DHL Invoicing

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

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.

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

Contents

1 Process - Creating DevOps for an Opportunity

The intention of this process is to standardise the creation of DevOps jobs associated to an opportunity, for PS and R&D to work in a standardized way.

1.1 Process

1. New case is logged on salesforce and a devops job created - This will have a general heading for the project and will become the Parent case. All requests for estimate, spec, development, release requests(if they come through here), bugs/fixes will be added to this 1 case on here.

2. All cases from Salesforce will be assigned to the swimlane projects.

a. If it is a user story then after analysis a child job will be created and assigned to a member of the R&D team for estimate, spec, development / fix, test, release. The Parent case will always be left on this swimlane.

b. The case assigned to a member of the R&D team for fix to be done.

3. The R&D team will work on the case assigned to them and reassign back to the tester.

Bug fixes will have a comment will be added using **@boomi_integration** so the customer can then be updated or specs/estimates issued.

If estimate / Spec has been produced a comment will be added using **@boomi_integration** so the specs/estimates can be issued to the customer. The case will then be added to the specs or estimates pending swimlane. Once approved a case will be added to the case for this to be scheduled for development.

4. Owner of the salesforce case will update customer and send estimates and quotes on the salesforce case only.

5. The child cases will be closed once software is released to test, the Parent case will not be closed until all the jobs and UAT have been done.

With the above this will give us full visibility of what is happening from both salesforce and devops.

2 Process - DevOps Testing Process

The intention of this process is to confirm the process that all should follow so that the reporting of rework reasons can be followed.

2.1 Finishing Development

R&D

- update Resolution Notes.

- assigns case to Development Manager.
- Set Sub state to *SCR Done*

Dev Manager

- Assign to release manager to be released

Release Manager

- release to 2ndary test environment.
- Assign to Dev Manager

Dev manager

- Assign to Tester.

- Set Sub-state to *On Test*.

2.2 Testing

Testing shall produce a POT/PO2T document in the shared directory \\DGA1FS01OBS\Test_Logs\{System}\{Customer}\{DevOps}.

The document shall be placed in a sub-folder named matching the description of the DevOps case being tested.

The document shall be named POT/PO2T followed by the same matching name of the case being tested.

The format is variable, but the appropriate template should be followed.

Template: File:POT-SFNumber Description v1.1.dotx

Tester:

- Set Sub-state to *Being Tested*

2.3 Passing Testing

- Set Sub state to *Ready for Rel*.
- Update Notes.
- Link in POT document
- Assign to Dev Manager

2.4 Failing Testing

What to do to the original case

- Set Sub state to *Test Failed*.
- Raise a new DevOps **Bug**.
- Update comments.
- Link in POT document
- Assign to Dev Manager

Note: When multiple bugs are found, raise a DevOps case for each bug. Bugs can be consolidated together t=if this makes sense and they can be done together.

2.5 Raising DevOps Bugs

Every error in testing shall be raised as a separate DevOps case:

- The Name shall describe the system, customer/DB and short description of the fault.
- The Description should be described and, if necessary, placed in a document and attached or linked to in the description.
    - Describe in detail or
    - Upload a DOCX file or
    - Add a link to the shared POT document in the shared filesystem.
- Any tags being used by the customer/R&D should be set
- Area - should be set to the system and product.
- Iteration - to the latest iteration for the R&D group.
- The Repro Steps should be noted in detail or linked to the document uploaded/linked.
- The customer must be set
- Priority - the priority of the fix required. Set as per the standards. See table below.
- Severity - the severity of the bug being raised. See table below.
- Bug Type - root cause analysis - see table below.
- Fault - root cause analysis - see table below. Where there are multiple bugs consolidated into one, choose the worst fault.
- Sub State - use the list of values, values defined below
- The parent of this case shall be the User Story being tested.
- Assigned to the development manager, Karthik or Carl.

2.6 Priority, Severity, Bug Type and Fault Cross-reference

Priority and Usage

Priority	Description
1	Must be completed now/next
2	Must be completed before the next release
3	Must be completed for final release
4	Nice to have, not essential to complete

Severity and Usage

Priority	Description
1 - Critical	Showstopper.
2 - High	May have a poor workaround. Must be completed
3 - Medium	Impacting the customer, but with workaround
4 - Low	Nice to have, UI issues.

