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.

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 Browserless Appliction Creation

Note: For internal use only

2.1 Introduction

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

2.2 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.

2.3 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**

2.4 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.

3 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

4 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.

5 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
```


6 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.

6.1 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.

6.2 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

7 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

8 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.

8.1 Contents

8.2 Configuration

8.2.1 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

8.2.2 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.

8.2.3 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

8.2.4 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

8.3 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.

8.4 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.

9 Technical Information

9.1 Packages

- DP_PAR_API
- DP_PAR_API_STS

9.2 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
```


9.3 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
```

9.4 Tenant Keys

Keys matching dates/schedules to Paragon run keys

```
SELECT * FROM PAR_KEYS

SELECT * FROM PAR_KEYS_ROUTE
```

9.5 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.
```

9.6 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';
```


10 DHL Invoicing

11 DHL Invoicing

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

12 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.

13 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

14 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

14.1 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

15 Ford Orders EDI

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

15.1 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

15.2 Process

Package DP_FORD_EDI_IN

Process_file_orig - presumably old.

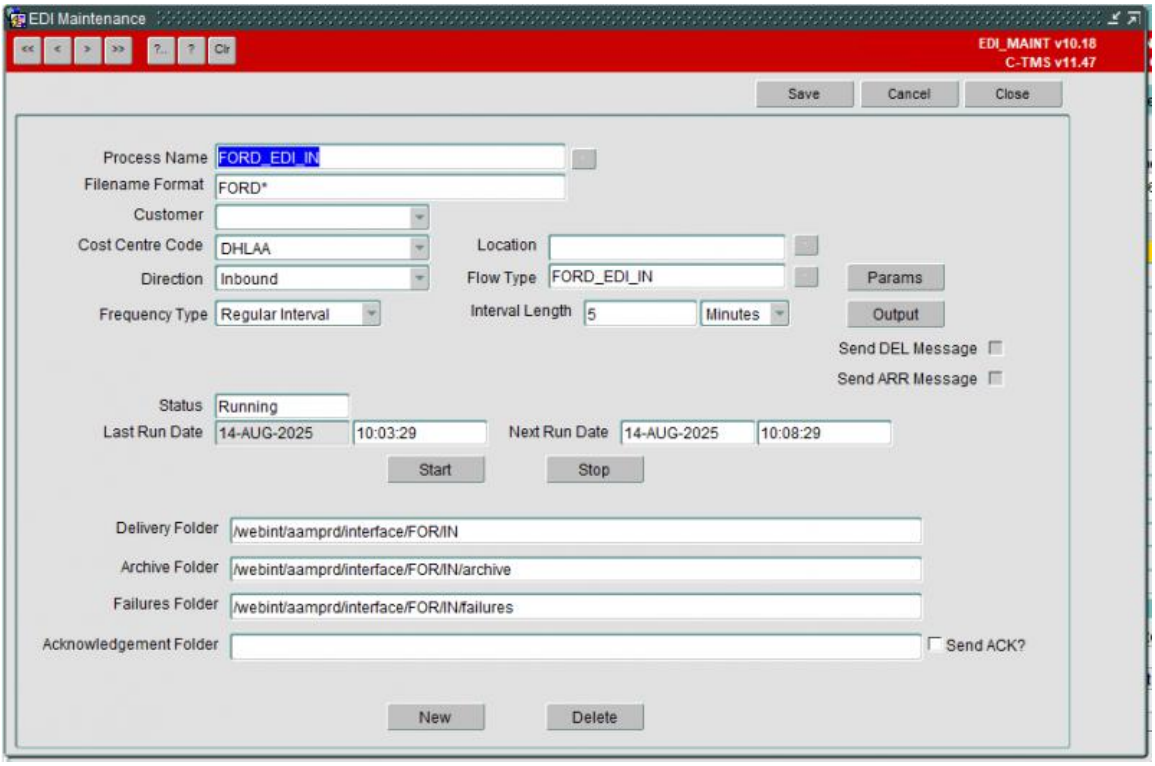
Process_file - presumably new

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

15.3 Implementation

Create an EDI



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

15.4 Management

Warning: Unknown

16 Support - Tesla EDI Inbound Guide

16.1 Order Template

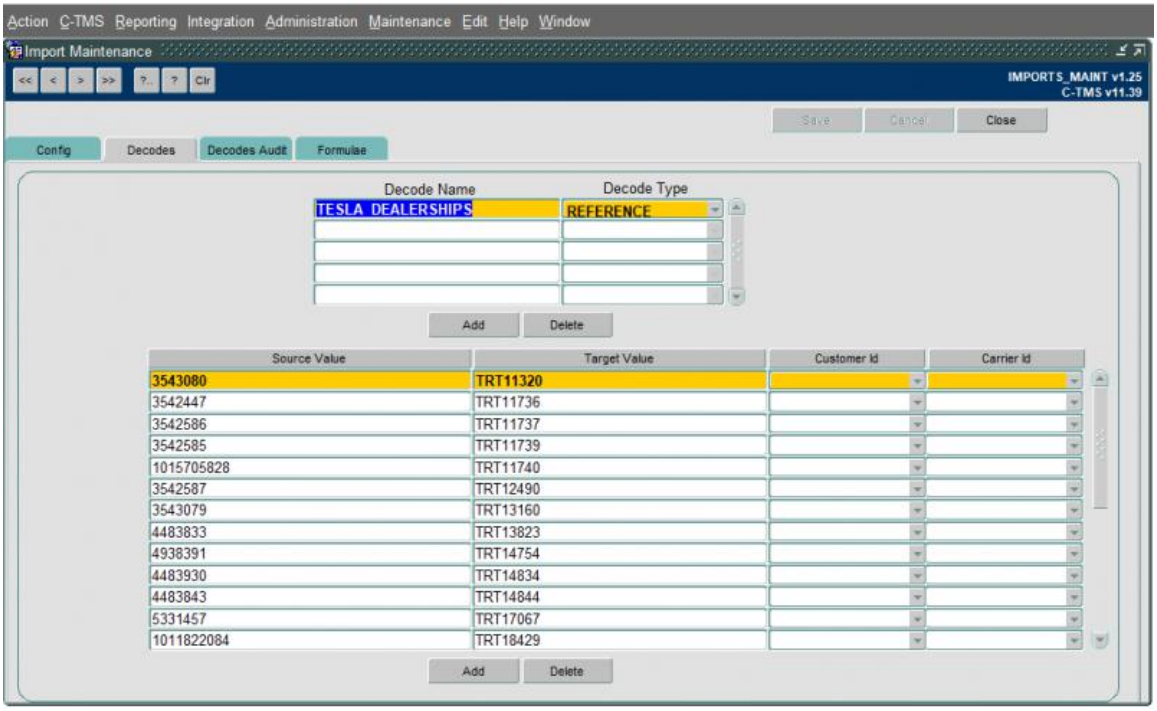
