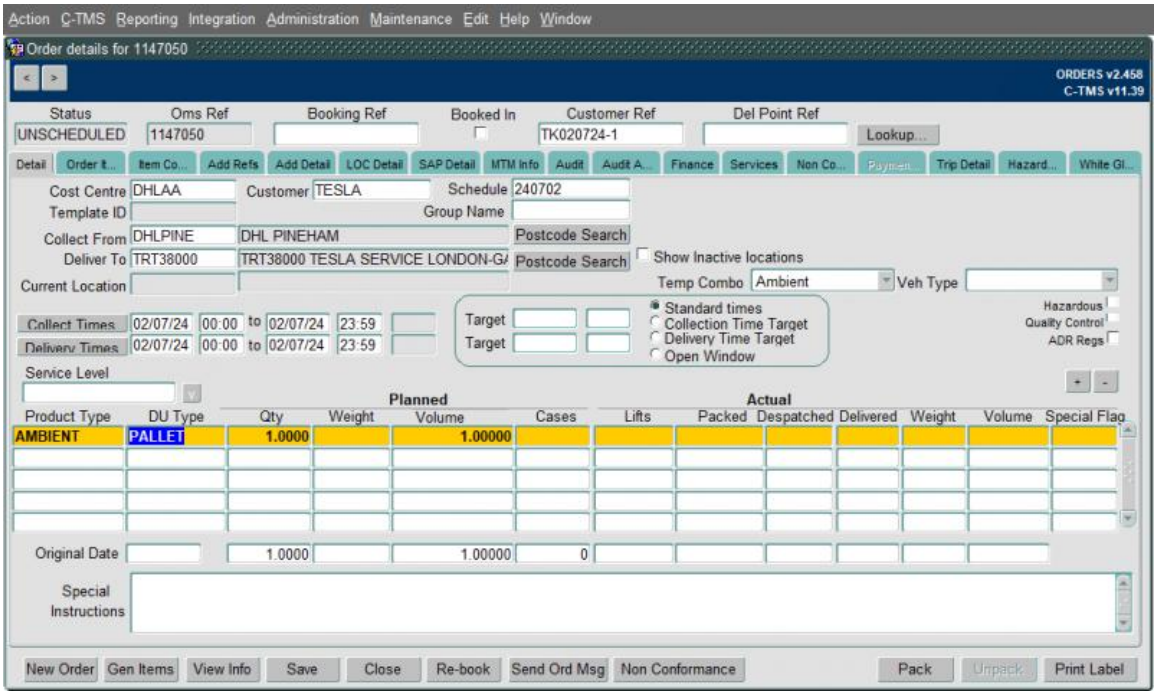
An order will exist on the correct schedule for each dealership in the inbound file the delivery date can be found in the ?deliveryStartDateTime? tag and the dealership in the ?referenceField7? tag

```
<order>
<orderNumber>0009820014</orderNumber>
<carrierName>TTN</carrierName>
<carrierServiceLevel>STD</carrierServiceLevel>
<referenceField6>VIN</referenceField6>
<referenceField7>3543103</referenceField7>
<referenceField8>45516</referenceField8>
<referenceField9>0</referenceField9>
<pickupStartDateTime>2024-06-16T06:29:00</pickupStartDateTime>
<deliveryStartDateTime>2024-07-02T10:57:41.828</deliveryStartDateTime>
```

The correct CTMS dealership location will then be found using the decode




A skeleton order will exist in CTMS which will have been created by an existing process note the schedule and delivery location agree with your inbound file, if no order is found on the schedule for the dealership an error will be raised and the order will not be processed.

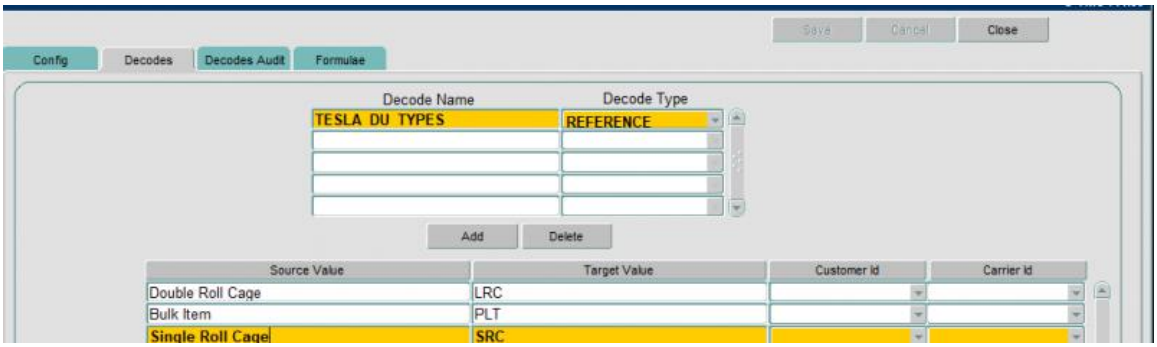



The product type is not contained in the inbound file so therefore must exist as a default parameter on the EDI flow. If no default is present an error will be raised and the order will not process.

The Du type must also exist as an EDI parameter on the flow. An attempt will be made to decode the passed in palletSize value which contains a du type which has been mapped so in this example

```
]<pallet>
 <palletID>PAL01072024</palletID>
 <palletType>Roll_Cages_TTN</palletType>
 <palletSize>Single Roll Cage</palletSize>
```

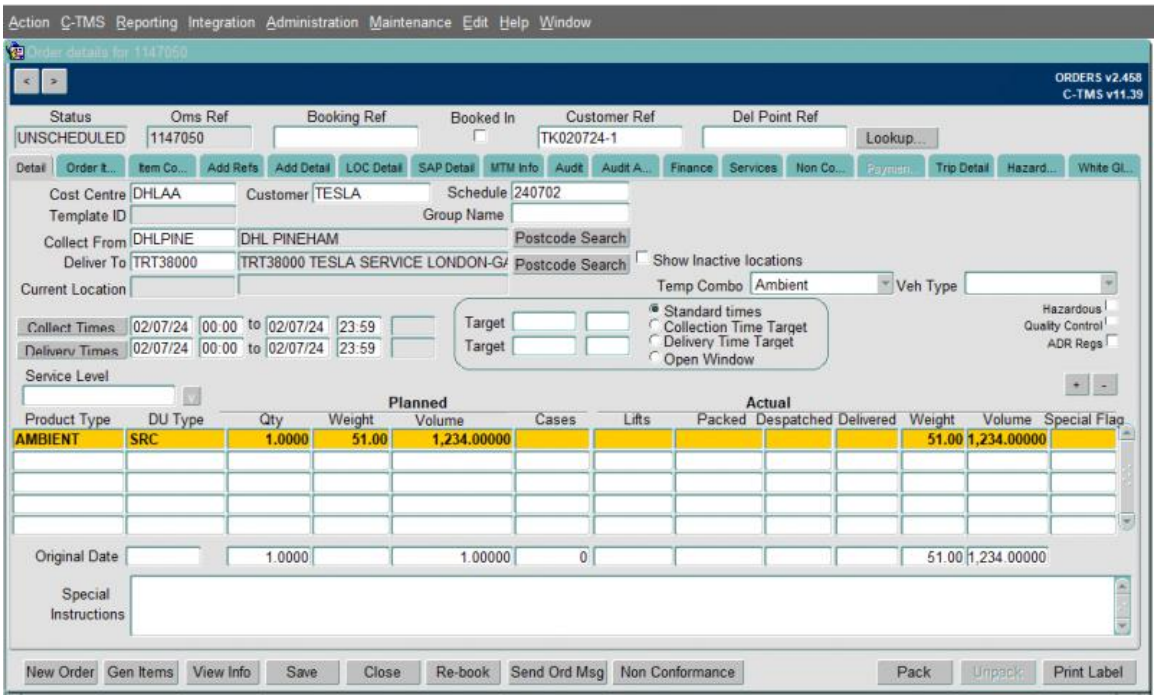
Maps to




If no mapping is found the EDI default will be used. If no default is present an error will be raised and the order will not process.

Once the correct order has been located and the product and DU type validated. The order will be updated with the details in the inbound file.