Bug Type Usage

Bug Type	Description
Configuration	
Customization	
Deployment	
Documentation	
Enhancement	
Environment	Has the bug broken or been caused by an environment issue, such as interfacing, filesystem, etc?
Functional	If there is a function simply not working, choose this option
Invalid	
Legacy	If whilst testing functionality, you note that an existing function not related to the development has been broken, choose this.
Localization	
Missed Requirement	If you note that the specification is explicitly missing the requirement, choose this one.
Performance	Functionality works but is unacceptably poor.

Bug Type	Description
Regression	If whilst testing functionality, you note that an existing function related to the development has been broken, choose this.
Security	
Setup	
Third Party	
Usability	
User Interface	Functionality works but implementation of the UI is poor.

Fault Usage

Fault	Description
Advice Provided to User	
Datafix	
Deploymemt	If a DevOps bug has been raised by R&D SOLELY for release of product changes, then the bug type should be set to this.
Design Constraint	Bug is as designed, but requirement is not fully met
Design Error	Bug is as designed, but not fulfilling requirement (FS Fault)
Development	Development issue
External Interface	Caused by external system
Hardware/Comms error	
Implementation	Caused by Implementation failure
Missing Function	Function completely missing.
No Fault Found	
Program Error	New bug
Program Error (Rework)	Bug directly with new work.
Revised Requirement	FS change
Unable to Replicate	

Sub-states and their usage

Sub-state	Who	Description	Action
For Investigation	Tester/PS	Set initially when a bug is raised.	Assign to Dev Mgr
Investigating	R&D	When investigation begins	Assign to R&D
Being Programmed	R&D	Optional - if the development/fix is multiple days.	
SCR Done	R&D	Work complete. Assign back to development manager	Assign to Dev Mgr
On Test	Dev Mgr	When assigned to tester.	Assign to Tester
Being Tested	Tester	When being tested	
Test Fail	Tester	If the issue has failed.	Assign to Dev Mgr
Ready for Rel	Dev Mgr	When the code release is ready	Assign to Rel Mgr
Installed on TST	Rel Mgr	When the update is released to test	Update Hotfix in Release to patch/release version
Installed on PRD	Rel Mgr	When the update is released to production	Update Hotfix in Release to patch/release version
On-hold	Anyone	When the case is put on hold	**USE WITH CARE. ENSURE ALL MANAGERS ARE NOTIFIED.**

2.7 Developer Updates

R&D

- New Bug
  - Follow similar process for Finishing Development above re:
    - Sub State
      - Resolved Details
    - Update

- Bug Type
- Fault
- Comments

◊ Assign to Development Manager

♦ Existing DevOps Use Case (if assigned to R&D)

◊ Update comments

◊ Assign to Development Manager

Note: If multiple bugs are being fixed by the same release (i.e. in the same program/package), then ALL bugs must be assigned back to the Development Manager and on to the tester. Furthermore, one of these bugs should be considered the master - select one. Add a "Related" link to the other bugs showing that they are related to the first DevOps bug. This will help inform the Development Manager, Release Manager and Tester as to what can be tested and fixed together.

Development Manager

- The original use case and updated bug(s) should be returned to the original tester.

2.8 Releasing

Once all associated bugs are fixed, retested and assigned back to the Development Manager, they can all be released to the Release manager for releasing to the next area (TST, PROD, etc) using the standard product-related release procedures.

See also:

- Creating Release Notes

3 Process - Estimate Document

This page clarifies the Estimation process, specifically production of an EST document.

3.1 Pre-Requirements

- One or many of:
  - ♦ BRD
  - ♦ SDD
  - ♦ Combined EST

3.2 Process

Note: Regardless of who is doing the FS, the technical aspects should already have been discussed with R&D and noted.

3.3 Creating an EST

The Estimate may be created as a Word doc or within Assist Calidus Hub and exported as a PDF.