For each dealership an order will be created each day, this is a placeholder which will be added to by the import process. This is existing fixed schedules functionality which has not been modified for Tesla.

16.2 System parameters

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

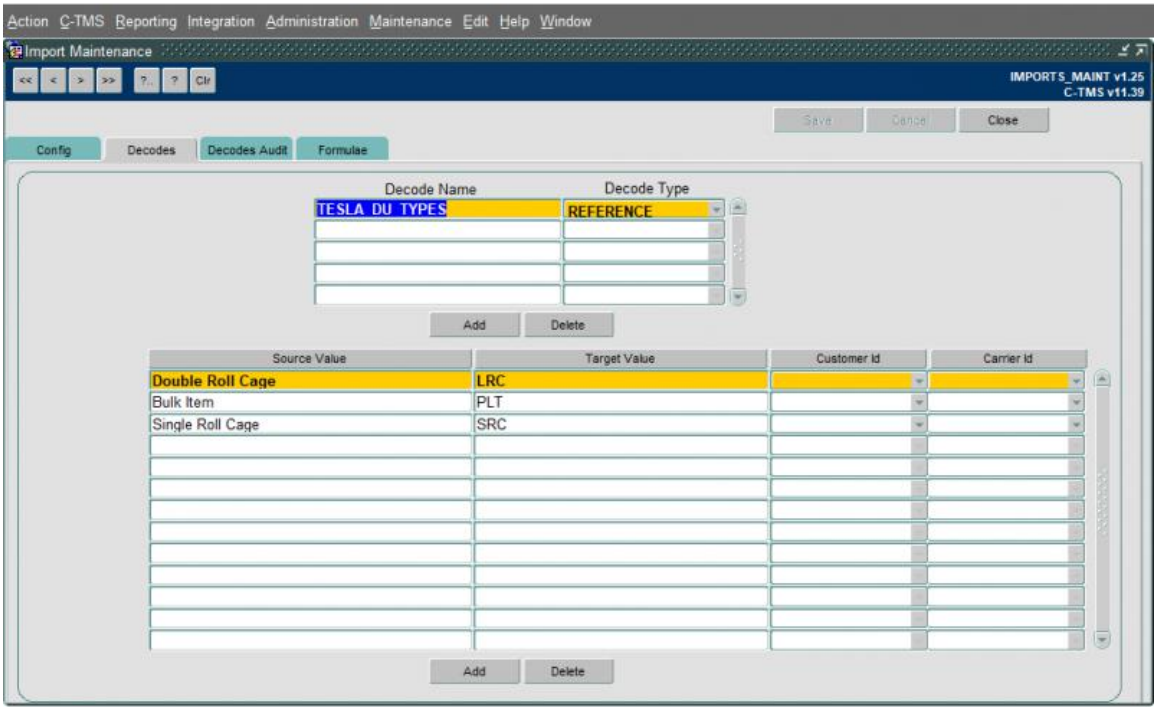
16.3 Dealership Decodes

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



16.4 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



16.5 Inbound Files

An example inbound file is included below

16.6 EDI Tables

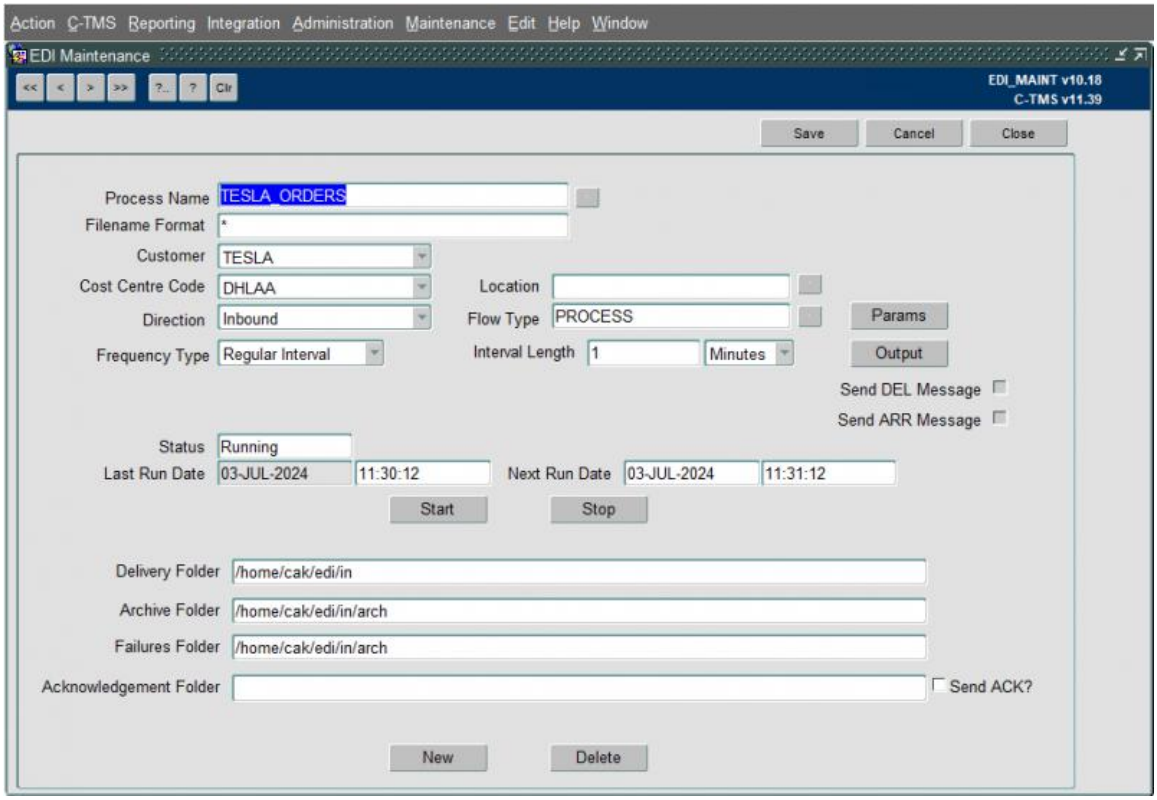
Two new tables have been created to hold the details of the Inbound files

- 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.

16.7 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.

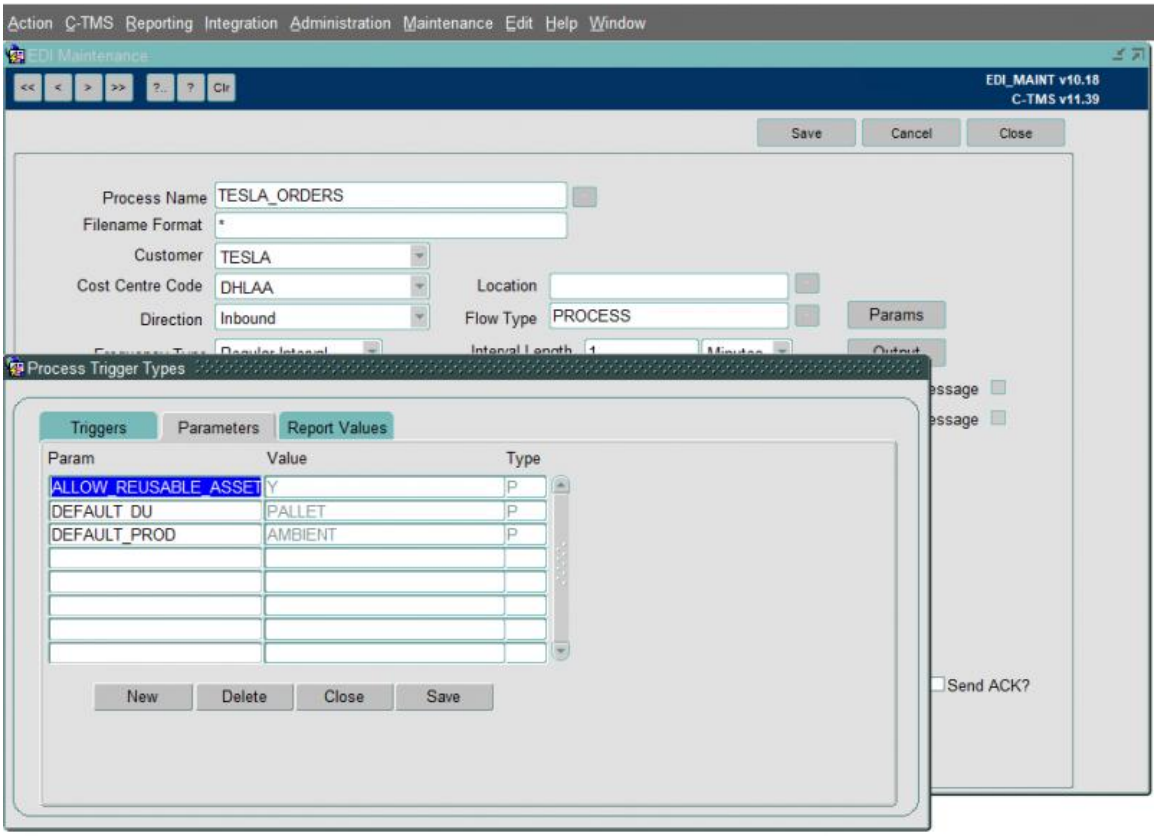


16.8 File Format

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.

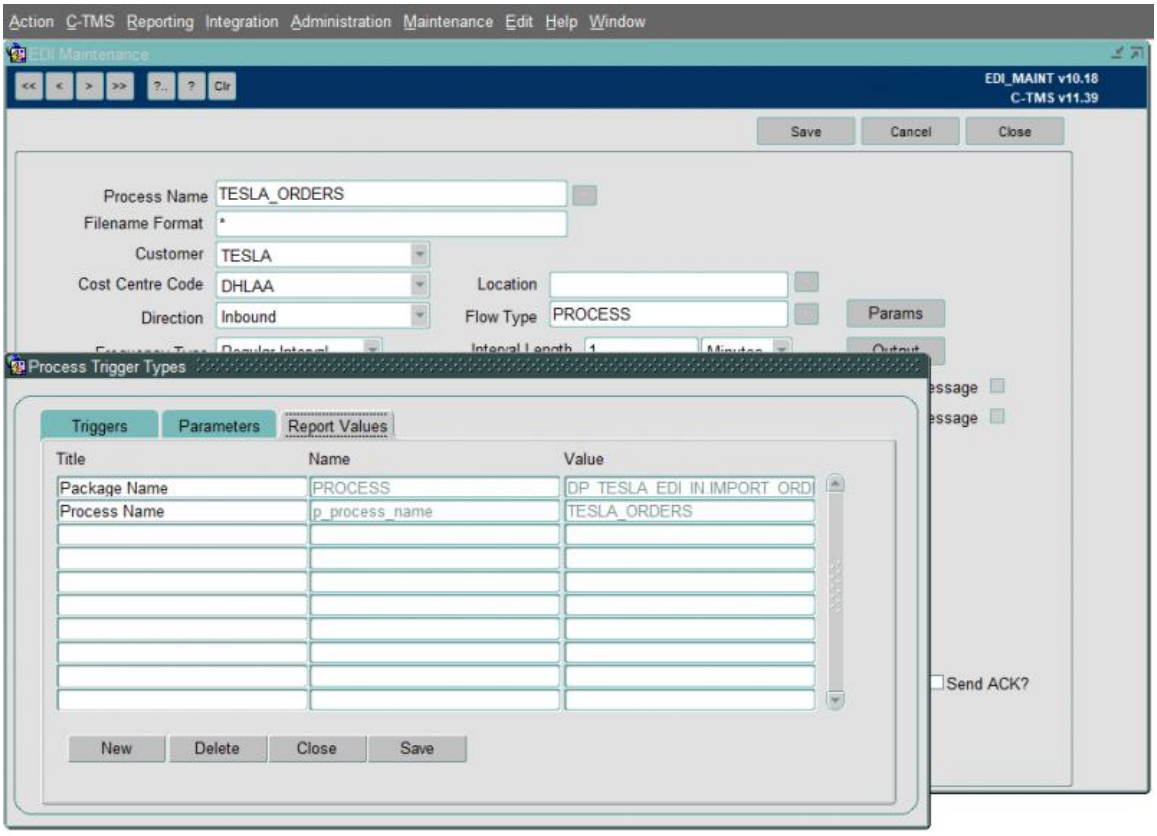
16.9 EDI Parameters

The following Parameters are required



16.10 EDI Report Values

The following Report values are required

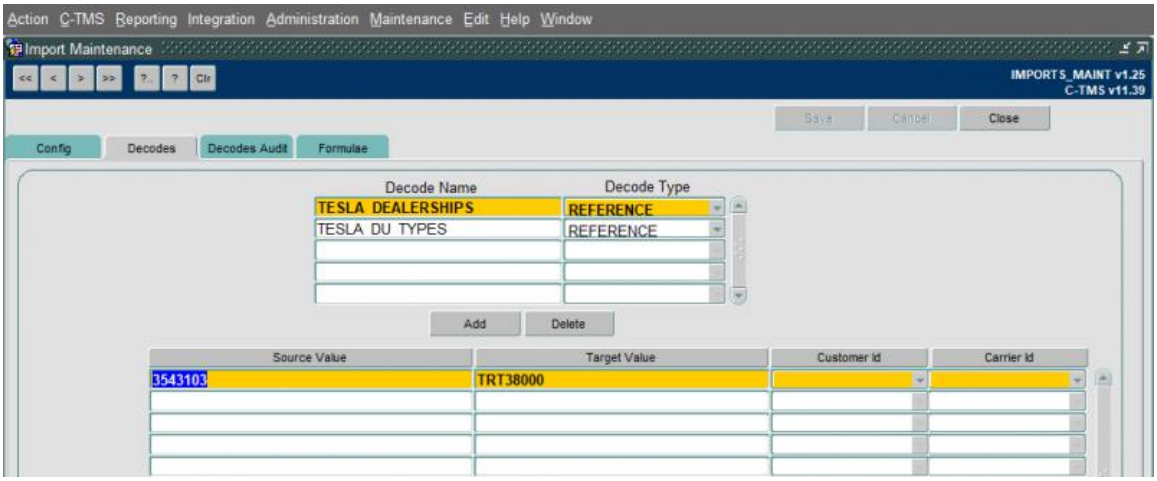


16.11 Order Processing

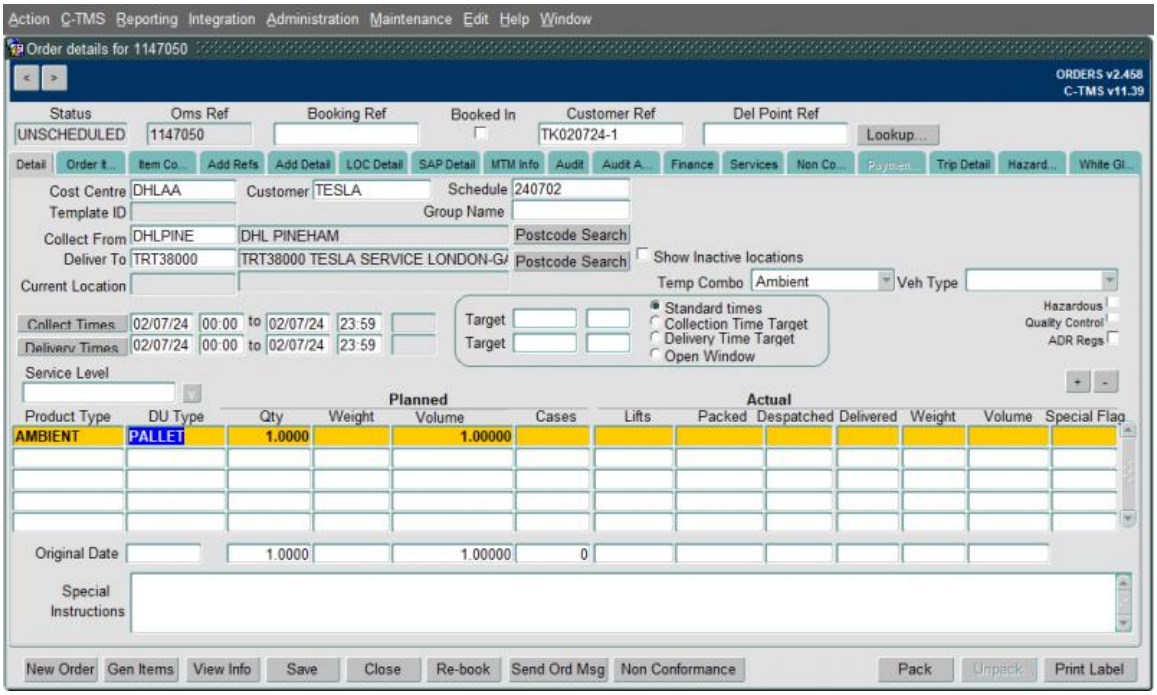
An order will exist on the correct schedule for each dealership in the inbound file the delivery date can be found in the ?deliveryStartDateTime? tag and the dealership in the ?referenceField7? tag