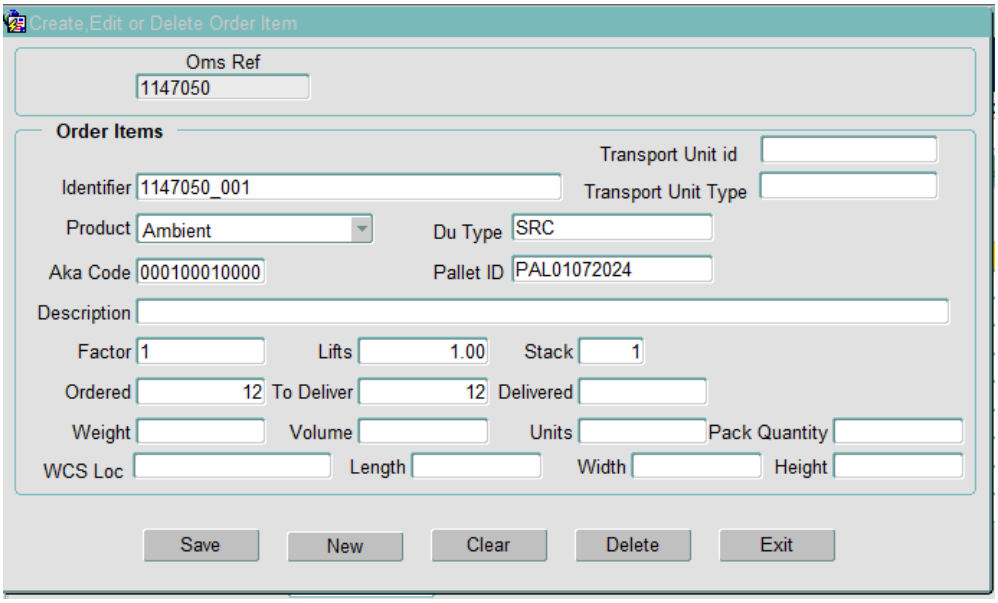
If the DU type doesn?t exist on the current order a new line will be created for that DU type, the placeholder line will be removed at this point if no items exist with the same du type.



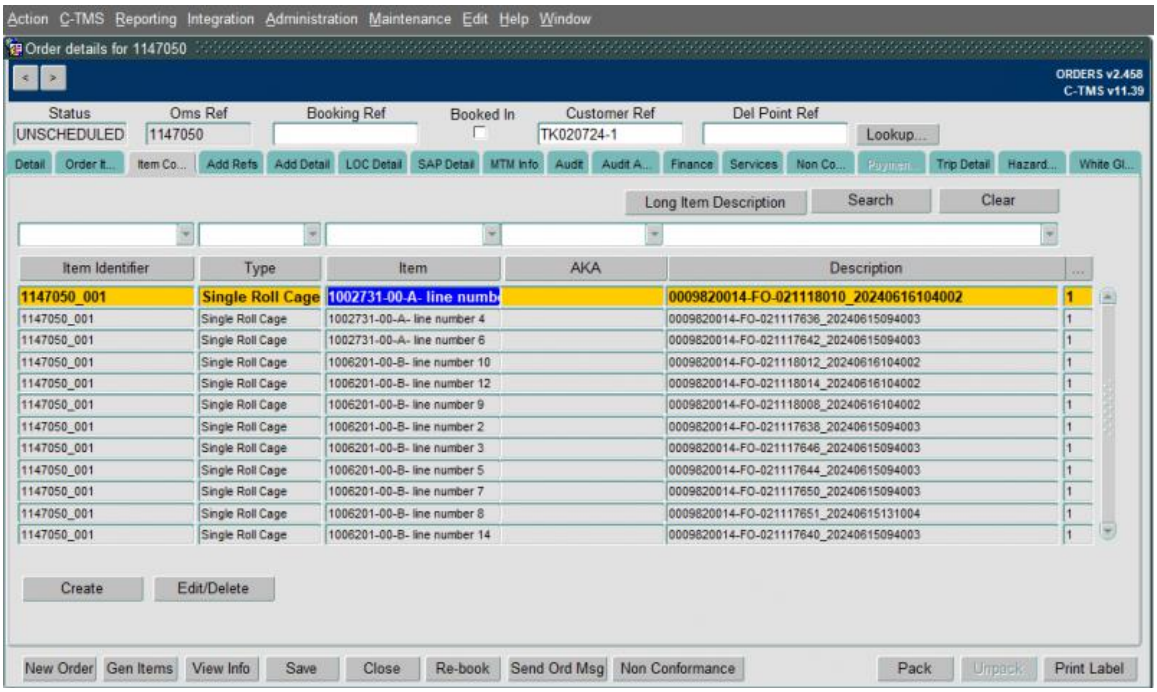

Note the weight and volume on the line these are taken directly from the inbound pallet section and should not be manipulated.

```
<pallet>
<palletID>PAL01072024</palletID>
<palletType>Roll_Cages_TTN</palletType>
<palletSize>Single Roll Cage</palletSize>
<palletLength>85</palletLength>
<palletWidth>74</palletWidth>
<palletHeight>169</palletHeight>
<palletWeight>51</palletWeight>
<palletVolume>1234</palletVolume>
```

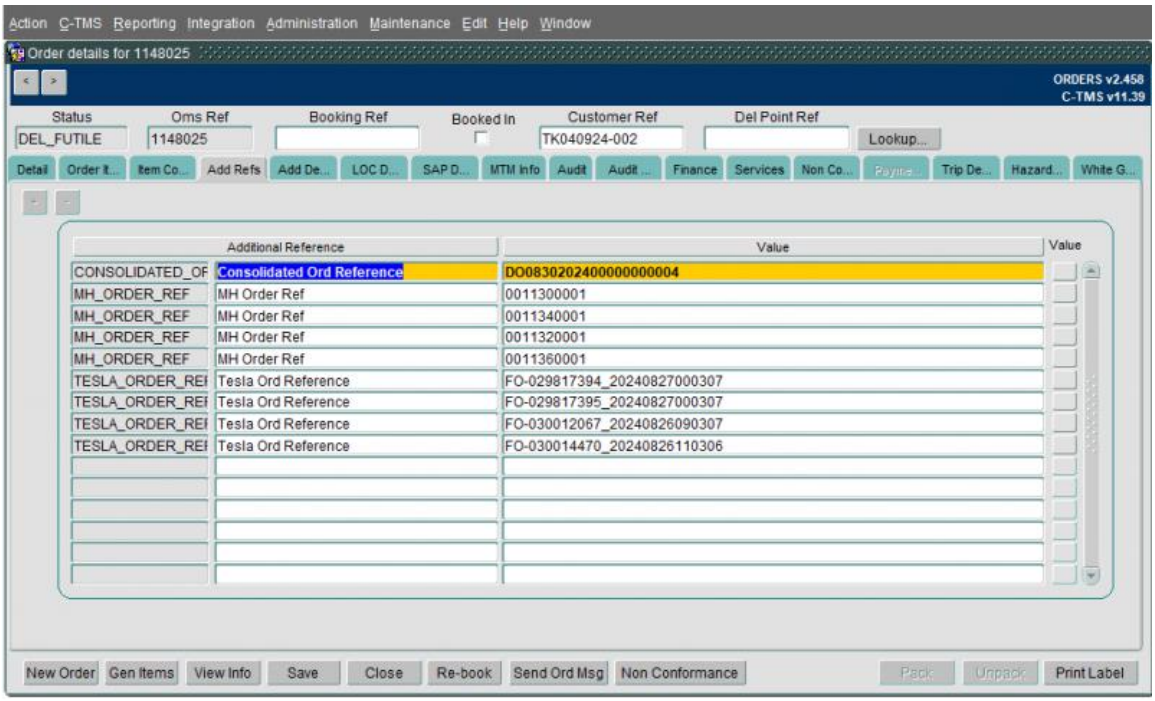
Order items will have been created




Item contents will have been created




Additional References will have been created to indicate the Consolidated order reference, MH Order references(s) and Tesla references(s)




Each reference should only be stored once and the values are found as follows

Consolidated Order - ?aggregatedOrderNumber?  tag, only 1 per order will be stored

```
<order>
<orderNumber>0011320001</orderNumber>
<aggregatedOrderNumber>DO08302024000000000004</aggregatedOrderNumber>
<carrierName>DSC  TTN</carrierName>
<carrierServiceLevel>STD</carrierServiceLevel>
<referenceField6>VIN</referenceField6>
<referenceField7>1008612153</referenceField7>
```