3.4 Using the EST Word Template

Note: If an EST spreadsheet of any form has not been done yet, do this now. Instructions for doing this are here: Process - Using Office Templates

- Double click on the EST Template - this will create a new document
- Replace placeholder text with your text.
- Populate the header
- Copy the requirements from the EST/SDD/BRD into the Client Requirement section
- Create the Solution into the Solution section.
  - ♦ Strongly scope - if there are things that are not included or limited in the solution, make that clear in this section.
- Paste in the estimate values. Ensure that you account for your time, plus any time to issue/review the EST.
  - ♦ If multiple departments/people are involved in the specification of this change, then you should note your time on the estimate in hours in total (plus delta if you have returned to this following review) as a review note.
  - ♦ One of the authors should be assigned the final review, to total up all this time and amend the FS time, and remove any notes relating to this.
- Save as EST-{CaseNo}-{Client} {Desc} v{Version}.docx

{{Note|

- Document properties should be used for the Title.
- These are used in the Heading.
- You can refresh all references in the title using [CTRL-A]-[F9] when in the title section.

3.5 Using Assist Calidus Hub to create an EST

Note: You must be logged in to be able to create new pages in Assist.

- Click the link here: Create New EST from Template
- Enter the EST name following convention EST-{CaseNo}-{Client} {Desc}. **Note:** No version in the page name.
- This will start an editor - this may be in Source mode or Visual Editing mode.
- Enter the #vardefine values, following the guidelines in the template.

- Enter text in all sections following the guidelines for the Word Template above.
- Enter the estimate values in the Appendix template provided.
- Save and provide reasonable change notes e.g. v0.01 - initial creation.
- Export the PDF using the link provided - this will append the version number.

Note:

- Follow the guidelines and help here: Assist Editing Guide.

3.6 General Notes

Note: This document should NEVER contain technical notes e.g. packages/procedures, form names, etc. Be general - describe what is being provided, call out which areas are being changed and what is being changed and why this will fulfil the requirements. The ONLY exception is when this estimate is being provided to technical representatives rather than direct to the customers' operational staff.

4 Process - Functional Specification

This page clarifies the Functional Specification Process

4.1 Pre-Requirements

- One or many of:
  - BRD
  - SDD
  - EST (spreadsheet and document)
  - SOW
  - Combined EST

4.2 Process

- Requirement and functional overview created by PS lead for the project/change. Also technical notes as to how this was estimated.
- Handover to R&D.
- Testplan created at this stage, wrt scenarios (see below)
- R&D to create technical notes to implement the change as requested.

Note: Regardless of who is doing the FS, the technical aspects should already have been discussed with R&D and noted.

4.3 Creating an FS

The Functional Specification may be created as a Word doc or within Assist Calidus Hub and exported as a PDF.

4.4 Using the FS Word Template

- Double click on the FS Template - this will create a new document
- Replace placeholder text with your text.
- Copy the requirements from the EST/BRD into the Client Requirement section
- Copy the Solution into the Solution section.
- Take any scoping comments from the Solution and move to Scoping.
  - Strongly scope - if there are things that are not included or limited in the solution, make that clear in this section.
- Ensure that Pre-requisites are populated if required.
- Data section should include any backing data that should be set up and indicate whether this will be done by the client or by Aptean. Indicate whether this will be automated through a script at release, or post-implementation task.
- Implementation Advice should show whether anything else should be setup e.g. EDI, ORS reports, recreating reports, plus details.
- Functional Description should expand on the solution, focussing on detailing the functional components that will change (or not change) as part of this change.
  - For example, the solution overview may say that a screen is changing to add a new field X. The Functional description should include a prototype screen layout or detailed description of where this field should be, what screen, etc.
  - **Note:** This section is critical - this section is what is signed off by the customer. Use customer wording or phrases if appropriate, to ensure that the customer understands the functionality in detail. Explain why this solution will fulfil their requirements.
- Technical notes may be structured and formatted in any way you please. However, this section is to be written/reviewed by R&D BEFORE release to the customer.
- The Test Plan should be created (from a template) and embedded in the document in this section, or pasted into this section, as long as this section is landscape.
  - The test plan should be focussed more on scenario testing than unit testing. It is expected that the R&D

operative will conduct appropriate unit and limit testing WITHOUT having to have explicit instructions, UNLESS that unit/limit testing is core to this change.
- Paste in the estimate values. Ensure that you account for your time, plus any time to issue/review the FS.
    - If multiple departments/people are involved in the specification of this change, then you should note your time on the estimate in hours in total (plus delta if you have returned to this following review) as a review note.
    - One of the authors should be assigned the final review, to total up all this time and amend the FS time, and remove any notes relating to this.
- Save as FS-{CaseNo}-{Client} {Desc} v{Version}.docx