```
<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.

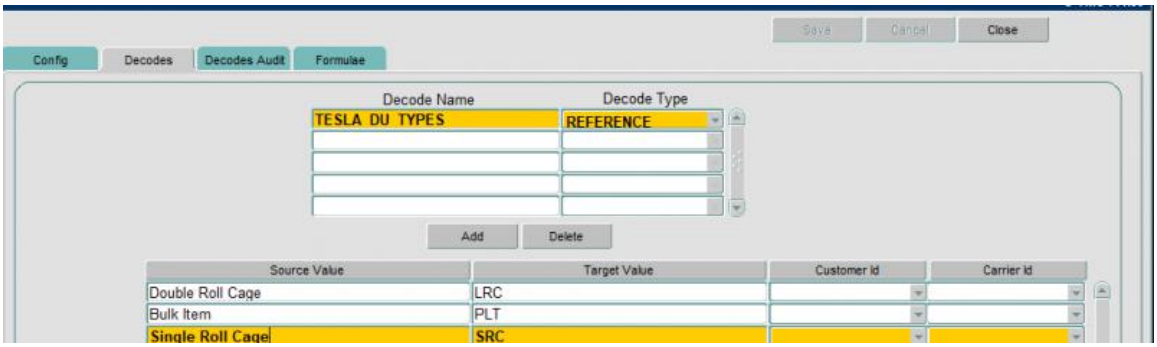


The product type is not contained in the inbound file so therefore must exist as a default parameter on the EDI flow. If no default is present an error will be raised and the order will not process.