MH(Manhattan) Order ref - ?OrderNumber? tag

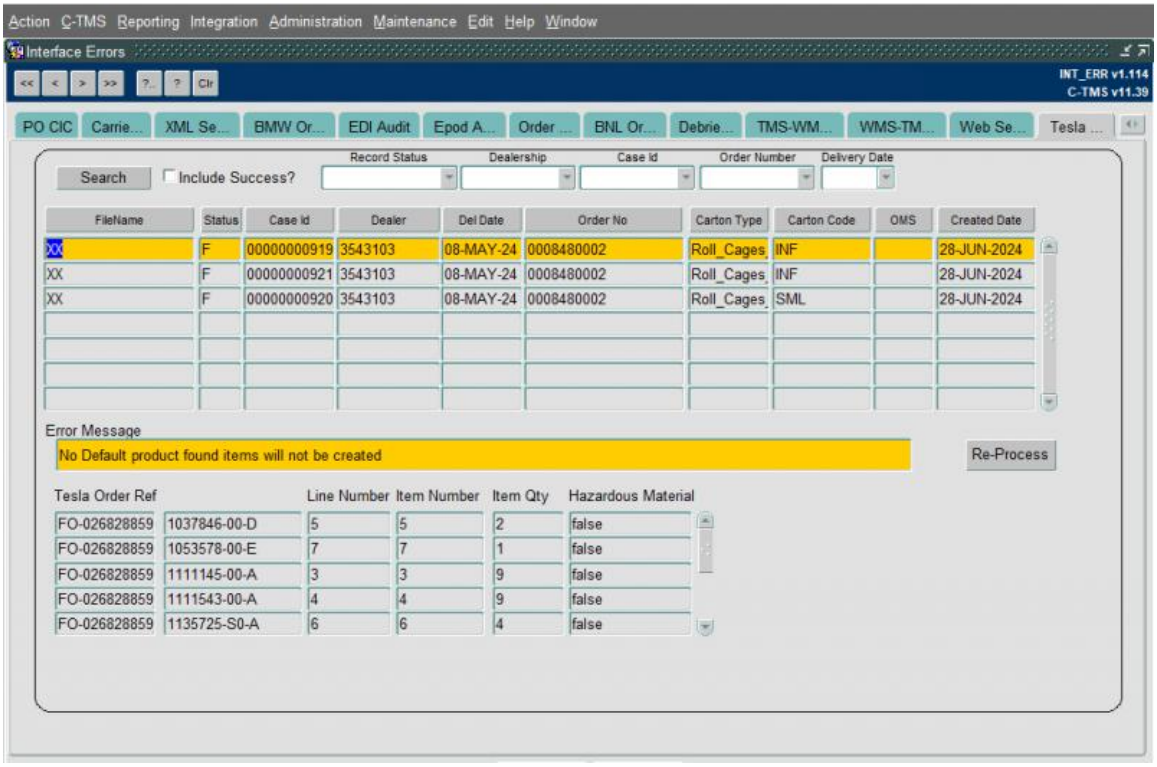
```
<order>
<orderNumber>0011320001</orderNumber>
<aggregatedOrderNumber>DO08302024000000000004</aggregatedOrderNumber>
<carrierName>DSC  TTN</carrierName>
<carrierServiceLevel>STD</carrierServiceLevel>
<referenceField6>VIN</referenceField6>
<referenceField7>1008612153</referenceField7>
```

Tesla Reference - ?referenceField4? of the line section

```
<orderLines>
<line>
<lineNumber>11</lineNumber>
<itemNumber>1002731-00-A</itemNumber>
<orderedQuantity>1</orderedQuantity>
<shippedQuantity>0</shippedQuantity>
<unitOfMeasure>Units</unitOfMeasure>
<inventoryStatus>A</inventoryStatus>
<referenceField1>STOCK</referenceField1>
<referenceField2>7505448916</referenceField2>
<referenceField3>DELIVERY</referenceField3>
<referenceField4>FO-021118004_20240616104002</referenceField4>
<referenceField5>58771966</referenceField5>
```

## 22.12 Interface Errors

A new Tesla Orders tab page has been added to the interface errors screen which will contain the success/failure stats of each or the file imports and allow the user to reprocess any failed records after the issue has been corrected. This process will use the same package functionality as the inbound orders.




## 22.13 Specifications

P:\Development\CTMS\DHL\543347 SCR - Tesla Order Interface

P:\Development\CTMS\DHL\613005 SCR - Changes to the Tesla Order Interface

P:\Development\CTMS\DHL\632041 - Changes to Tesla order interface for Aggregated order number

# 23 Support - Tesla EDI Outbound Guide

## 23.1 Order Tracking

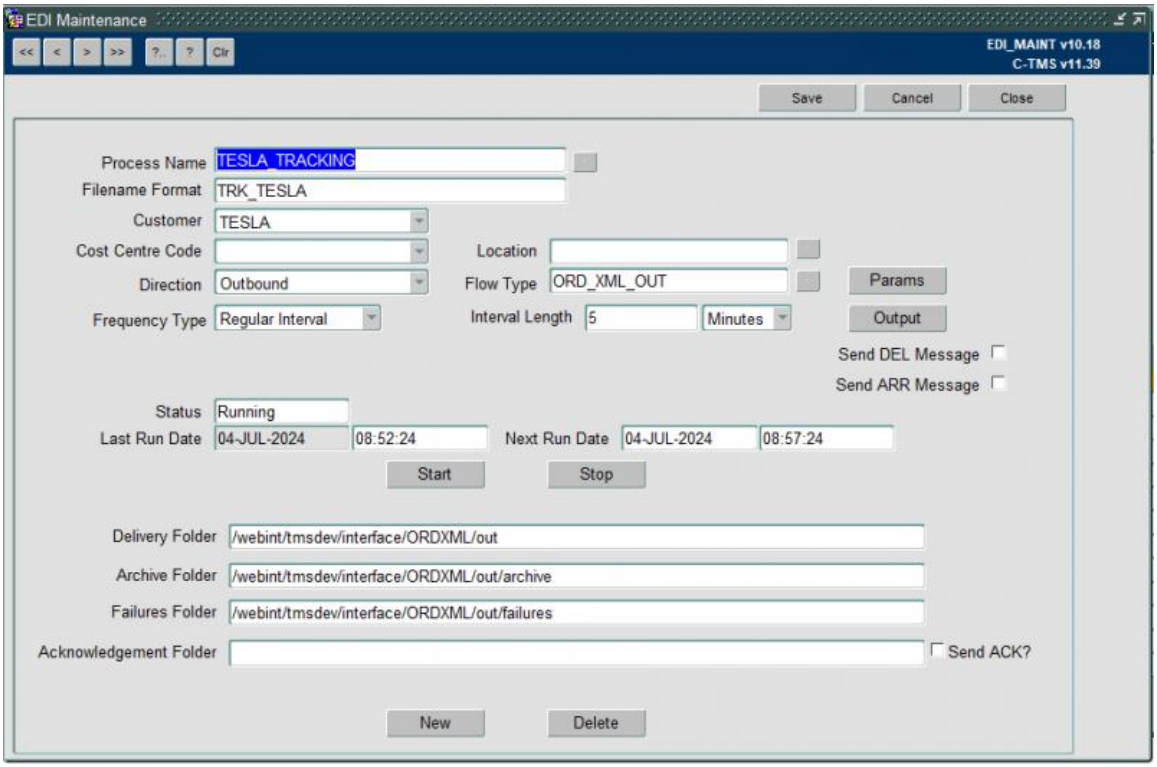
A customer specific tracking interface is required to track the milestones associated with an order.

## 23.2 System parameters

USE_XML_ORDER_SUB_REFS should be set to Y for the appropriate cost centre.