Note:

- Document properties should be used for the following:
    - Title - without references
    - Version - 2dp e.g. 0.01
    - Yearcopyright - e.g. 2025
    - Client Company Name - {Client Name}
    - Area or Project/System - {System & Release Version}
    - Reference - {Case No - Cust Ref}
- These are used in the Heading, Footing and title pages.
- You can refresh all referenecs in each section using `[CTRL-A]-[F9]`

4.5 Using Assist Calidus Hub to create an FS

Note: You must be logged in to be able to create new pages in Assist.

- Click the link here: Create New FS from Template
- Enter the FS name following convention FS-{CaseNo}-{Client} {Desc}. **Note:** No version in the page name.
- This will start an editor - this may be in Source mode or Visual Editing mode.
- Enter the #vardefine values, following the guidelines in the template.
- Enter text in all sections following the guidelines for the Word Template above.
- Enter the estimate values in the Appendix template provided.
- Save and provide reasonable change notes e.g. v0.01 - initial creation.
- Export the PDF using the link provided - this will append the version number.

Note:

- Follow the guidelines and help here: Assist Editing Guide.

4.6 General Notes

Note: All technical restrictions should be referred to the project PS lead in a timely fashion. This should not be an email or automated devops - this is directly affecting progress on development - organise a meeting asap to resolve the technical restriction. Bring examples and possibilities to the meeting.

5 Process - Reporting Rework Faults

The intention is to standardize the way that DevOps cases are documented, so that the issues list can identify rework bugs and what caused the issue.

5.1 Using

Devops/Queries.

5.2 Configuration

Create a query with the following columns:

- Title
- Assigned To
- State
- Sub state
- Priority
- Severity
- Created Date
- Tags
- Work Item Type
- Bug Type
- Fault
- Hotfix in Release

Add criteria to select the data, typically:

- Work Item Type - Any
- State - Any
- EITHER/BOTH/NEITHER - depending on requirements.
    - Tags contains - select your tags
    - Custom Customer Contains - Enter a part of the customer

If possible, use a shared query or create a shared query.

An example query is here:

- Shared Queries/DevOps Extract for Analysis

Create a DevOps Bug List spreadsheet from template.File:Client BugList Analysis.xltx

5.3 Exporting

Run the Query.

Use the triple-dot menu on top right.

Choose Export to CSV

Open CSV

Copy all data (minus headers)

Open DevOps BugList spreadsheet

Paste data into first tab (DevOps Logs).

Go to second tab (Analysis)

Click Data/Refresh All.

Save.

The Analysis tab will break down the issues in the list and summarize the data.

The analysis provided:

- Total Bugs
  - ♦ Filters:
    - ◊ By State - you can select the State from the "Row Labels" drop-down filter in the header.
  - ♦ Reporting count and percentage per state.
- Bugs by Severity
  - ♦ Filters:
    - ◊ By State - you can select the State from the "State" drop-down filter in the header.
    - ◊ By Severity - you can select the severity from the "Row Labels" drop-down filter in the header.
  - ♦ Reporting count and percentage per severity.
- Bugs by Priority
  - ♦ Filters:
    - ◊ By State - you can select the State from the "State" drop-down filter in the header.
    - ◊ By Priority - you can select the severity from the "Row Labels" drop-down filter in the header.
  - ♦ Reporting count and percentage per priority.
- Bugs by System
  - ♦ Filters:
    - ◊ By State - you can select the State from the "State" drop-down filter in the header.
    - ◊ By System (Area Path)- you can select the severity from the "Row Labels" drop-down filter in the header.
  - ♦ Reporting count and percentage per system (area path).
- Bugs by Bug Type
  - ♦ Filters:
    - ◊ By Bug Type - you can select the bug type from the "Row Labels" drop-down filter in the header.
  - ♦ Reporting count and percentage per bug type, and fault beneath that.
- Bugs by Fault
  - ♦ Filters:
    - ◊ By Fault - you can select the bug type from the "Row Labels" drop-down filter in the header.
  - ♦ Reporting count and percentage per Fault.
- Bugs by Parent
  - ♦ Filters:
    - ◊ By Parent - you can select the parent case from the "Row Labels" drop-down filter in the header.
  - ♦ Reporting count and percentage per Parent case.