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>
```

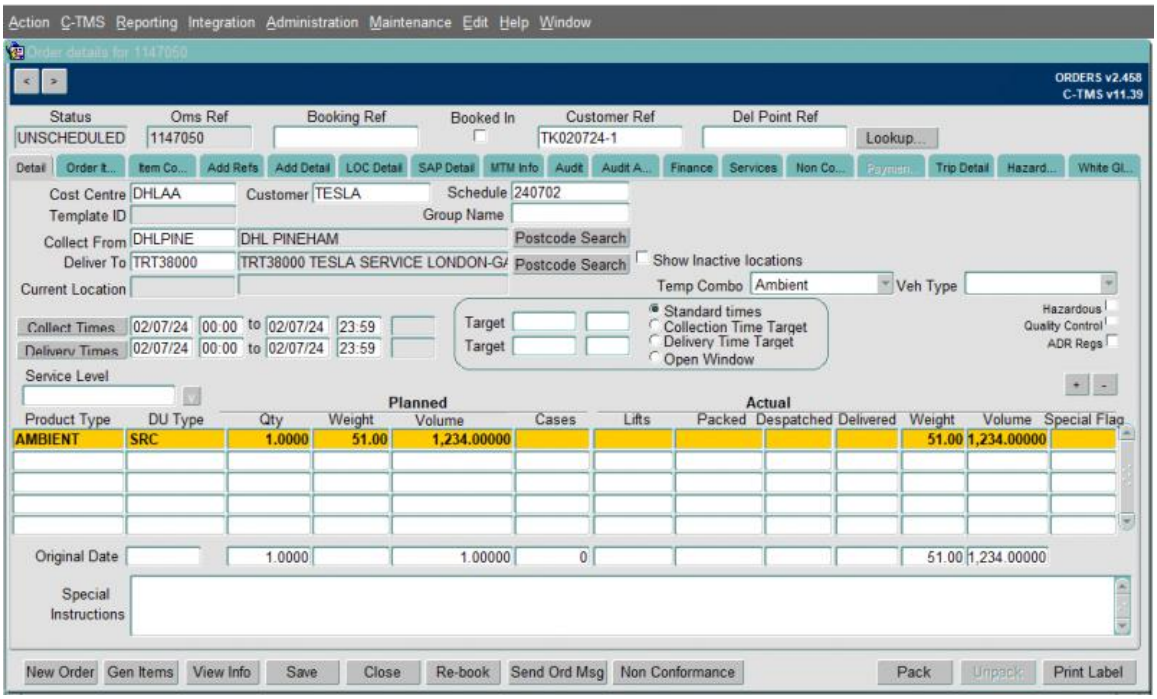
Maps to



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.

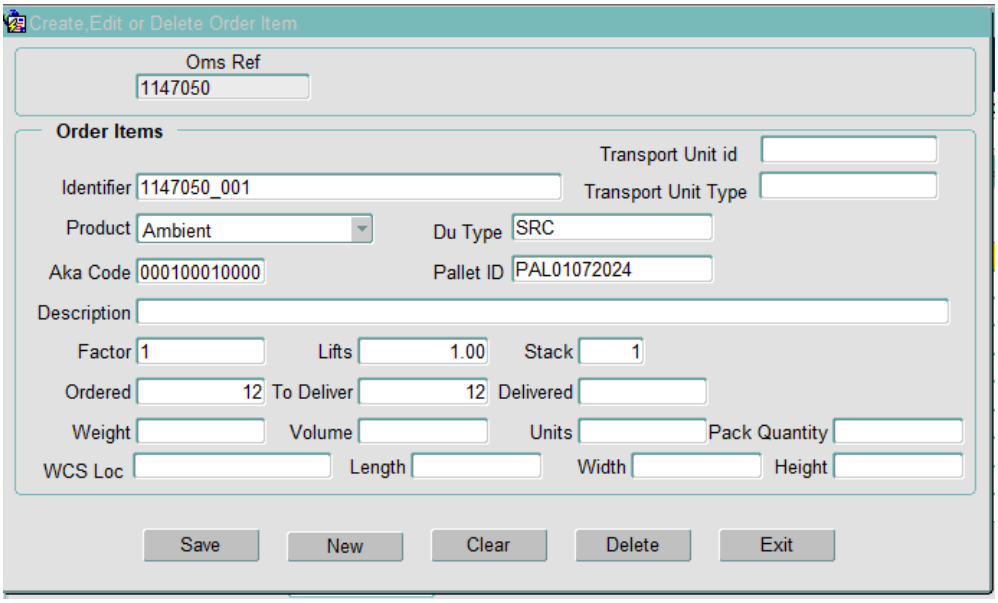
If the DU type doesn?t exist on the current order a new line will be created for that DU type, the placeholder line will be removed at this point if no items exist with the same du type.



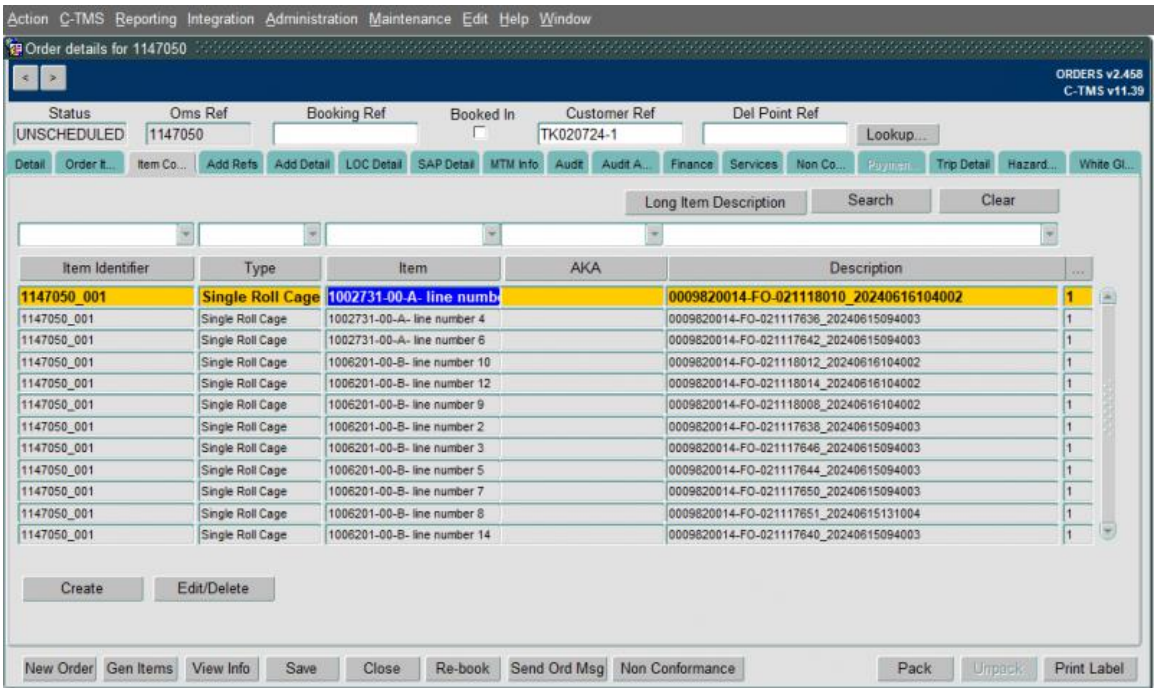
Note the weight and volume on the line these are taken directly from the inbound pallet section and should not be manipulated.