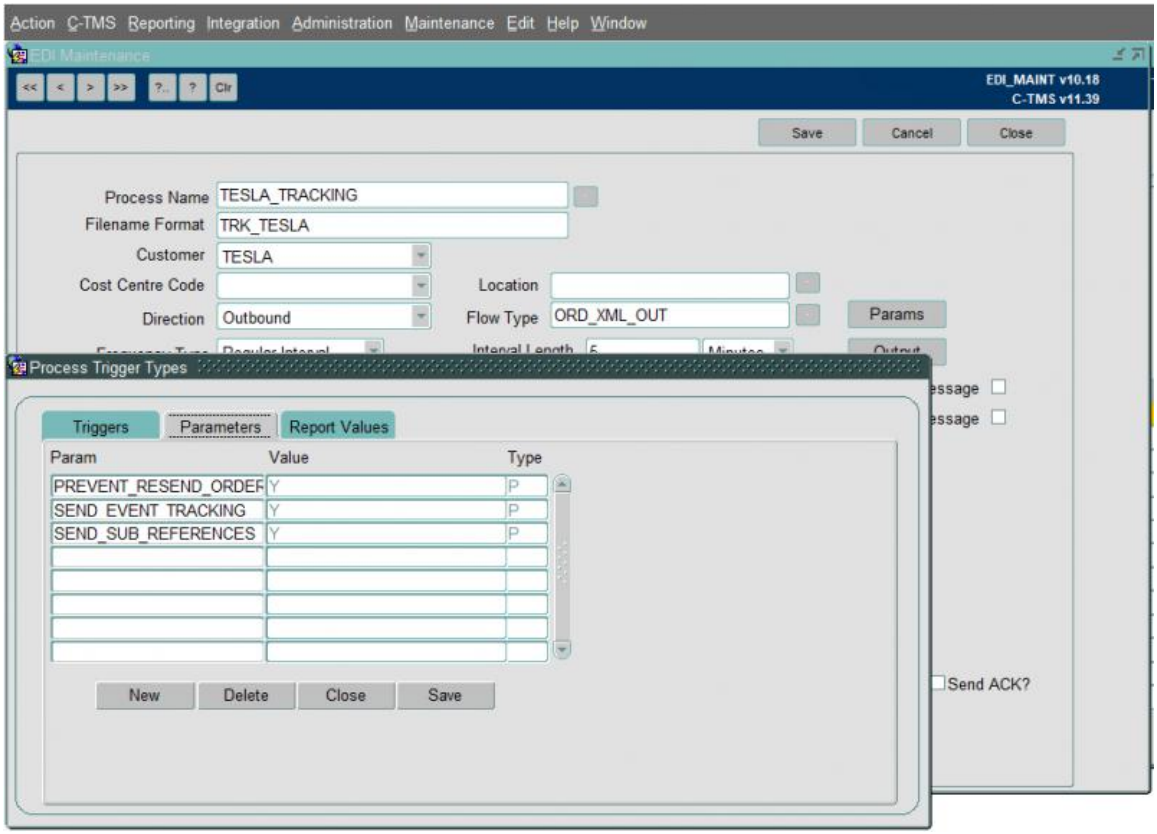
## 23.3 Outbound EDI

A new EDI flow will be present for the DHLAA cost centre and the Tesla customer. The flow uses the standard ORD_XML_OUT flow type and includes a Milestones section.




## 23.4 EDI Parameters

The following Parameters are required




## 23.5 Messages sent

The following message types will be sent

- ORD
- CAN
- DEP
- ARR
- DEL

## 23.6 Triggering Events

An **?ORD?** message will be triggered when the first item is added to an order and will only be sent once. The EDI parameter called ?PREVENT_RESEND_ORDER? will be set to ?Y? to ensure that a control record for an ?ORD? message is not created when an earlier message has been processed successfully. The trigger ?TRG_SOI_XML? will create the control record. The OMS status is ?UNSCHEDULED? and the milestone comment is ?Booking Acknowledged?

A **?DEP?** message will be created when the actual departure time on the collection stop of an order is updated to show that the vehicle has left the collection location.

The trigger ?TRG_STS_XML_OUT? will create the control record. The OMS status is ?SCHEDULED? and the milestone comment is ?Departed Origin? if this is the initial collection location or ?Departed Consolidation Facility? if the location is a x-dock location.

An **?ARR?** message will be created when the actual arrival time on the delivery stop of an order is updated to show that the vehicle has arrived at the delivery location. The trigger ?TRG_STS_XML_OUT? will create the control record. The OMS status is ?SCHEDULED? and the milestone comment is ?At Dest? if this is the final delivery location or ?At Consolidation Facility? if the location is a x-dock location.

****Note** If an order exists on multiple trips the ?DEP? and ?ARR? messages will be sent for each leg of the orders journey.

A **?DEL?** message will be created when the delivered quantities on an order line are updated for the first time, i.e. when the delivered quantity is changed from null to 1, for example. The trigger TRG_SOL_XML will create the control record

A **?CAN?** message will be triggered if the status of an order is set to ?CANCELLED?. The trigger ?TRG_SCH_ORD_XML_INT? will create the control record. The OMS status is ?CANCELLED? and the milestone comment ?Booking Rejected?.

## 23.7 Specifications

P:\Development\CTMS\DHL\545095 - SCR-CTMS-02428249-05 - CTMS - Customer-Specific Tracking File