If you want more information on each of the bugs that were part of that group in the analysis, double-click on the count to the right of the group - a new tab will open up showing all of the bugs that match the count (usually called "Detail1").

6 Process - Updating Documentation

The intention of this process is to confirm the process that all should follow for documentation of new changes to software.

These are guidelines.

The goal is the important thing - everyone updates documentation.

Some products, or some smaller changes, may not require the full process being followed, but the principles are the same - ensure that documentation for changes to our systems is completed in a timely fashion, in a quality form.

6.1 When should documentation be updated?

For customer-facing documentation, whenever a release is made to the software to a customer system. So, whenever this get's a release note, ER or otherwise. There are exceptions, which should be managed by the development manager and project managers together, such as:

- Long-running project changes.
- Internal development.

For internal documentation, on processes being changed, new changes upcoming that require testing, etc, as and when this is passed to others. Also, this is a continuous process - if others are expected to use software to test or discuss with customers, then technical guides should exist for these functions.

6.2 Who should update documentation?

Everyone.

It is everyone's responsibility to update documentation.

- The developer completes a new development and creates a technical guide.
- The tester updates the technical guide.
- The release administrator updates release documentation.
- When released, the implementer uses the technical guide, the release notes and the specification to update the customer-facing documentation. They also correct any errors in the release documentation.
- The implementer may refine this through testing with the customer (s). This customer-facing and internal documentation is used as the basis of support handover.
- The support team update the documentation and FAQ guides to reflect real-world usage.

6.3 What should be documented or updated?

Everything.

In summary that is:

- Technical guides.
- Screen help.
- Processes.
- Release notes.

All of these combine together to provide us with quality documentation - for our customers and for ourselves.

6.3.1 Technical Guides

Technical guides should exist within CALIDUS Hub (this wiki).

They should be categorised for the product that they are relevant to, and classified as a technical guide.

This is not the same as the technical areas of a functional specification - things can change and adapt as development continues. Therefore this should contain enough pertinent information for the users (i.e. other Aptean staff, departments, etc) to be able to get this process running, for testing and release purposes.

This is the place for fields, tables, SQL if this helps,.

Cross reference any customer-facing documentation in other Assists, so that you do not have to re-type everything.

A good example is in this wiki: CTMS Paragon Interface.

6.3.2 Customer-facing Documentation

This exists with the appropriate product Assist implementation.

Any change, no matter how small, should be reflected in the appropriate pages within the Assist.

If a new page is created, it is important that that page is included in categories for the documents that are being produced for that particular area. This changes per Assist. As for information.

Change details should be provided with each change to each page made.

This should be consistent, but should reflect the patch, ER or release number.

See Saving your changes in the Assist Editing Guide for more details.

Change or reflect EVERY page that is required. For example:

- If this change affects maintenance screens, add the new field to the screen documentation
- If this change adds system parameters or settings, update those as well.
- If this change affects EDI processes, update those EDI processes as well.
- If this change requires implementation from Aptean staff in order to be enabled, reflect that in the documentation.
- If this change affects a general process, update that process documentation as well.

In general:

- If the screen documentation doesn't come up to the standards required, fix it as you edit.
- Change screenshots where required.

For example:

- The change adds a flag to Carriers on the screen, plus a new system parameter.
- The new flag can be imported.
- The result of these changes is to affect the scheduling engine.
- The change is release in ER47-189.

Your actions:

- Each change is added noting the release number in the change documentation.
- You update the carriers screen with the new field. You update the screenshot. You notice that some other areas are missing and update those.
    - ♦ Change comment "ER47-189 - Added X field and updated in general."
- You update the list of system parameters with your new parameter and description.
    - ♦ Change comment "ER47-189 - Added new system parameters for X"
- You update the import affected with the new field.
    - ♦ Change comment "ER47-189 - Updated Y import with new field X."
- You update the scheduling engine process documentation to show the affects of this change. You ensure that the system parameter that controls this is reflected in both the documentation and in any list of applicable system parameters within the guide. If one does not exist, you create it.
    - ♦ Change comment "ER47-189 - Added details of X functionality."