```
<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>
```

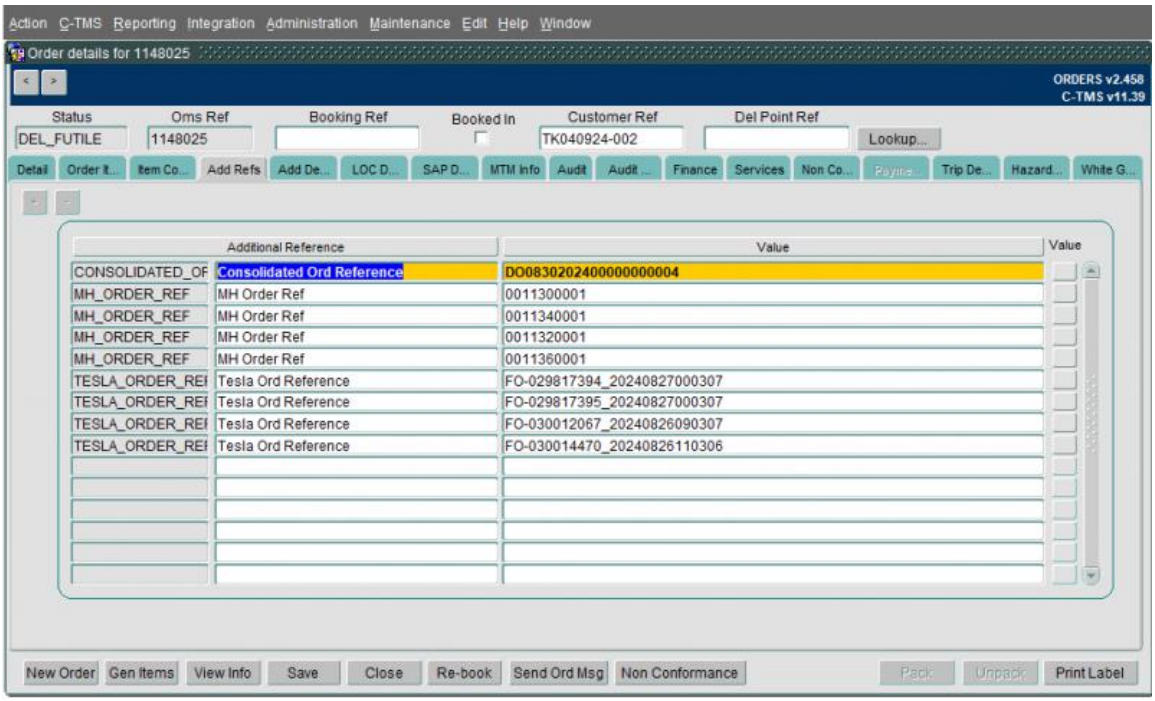
Order items will have been created



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

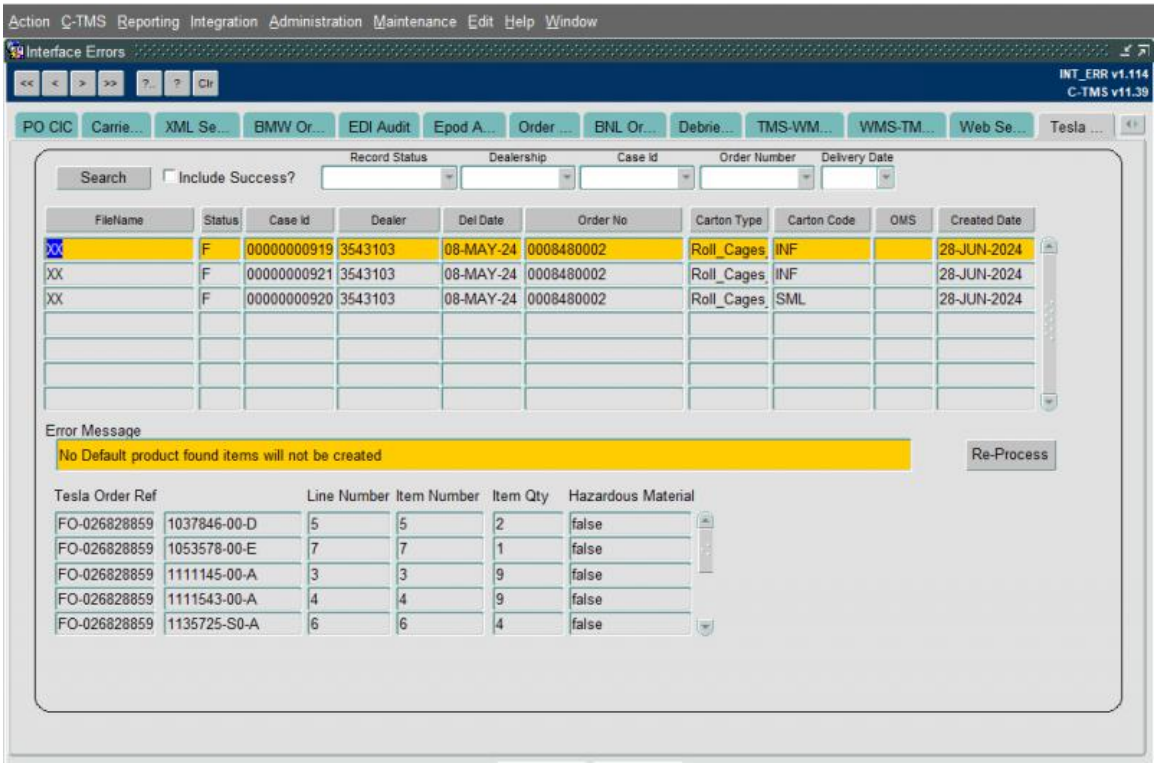
```
<order>
<orderNumber>0011320001</orderNumber>
<aggregatedOrderNumber>DO08302024000000000004</aggregatedOrderNumber>
<carrierName>DSC TTN</carrierName>
<carrierServiceLevel>STD</carrierServiceLevel>
<referenceField6>VIN</referenceField6>
<referenceField7>1008612153</referenceField7>
```

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>
```

16.12 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.



16.13 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

17 Support - Tesla EDI Outbound Guide

17.1 Order Tracking

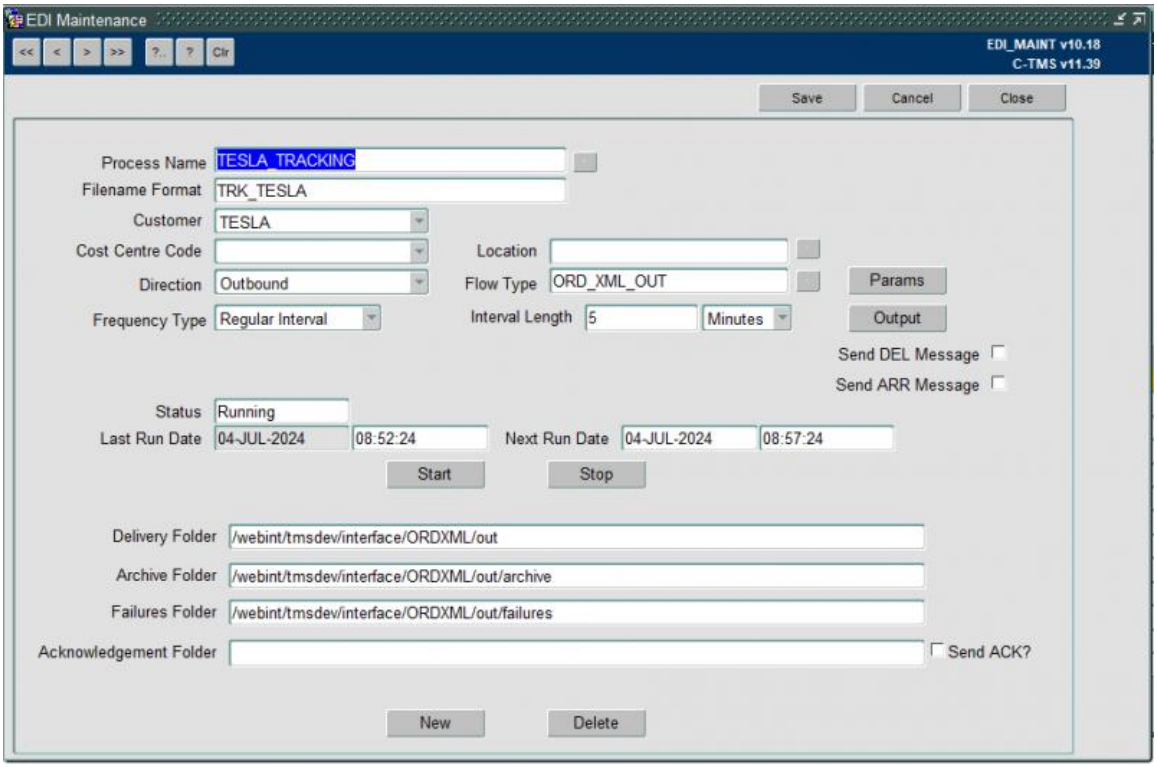
A customer specific tracking interface is required to track the milestones associated with an order.