P:\Development\CTMS\DHL\623640- Changes to Tesla tracking milestones

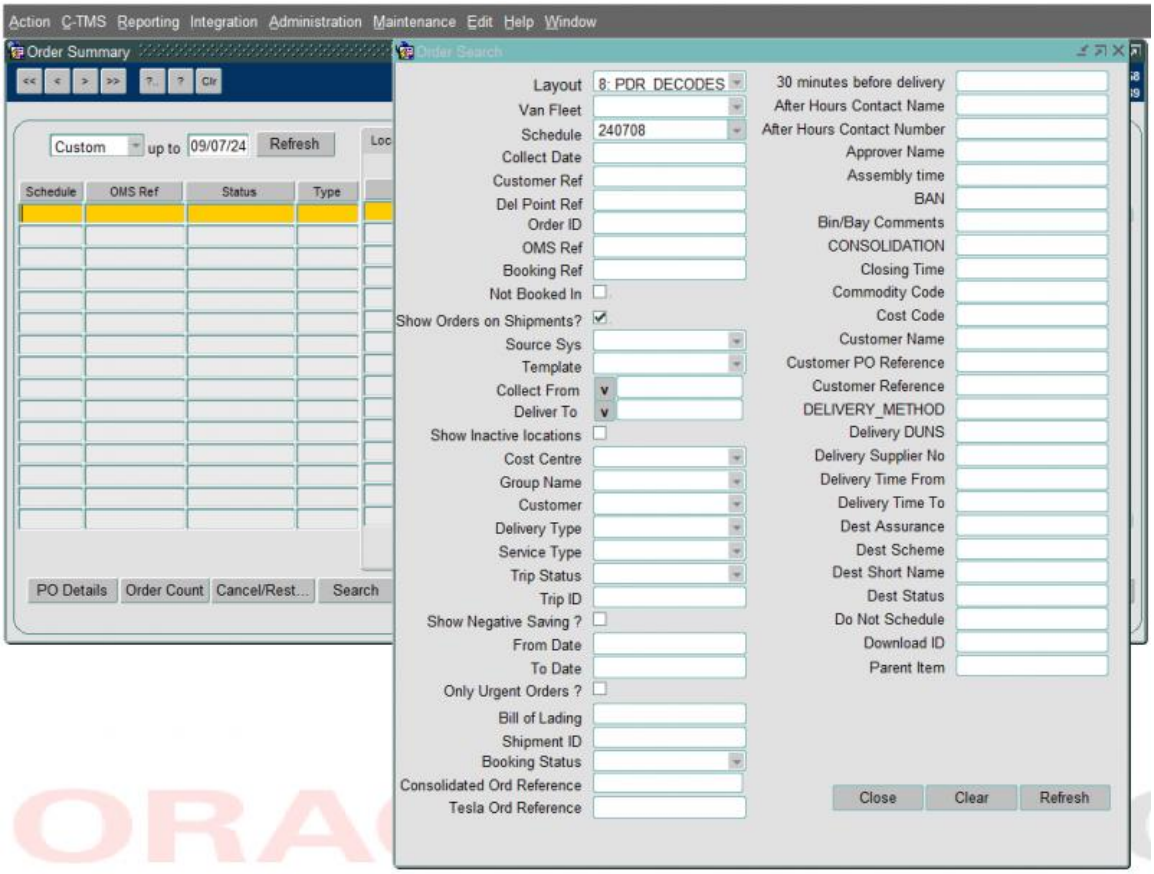




# 24 Support - Tesla Order Search Guide

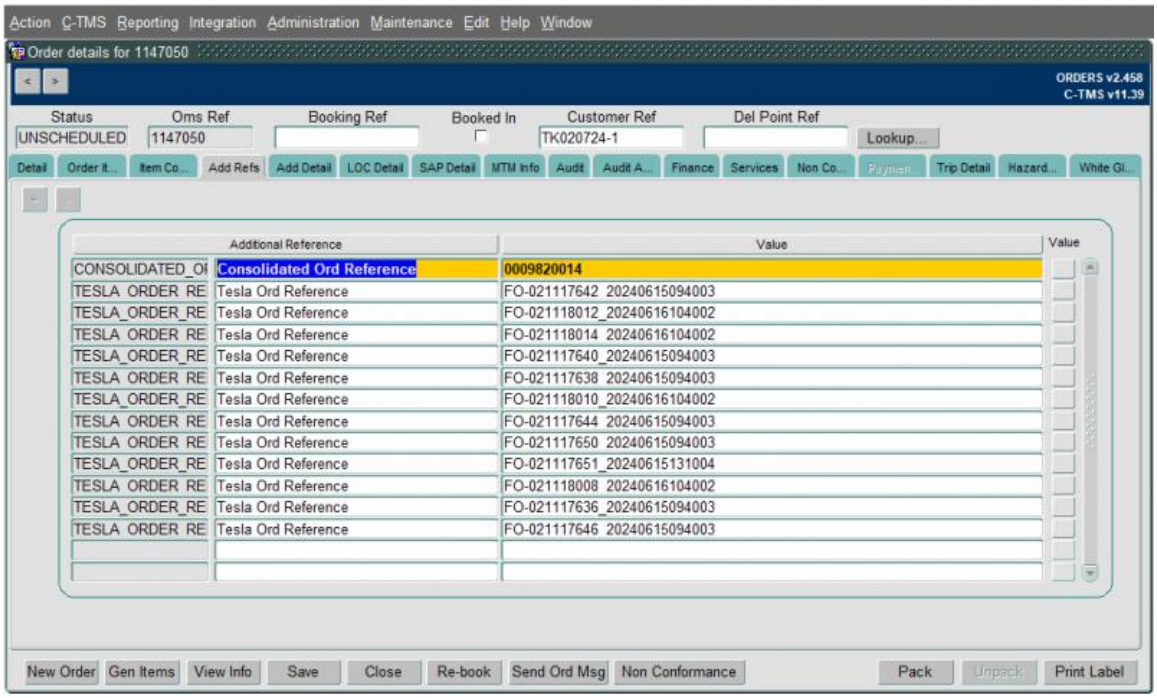
## 24.1 Order Search

Two new items have been added to the Order search in both the orders and new orders screens.




These will allow the user to search for orders containing a Tesla or consolidated order reference. These values are stored in the SCH_ORD_REFRENCES table and can be viewed in the Additional References tab page.

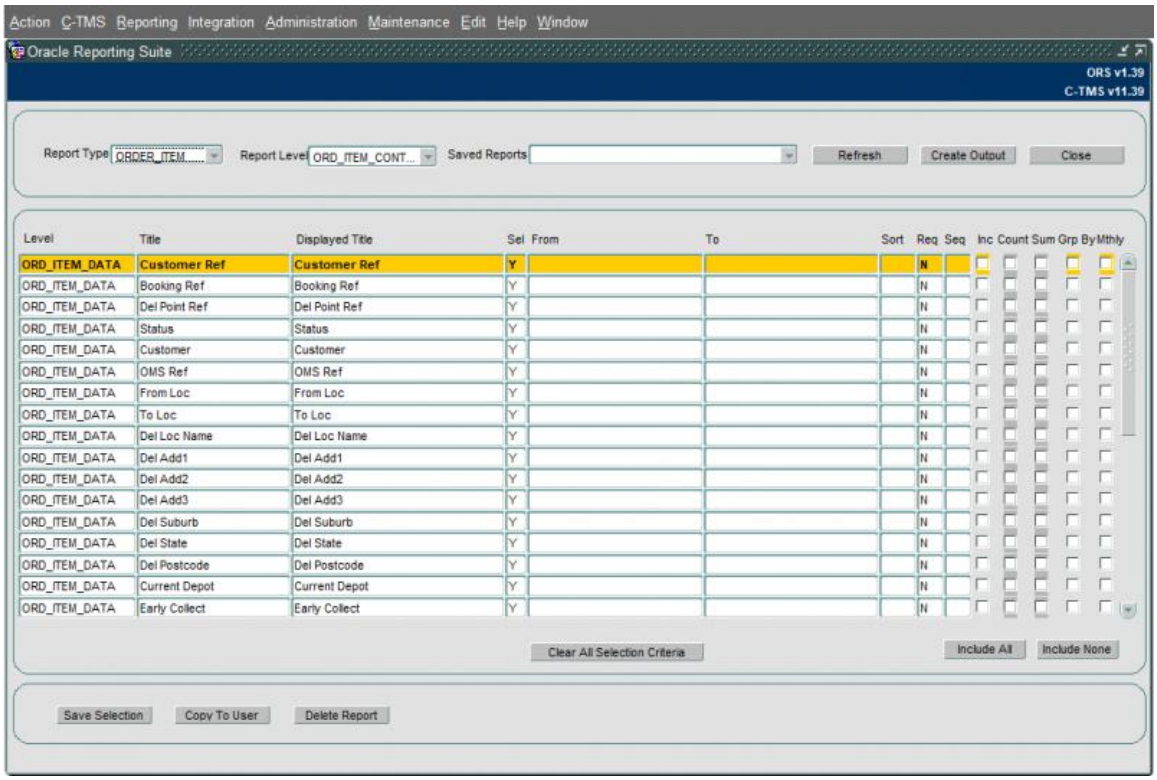
An order may have multiple references




System parameter ALLOW_MULTIPLE_ADD_REFS will allow the user to enter multiple additional references with the same type manually if required.

## 24.2 ORS extracts

The existing ORDER_ITEM ORS extract has been modified to add a new content level which will be used by Tesla full mapping details for the extract can be found in the Functional Spec located on the projects drive here -




## 24.3 Specification

P:\Development\CTMS\DHL\528798 SCR-CTMS-02428249-04 - CTMS - Reports - Tracking Report - Spec

# 25 Category:Support Documents

This category contains all support handover documents. This is naturally also part of the technical guides.





# 26 TCM Container Yard

This page is intended to show some of the technical details for TCM/Terminal Container Management/Container Yard and related WMS tables.

## 26.1 Container Yard Tables

| TABLE_NAME | DESCRIPTION |
|---|---|
| CY_CONTAINER_OWNER | Links to WMS Owner |
| CY_JOB | Jobs |
| CY_JOB_VEHICLE | Vehicles doing a job |
| CY_ACCOUNT | Accounts |
| CY_CONTAINERS | The actual containers |
| CY_MOVES | Every move of containers that has ever taken place |
| CY_SIGNATURES | Driver/haulier signatures (ToC) |

CY_SIGNATURES

| COLUMN_NAME | DATA_TYPE |
|---|---|
| CREATED_DATETIME | DATE |
| SIGNATURE | BLOB |
| SIGNATURE_ID | NUMBER |

CY_MOVES

| COLUMN_NAME | DATA_TYPE |
|---|---|
| ORDER_NO | VARCHAR2 |
| SERVICE | VARCHAR2 |
| CARRIER | VARCHAR2 |
| HAULIER_CODE | VARCHAR2 |
| REGISTRATION | VARCHAR2 |
| TYPE_CODE | VARCHAR2 |
| MOVEMENT_TYPE | VARCHAR2 |
| ACCOUNT_ID | VARCHAR2 |
| CONTAINER_ID | VARCHAR2 |
| WAREHOUSE_ID | VARCHAR2 |
| COMPANY_CODE | VARCHAR2 |
| MOVEMENT_ID | NUMBER |
| EDI_SENT_DATE | DATE |
| WEIGHT | NUMBER |
| SEAL | VARCHAR2 |
| MOVEMENT_BY | VARCHAR2 |
| UPDATE_COUNTER | NUMBER |
| WEIGHT_CODE | VARCHAR2 |
| POSITION_CODE | NUMBER |
| SIGNATURE_ID | NUMBER |
| USER_ID | VARCHAR2 |
| LINKED_REGISTRATION | VARCHAR2 |
| TO_LOCATION | VARCHAR2 |
| FROM_LOCATION | VARCHAR2 |
| ENTERED_DATETIME | DATE |
| DUE_DATETIME | DATE |
| CREATED_DATETIME | DATE |
| MOVEMENT_DATETIME | DATE |
| CONTAINER_STATUS | VARCHAR2 |
| UN_NUMBER | VARCHAR2 |



| COLUMN_NAME | DATA_TYPE |
|---|---|
| FULL_DAMAGED_EMPTY | VARCHAR2 |
| HAZARDOUS_IND | NUMBER |
| STATUS | NUMBER |
| OWNER_CODE | VARCHAR2 |
| JOB_REFERENCE | VARCHAR2 |
| JOB_ID | NUMBER |
| BD_PROCESSED_IND | NUMBER |
| QUANTITY | NUMBER |