For example:

- You create a new Quarantine screen for CTMS, and this affects and is affected by changes in other systems (e.g. MCS).
- It is affected by various existing and new system parameters.

Your actions:

- Create a new page for the Quarantine screen.
- Create a redirect for the form name in CTMS.
- You add this Quarantine page to the appropriate overview guide (through categorisation).
    - ♦ You should look for examples of other pages and ensure that the appropriate categories are added.
    - ♦ In this case, as this is part of the Maintenance menu, you would add this to the Maintenance guide, the User Guide and to the Modules guide. That will ensure that the page is added to the guides when exported to PDF. Example:

```
<noinclude>
[[Category:Maintenance|150]]
[[Category:C-TMS Modules|D-150]]
[[Category:C-TMS User Guide|BD-150]]
</noinclude>
```

- You ensure that documentation for MCS is released and updated as well (it may not be you doing it, but it's your responsibility to make sure it's all done).
- As this is a brand new process, you create a new user guide for this as well, describing how this affected across all systems.
    - ♦ For example: "CALIDUS Quarantine User Guide".
    - ♦ If there are pages you want to pull in across systems, you can do this through interwiki links - see here: Interwiki.
    - ♦ This would be formatted as a formal Aptean document. Example: ctms:UG_CTMS_LogiNext_Interface_Guide
    - ♦ You may decide not to use the existing full page but just the screenshot. You do not need to create the screenshot again, just link to the existing screenshot.
    - ♦ You may decide that the MCS changes do not need to be documented here, but just referenced. You can do that using interwiki links to the documentation within the other Assists.

6.3.3 Customer-Specific Documentation

In general, our product documentation is exactly that - PRODUCT documentation, not customer-specific.

Any change that we undertake, whether product or customer change, is considered product, and should be documented as such.

This is accessible and visible to all of our customers, so we should steer clear of documents that mention specific customers.

Examples of good generic documentation

Bad	Good
Stapletons	Tyre transport operation
Stapletons Scheduling Engine	Fixed Drop Scheduling Engine

From time to time, a customer may ask us to create specific documentation for them. This is a chargeable task.

We should ONLY host that customer-specific documentation within the product Assist instance if

- the customer is aware that this will be visible to ALL Aptean Calidus customers.
- the customer agrees to this.

Regardless, customer-specific documentation does have a place in Assist. For example, these process-style documents are incredibly useful for project handover to support. So in that instance, they should exist within the (Aptean internal only) Calidus HUB Assist instead.

There is no issue with FAQs being put up within Product Assist instances, as long as they are removed of any customer names or true specificity, or the customer has agreed that this will be visible to other customers.

6.4 How much is this going to help?

By following this process, you have updated the affected pages (screens, processes, etc) only once.

These pages and screenshots are re-used and applied into ALL documentation.

This reduces fragmentation and increases reusability, and therefore reduces the time taken to produce all documentation across all systems for everyone.

The systems can directly link to the Assist pages you created and updated directly from the UI, meaning that the user has direct access to the latest documentation, not waiting for a published PDF to be provided to them or custom documentation being written with single-use scenarios. This reduces support calls and implementation time.

The old system of documents piling up for each customer isn't relevant any more - they can download the equivalent documents direct from the Assist. So no updating many documents because of a single change or hunting for the "latest document that was created when we last did this, but not that one because it's too specific".

As you are creating technical documentation as well, but also linking to customer facing documentation, you are providing an easier understanding path for other Aptean resources and departments to be able to understand, test and implement your change, not just one person who wrote a requirements document.

You are reducing inter-department calls and handover times - because quality technical and user documentation exists, you will not be pestered multiple times by testers, implementers and customers. We can all use the documentation to answer our own questions.

As everyone is responsible for updating and checking documentation, there is less chance that undocumented features are present, and any poorly-described functionality is refined by multiple hands, improving the quality of the documentation for the systems.

7 Process - Using Office Templates

This page ins intended to help with the process of using Office Document Templates, with specific instructions on BRD, EST and COST sheet templates.