17.2 System parameters

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

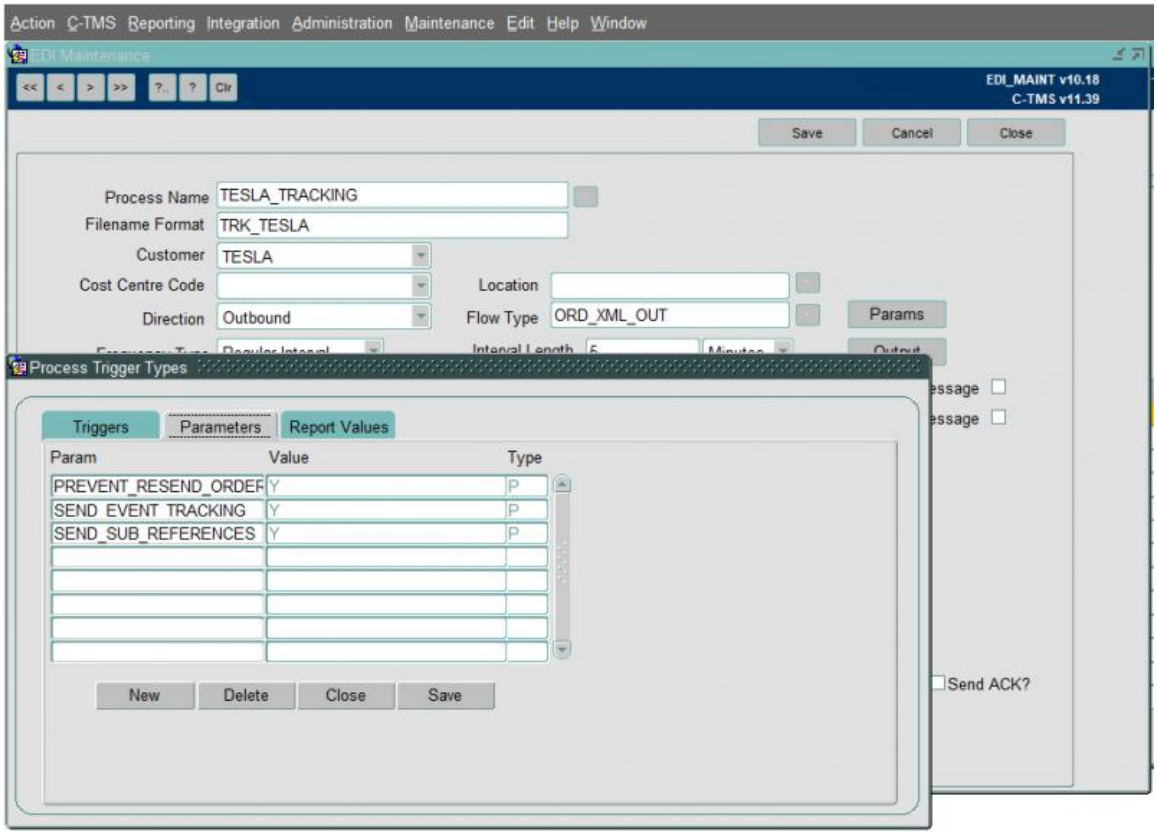
17.3 Outbound EDI

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.



17.4 EDI Parameters

The following Parameters are required



17.5 Messages sent

The following message types will be sent

- ORD
- CAN
- DEP
- ARR
- DEL

17.6 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?.

17.7 Specifications

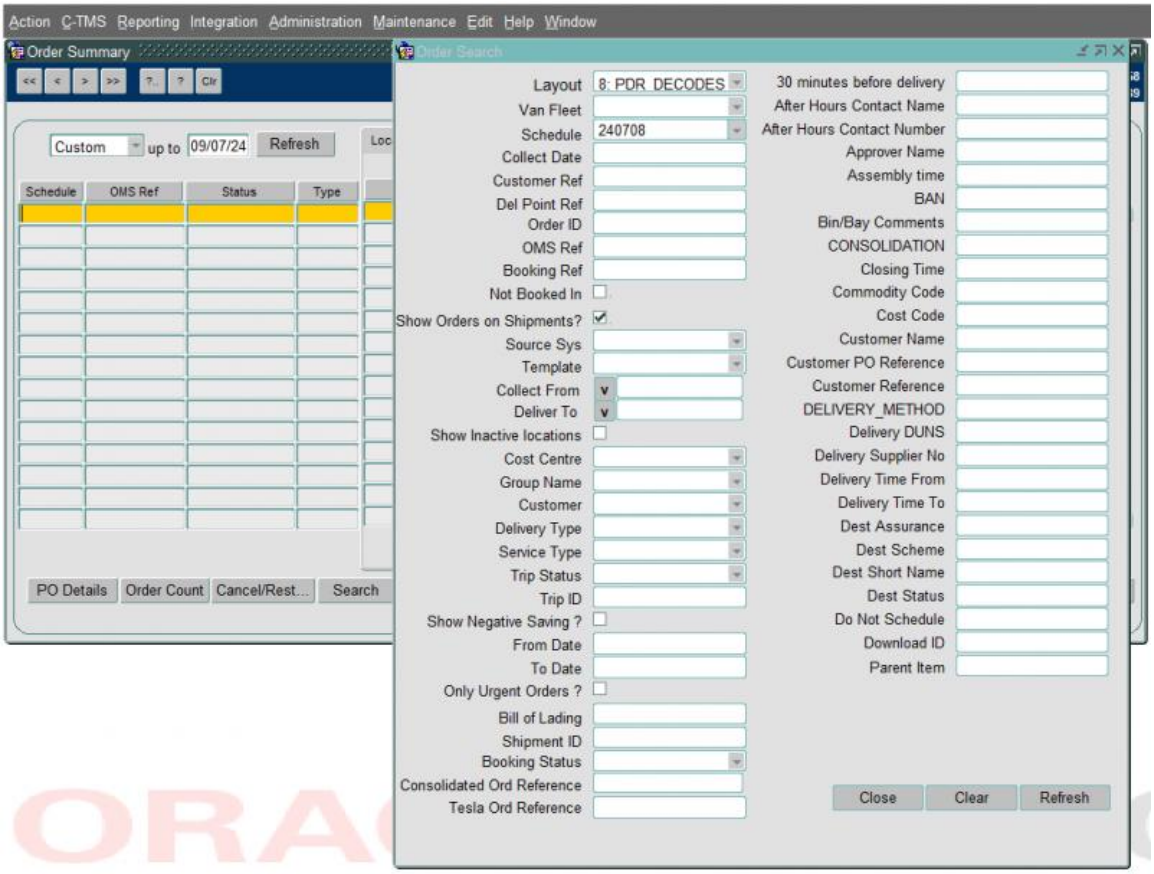
P:\Development\CTMS\DHL\545095 - SCR-CTMS-02428249-05 - CTMS - Customer-Specific Tracking File