CY_JOB_VEHICLE

| | |
|---|---|
| JOB_UPDATE_COUNTER | NUMBER |
| VEHICLE_REGISTRATION | VARCHAR2 |
| JOB_ID | NUMBER |

CY_JOB

| | |
|---|---|
| JOB_UPDATE_COUNTER | NUMBER |
| CREATED | DATE |
| ACCOUNT_ID | VARCHAR2 |
| WAREHOUSE_ID | VARCHAR2 |
| COMPANY_CODE | VARCHAR2 |
| BAY_DIARY_BOOKING_REF | VARCHAR2 |
| JOB_REFERENCE | VARCHAR2 |
| JOB_ID | NUMBER |

CY_CONTAINER_OWNER

| | |
|---|---|
| OWNER_UPDATE_COUNTER | NUMBER |
| OWNER_DESCRIPTION | VARCHAR2 |
| OWNER_CODE | VARCHAR2 |

CY_CONTAINERS

| | |
|---|---|
| INBOUND_METHOD | VARCHAR2 |
| HOLD_STATUS | VARCHAR2 |
| UPDATE_COUNTER | NUMBER |
| LAST_ORDER_NO | VARCHAR2 |
| ONSITE_DATETIME | DATE |
| LOCATION_DATETIME | DATE |
| AMENDED_DATETIME | DATE |
| CREATED_DATETIME | DATE |
| UN_NUMBER | VARCHAR2 |
| FULL_DAMAGED_EMPTY | VARCHAR2 |
| HAZARDOUS_IND | NUMBER |
| STATUS_CODE | VARCHAR2 |
| LOCATION_SEQ | NUMBER |
| LOCATION_CODE | VARCHAR2 |
| WEIGHT_CODE | VARCHAR2 |
| TYPE_CODE | VARCHAR2 |
| ACCOUNT_ID | VARCHAR2 |
| CONTAINER_ID | VARCHAR2 |
| WAREHOUSE_ID | VARCHAR2 |
| COMPANY_CODE | VARCHAR2 |
| OWNER_CODE | VARCHAR2 |
| WEIGHT | NUMBER |
| SEAL | VARCHAR2 |

CY_ACCOUNT



| | |
|---|---|
| CY_ENABLED | VARCHAR2 |
| CY_CARRIER | VARCHAR2 |
| CY_SERVICE | VARCHAR2 |
| STOCKIST_SUB_CODE | VARCHAR2 |
| STOCKIST_CODE | VARCHAR2 |

## 26.2 Additional Related WMS Tables

Static:

| TABLE_NAME | DESCRIPTION |
|---|---|
| WARE_LOCATION | Locations in the yard WHERE LOC_TYPE = 'Y' |
| CONTAINER_TYPE | Types of containers |
| STK_STOCKIST | Links to Account |
| HAULIERS | Hauliers for movements into and out of the yard. |
| CONTAINER_STATUS | lookup of statuses |
| TRANSLATION_TABLE | CWGHT, store weights of 20ft containers |
| FXM_USERS/ | |
| FXM_USER_COMP_WAREHOUSES | For logon purposes |
| FLEX_USER_COMP_OWNERS | |
| VOYAGES | |
| VESSELS | |
| CARRIERS | |
| HAZARDS | |
| WAREHOUSE_RULES | |
| WARE_PARAMS | |

Standard Transactional WMS tables:

| TABLE_NAME | DESCRIPTION |
|---|---|
| WARE_GOODS_ENTRY_HEADERS | CONTAINER_ID |
| MANIFEST_CONTAINER_HEADER | CONTAINER_NO |
| ORDER_HEADER | CONTAINER_NO |

## 26.3 Views

These views broadly contain all of the cross-reference data from other related tables such as those listed above from standard WMS and CY tables.

| VIEW_NAME | Notes |
|---|---|
| V_CY_CONTAINERS_DETAILED | |
| V_CY_JOB_BOOKING_REF | This links to data from Bay Diary (WARE_DIARY_HEADERS) to get the booking reference.<br><br>Used in V_CY_WORK_IN_PROGRESS below |
| V_CY_MOVES_DETAILED | |
| V_CY_WORK_IN_PROGRESS | |
| V_CY_WORK_IN_PROGRESS_DET | Used in V_CY_WORK_IN_PROGRESS above - Links CY_MOVES to Booking Reference, and unions on all types of movements (inbound/outbound gate, vehicle, vessel, etc). |

## 26.4 Example Queries

```
select * from cy_moves cm
where registration = 'EXP1503271200'

select * from cy_moves cm
where registration = 'EXP1802021102'

select * from cy_moves cm
where movement_type = 'OR'
and company_code = '997'
```





```
select * from cy_containers c
--where container_id like 'MSCU%'
where container_id = 'MSCU3951530'
for update

Select * from cy_account

SELECT * FROM WARE_LOCATION
WHERE LOC_TYPE = 'Y'
```



# 27 Tesla Orders EDI

Tesla EDI is a DHL AA interface, which is a pass-through from DHL Link from SAP.

## 27.1 Content

The content is XML from SAP, containing

- Pallets - a collection of pallet nodes with
  - Orders
  - Boxes
  - Order Lines
- Orders - a collection of order nodes with
  - Order_lines and contained tags.

Sample file:

- File:Order confirmation from seven CP23-0087137.txt

## 27.2 Process

Package DP_TESLA_EDI_IN

The IMPORT_ORDERS imports the filed.

PROCESS_IMPORT process

Stores in files:

- TESLA_EDI_ORDER_HEADER - mainly derived from /pallet/orders/order
- TESLA_EDI_ORDER_DETAILS - mainly derived from the box node

These are run through several times in order to turn the input the right way up (the import is in two sections, and the boxes lists the boxes, followed bu the orders and lines, and finally the pallet).

f_process_order finds the order from the details provided and

This process uses EDI parameters:

- DEFAULT_DU
- DEFAULT_PROD
- ALLOW_REUSABLE_ASSET

This process uses decodes:

- TESLA_DELAERSHIPS
- TESLA_DU_TYPES

The process

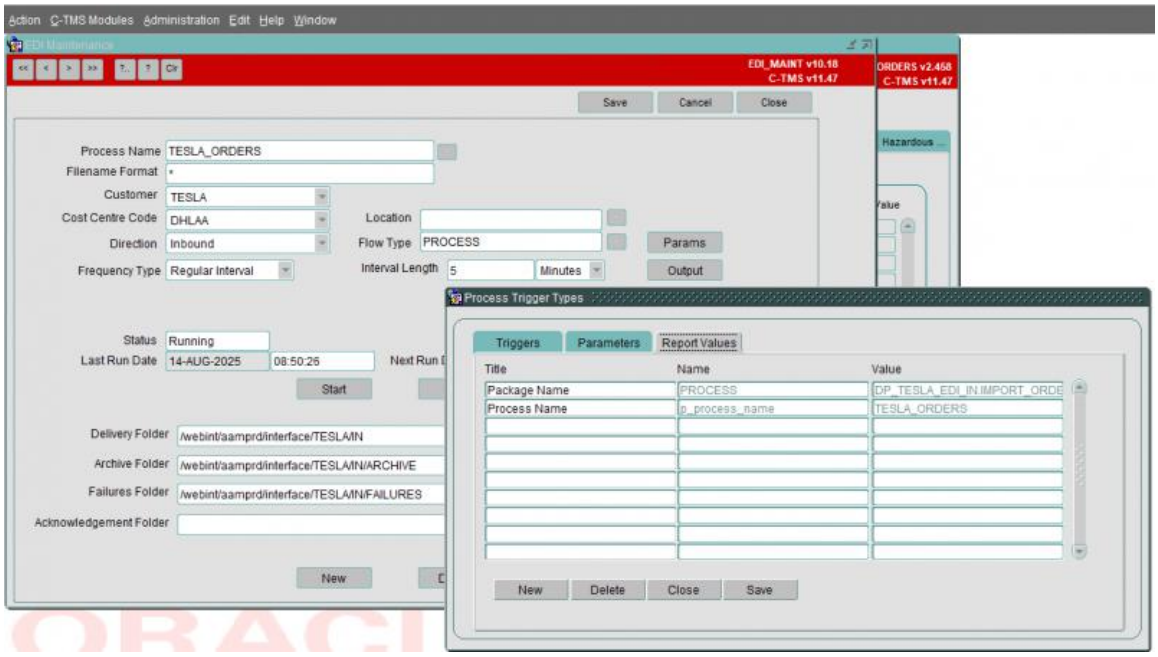
- Finds the schedule from the del date
- Finds the order using the dealership location stored, the schedule found and the EDI process' customer and cost centre
- Inserts the lines, items and contents records.
- Inserts order references
  - CONSOLIDATED_ORDER_REFERENCE - one of
  - MH_ORDER_REF - one of
  - TESLA_ORDER_REFERENCE - many, derived from the order lines

## 27.3 Implementation

Create an EDI








- Flow Type: PROCESS
- Parameters
    - ♦ Report Values
        - ◊ Package Name - PROCESS - DP_TESLA_EDI_IN.IMPORT_ORDERS
        - ◊ Process Name - p_process_name - TESLA_ORDERS
    - ♦ Parameters
        - ◊ ALLOW_REUSABLE_ASSET
        - ◊ DEFAULT_DU
        - ◊ DEFAULT_PROD

## 27.4 Management

Tesla Orders EDI files can be managed through the Tesla Orders tab on the Interface Errors screen.

You can search using the header fields:

- Include Success - a checkbox - by default the screen only includes failures.
- All other criteria are drop-down lists:
    - Record Status
    - Dealership
    - Case Id
    - Order Number
    - Delivery Date

The screen displays:

- Filename
- Status - S or F
- Case Id
- Dealer
- Del Date
- Order No
- Carton Type
- Carton Code
- OMS
- Created Date

You can sort the results by any of these columns.

Select a record on this results table and further information will be shown below:

- Error message - any associated errors whilst processing the file
- Pallet/Case Details:
    - Tesla Order Number
    - Line Number
    - Item Number
    - Item Qty
    - Hazardous Material - indicator whether the material is hazardous.

If the record is failed, you can reprocess it with the **Re-Process** button.



# 28 WCS Build and Release Process

The objective of this document is to clearly describe the steps required to build and release programs to the client?s site for an update to Calidus 3pl-Mobile (the WCS).

## 28.1 Scope

This process applies to all sites that use Calidus 3pl-Mobile and require releases, but do not utilise an automated release mechanism.

Where the document references updating the Supimix call logging system to particular statuses, the standard Supimix updating procedure applies.

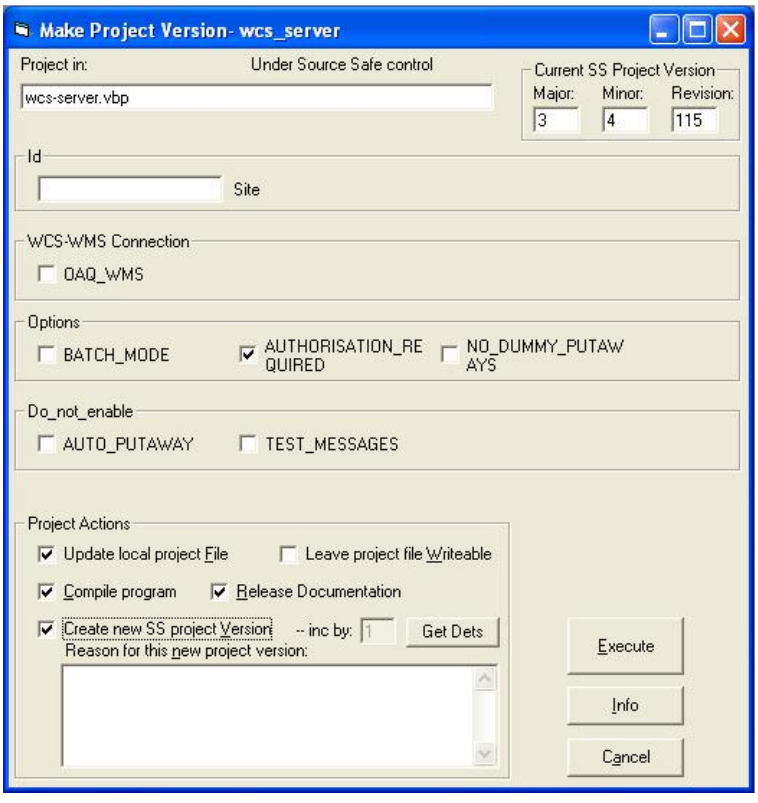
### 28.1.1 Responsibility

It is the responsibility of the person completing the testing that the programs be released out to appropriate representatives.

## 28.2 Building the programs

Once the changes for a particular log/number of logs are completed, the affected programs are built on the test server, using a utility for the purpose. The utility, Version Maker, is run directly from the programming environment. The interface presented depends on the program being built:

WCS:

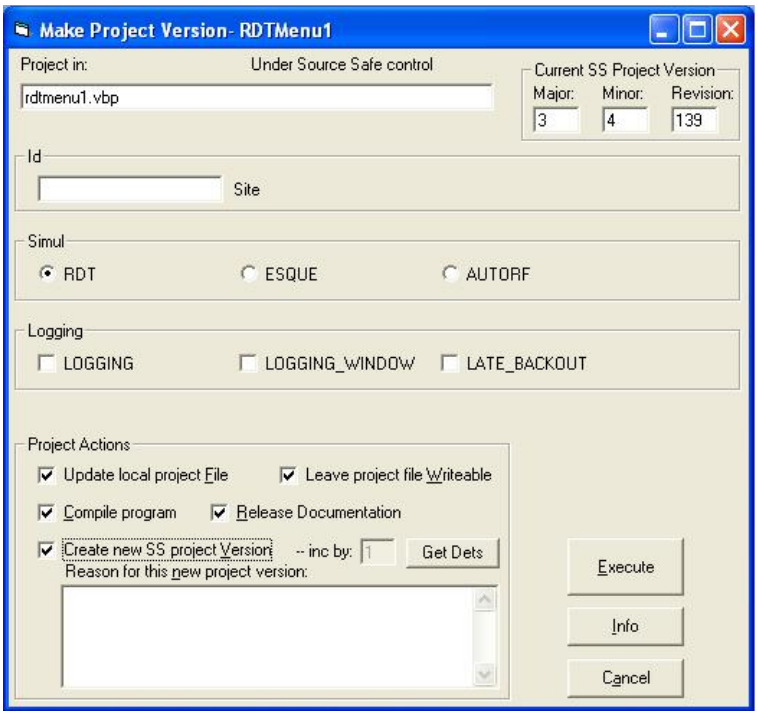



The screen shot shows the settings needed for a build of WCS-Server.exe. To build WCS-Server_OAQ repeat the call with OAQ_WMS ticked (after saving version number change to the .vbp file). Normally the same version number is

wanted for both so ?Create new SS project version? is un-ticked on the OAQ call.

The build is normally preceded by a ?Get Dets? (see button above). This extracts from VSS details of the changes included in the build (delete any duplicates at this stage). This information is automatically written to the release documentation.