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

18 Support - Tesla Order Search Guide

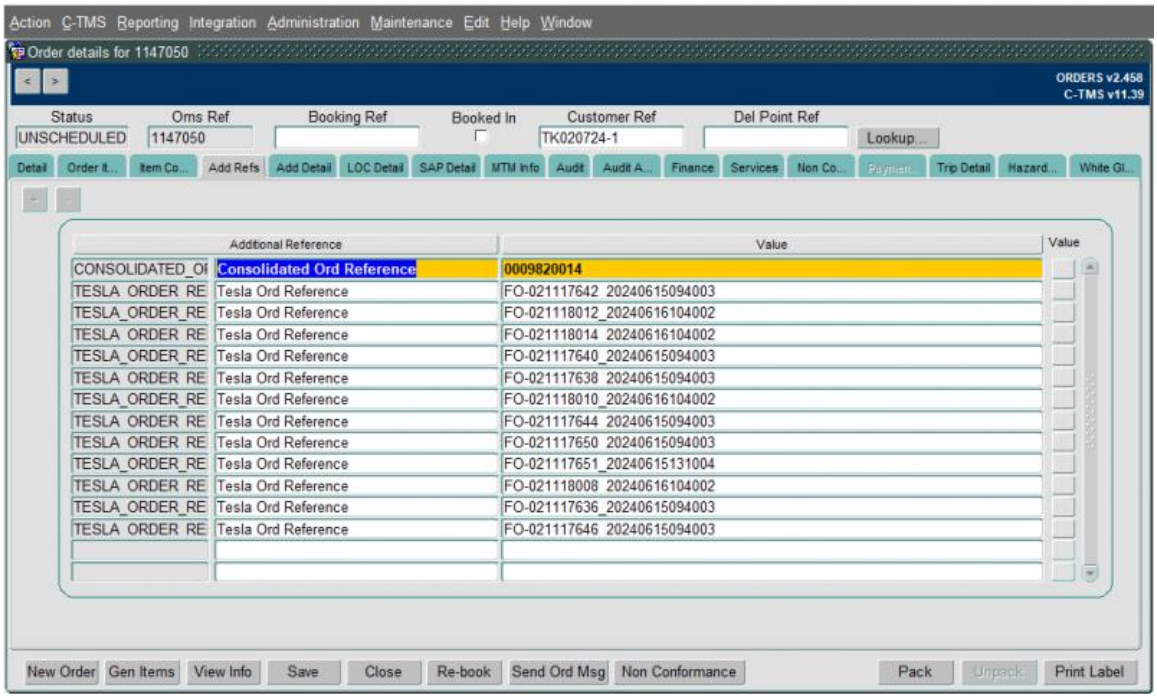
18.1 Order Search

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.

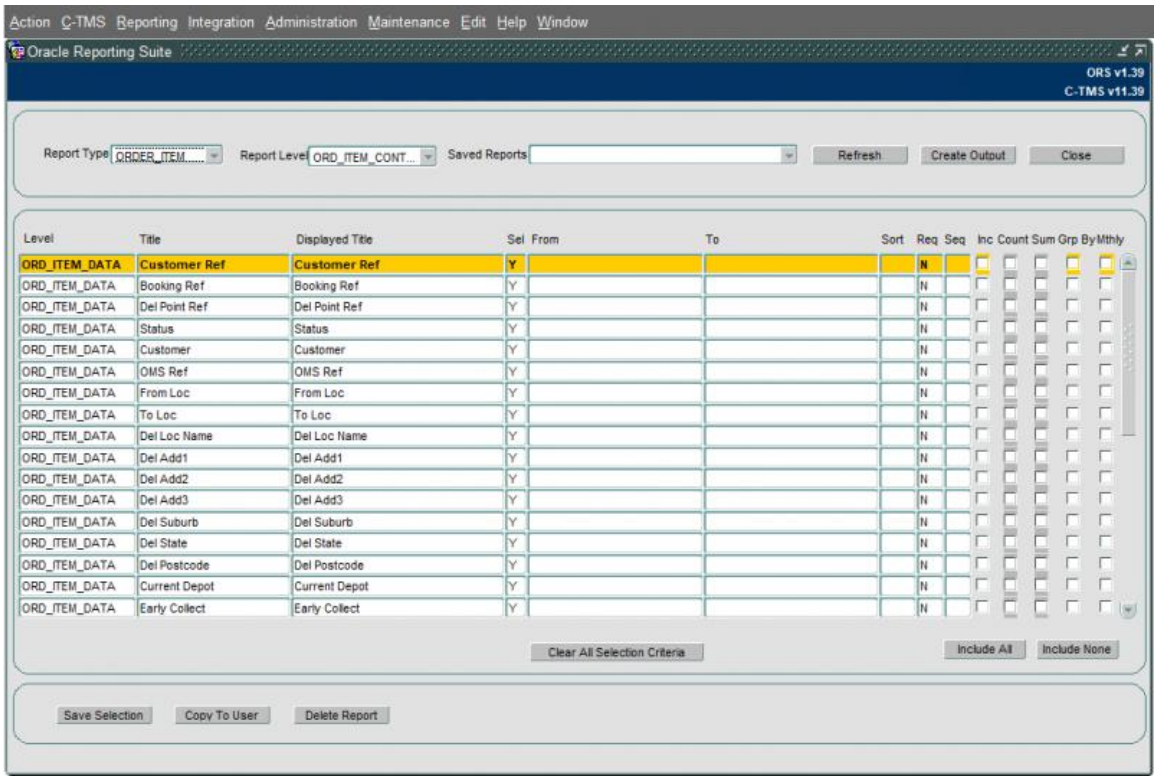
An order may have multiple references



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

18.2 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 -



18.3 Specification

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

19 Category:Support Documents

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

20 Tesla Orders EDI

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

20.1 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

20.2 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:

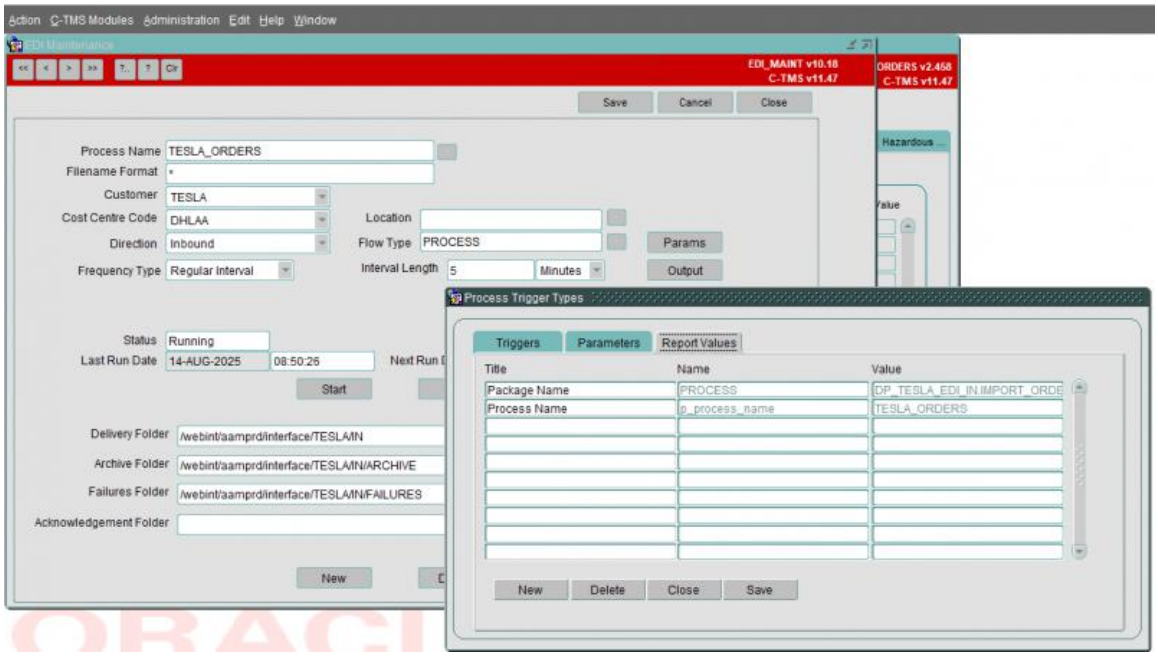
- TESLA_DELAERSHIPS
- TESLA_DU_TYPES

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

20.3 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

20.4 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.