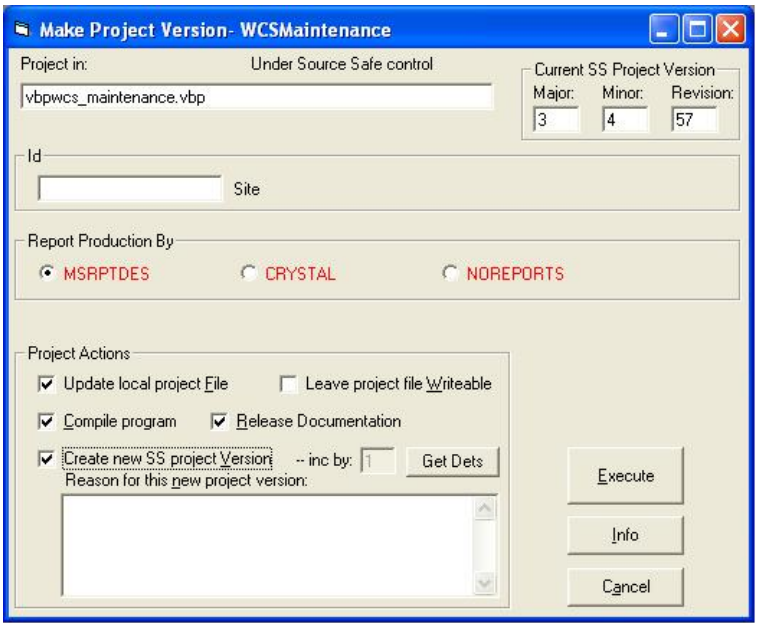
RDT:




The screen shot shows the settings required to build RDTMenu1.exe. To build Debug.exe click ESQUE and tick LOGGING and LATE_BACKOUT Normally the same version number is wanted for both so ?Create new SS project version? is un-ticked on the Debug call.

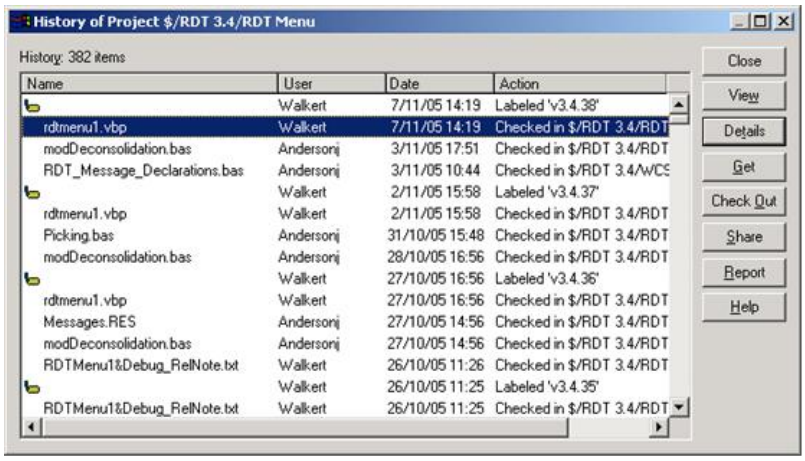
The build is normally preceded by a ?Get Dets? (see button above). This extracts from VSS details of the changes included in the build. This information is automatically written to the release documentation.

WCS Maintenance:

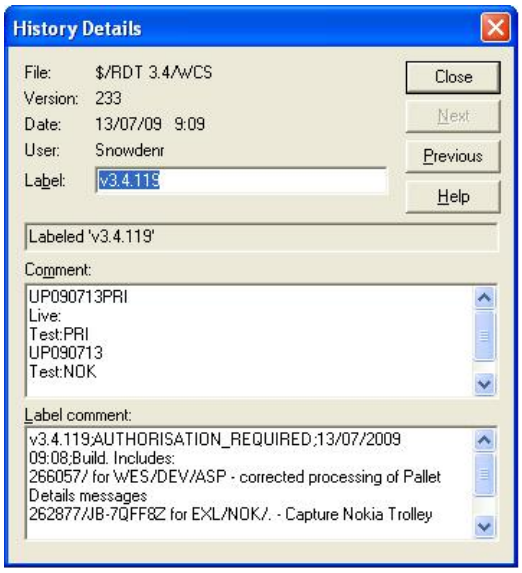



The build is normally preceded by a ?Get Dets? (see button above). This extracts from VSS details of the changes included in the build. This information is automatically written to the release documentation

Creating a new SS project version automatically labels the release in the source control environment.




Each label contains the build description, plus the parameters used to create the build:




The comment should be updated when the build is sent to the client, showing the sites to which the program has been released.

The programs are then moved to a test area and tested.

Update the Status of the Supimix call to ?Test?.

The test will be on a defined test machine, linking to a suitable Calidus-3pl test environment.

Once tested, A Proof of Testing document will have been created in the standard Test Logs directory, and the Supimix call will have been updated to Tested.

## 28.3 Building the Release

Once the build has been approved for release, all the programs concerned are moved to a releases area.

This releases area is in the network area:

`P:\RDT\Development\_Installers`

Under this area, there is a "Releases In Progress" area - the programs should be copied in here while they are being put together in a release.

**Note:** If the database, Server or RDT process has changed, all three must be released at the same time, to prevent potential issues when a client doesn?t release all available patches.

A release notes file is also created, an example of which is listed in Appendix A

It is recommended that the "Changes since last build" section is created by copy and pasting the release notes generated automatically by Version Maker. These will probably contain duplicate lines which should be deleted.

The programs to be released and the release notes should now be included into a zip file, named as follows:

`UP<YYMMDD>{<CLIENT>}{<SEQUENCE>}.zip`

`CLIENT` is optional and should only be used if the release is to a specific client or site only. This is the site from the Supimix call (e.g. CHE, CHE3). Upgrades should only be released to specific clients if:

- The problem requires immediate fixing and it is impractical to upgrade to the latest version, or;





- The site is on a bespoke version of the WCS

SEQUENCE is optional, used only if several releases are made on the same day.

The naming convention for release notes is <RELEASE> relnotes.txt.

When the patch is built, update the status of the Supimix call to 'Ready for Release'. Update the Patch field with the name of the patch file.

## 28.4 Releasing

**Note:** The following is a standard release process that should be followed for all new clients. There are some existing clients that require the patch to be released via email.

For a list of current clients and their release mechanisms, see Appendix B

FTP the patch out to clients? identified machine. Existing clients have shortcuts to release to specific machines.

Send an email to the general WCS list and the defined representatives of the specific client, and copy it to the members of the OBS WCS team. The email should identify the release name, and the place the release can be found, and include a copy of the release note.

Copy the release and release note to the client area in the releases area

```
P:\RDT\Development\_Installers
```

Under this area there are specific company areas. For all new installations, this will be the company from the call (e.g. EXL)

Under this area, the site or client is specified. For all new installations, this will be the site from the Supimix call (e.g. CHE, CHE3)

If the release is for all clients in a company (and the company has several sites), the release can be placed in a non-site-specific directory, usually above the site-specific areas. For example:

```
P:\RDT\Development\_Installers
\EXL                    - Company area
\_Sites              - Releases for all sites
\COV            | Site areas
\CHE            |
\CN-ATH         |
```

Update the SourceSafe label to show that the release has been made and which sites have it in test.

Once this is complete, update the status of the Supimix call to ?Acceptance Test? (5), sub-status ?Complete? (70).

For instructions on releasing patches directly onto client servers, please see the C3PL-M Installation Guide, referenced as item 1 in Appendix C





## 28.5 Sample Release Note

See also Creating a WCS Patch

Files included in release, plus versions:

| Program | Version |
|---|---|
| rdt1_struct.mdb | ss v107 |
| RDTMenu1.exe | 3.4.30 |
| Debug.exe | 3.4.30 |
| WCS-Server.exe | 3.4.23 |
| WCSMaintenance.exe | 3.4.15 |

Changes since last build:

| Log Number | Programs | Description |
|---|---|---|
| 198788 | RDTMenu1/ WCS-Server/ WCS Maintenance | Deconsolidation and Despatch development |
| 206084 | WCS-Server | Fix to creation of picking containers when short picking; fix to receiving stock drip feed messages. |

Installation Instructions:

Stop all RDTs running on the system.

Stop the WCS server on the TEST system.

Stop all WCS Maintenance sessions accessing the database

Make a backup copy of the old program, if required.

Copy the released versions of this program to the correct area on the TEST machine,

normally C:\Program Files\Warehouse Control Server

If the database structure has been released to you, the database should be converted as described in the WCS Installation guide.

Restart the programs.

Note: There are WMS portions of some of these changes. Please check WebRelease for:

198780/ML-6C2EZD

198790/ML-6C9BKD

198776/ML-6C2DWD