7.1 Files

- Templates - document templates
  - Standard Aptean Templates - you should check for updates on Sharepoint - these may be old. Check
    - https://apteanonline.sharepoint.com/teams/ProfessionalServices/Shared%20Documents/Forms/AllItems
    - Aptean PS General Schedule to SOW Template 2023.dotx
    - Aptean Workshop Session Agendas TEMPLATE.docx
    - Aptean Engagement Report TEMPLATE.docx
  - BRD SFNumber Client v0.02.xltx - capturing business requirements from notes fragments, cross-referencing to create SCRs for a full project,
  - Calidus Network Diagram Template.vstx - Technical Architecture Diagram Visio Template
  - Client_BugList Analysis.xltx - referenced in "Process - Reporting Rework Faults" - paste in DevOps extracted bugs here, get reported stats.
  - COST-SFNumber Client Cost Sheet v1.07.xltx - cost sheet for full system implementations/new projects.
  - DEVLOG-SFNumber-ClientCode-Development & Implementation Issues List.xltx - a template for a spreadsheet to capture UAT issues and feed back from DevOps.
  - EST-CaseNo-Desc v0.01.dotx - estimate document template
  - EST-SFNumber Client Description v1.02.xltx - estimate spreadsheet in hours and days, calculating based on parameters.
  - FS-CaseNo-Desc v0.01.dotx - FS template. Also in Assist
  - POT-SFNumber Description v1.1.dotx - Proof of Testing Template
  - Release Note Template Aptean POD - Calidus Edition v4.x.xx.xx.dotx - Release Notes - use Assist template instead.
  - SCR-CaseNo-Desc v0.1.dotx - Small Change Request template - also in Assist.
  - Screen Templates.xls - a spreadsheet to help design oracle forms/new .NET screens
  - SDD-CaseNo-Desc v0.01.dotx - Solution Design/Requirements template. Also in Assist.
  - SOW-SFOppNo-Desc v0.02.dotx - Statement of Work - see notes above for validity.
  - SUP-Cust-Desc v0.01.dotx - handover to support document template.
  - TP-CaseNo-Desc v0.01.dotx - Test Plan template
  - Unlikely to ever be of use to anyone:
    - BPA-CaseNo-Desc Agenda v0.01.dotx - Elucid agenda template.

7.2 Notes on Templates

7.2.1 Using Office Templates

Note that templates do not open natively in Teams - you have to download them to use them.

They are double-licked on, then the appropriate office app will create a copy and open up that copy.

You can then save that as the new document you want to create WITHOUT affecting the template on which it is based.

In summary:

- Download the template
- Double-click on it

If you want to edit the template, first open your office app (e.g. Word or Excel), then open the template. Any change you make and save will be saved to the template, and then any NEW documents created from that template will have your changes. This will not affect any documents already created.

Be kind - upload your modification back to Teams as a new version - maintain the system for everyone.

7.2.2 Using the EST template Spreadsheet

For a one-off estimate and SOW. See later for use to hold technical notes for a bunch of related estimates in a BRD or Cost Sheet.

- Calc tab
  - Enter the Customer and Rates year e.g. NHSBT, 2025. The rates for the sheet will populate below, if they have been set up - see Rates tab.
  - Fill in the yellow fields. Everything else will calculate based on settings. See Rates tab..
  - Fields in yellow highlight are designed for you to enter data.
    - REQ is time you have taken so far on the scoping.
    - EST is the time taken estimating, talking to R&D, etc.
    - REL costs should be 0.5d or 4 hours per system affected.
    - IMP costs should be the amount of time to run additional scripts, set up the data for the customer, any agreed post-implementation consultancy time required.
  - You can then copy the Cost Detail section for pasting into an Estimate, or the lower table for pasting into an SOW.
- Notes tab
  - Keep a note of all your discussions, and also your prosed technical solution notes here.
  - Ensure that System, Lang and Est columns are populated where required. Area is for your own notes.
  - The table will total your estimates at the bottom.
  - The sheet will calculate percentages by language and system - you may need to refresh the data source or tables to make this happed. Easiest is to select the Data tab menu at the top, and click Refresh All.
- Rates tab
  - You should enter the customer, year, activity, day rate and currency here if not present. Contact your manager/PM for the relevant rates. You should amend the template with the new rates, so you don't have to do this every time. Remember rates change every year. Hourly rates will automatically calculate.
  - Settings show how the Calc tab calculates the rest of the settings.
    - FS/ST/CON/PM - percentage of DEV to apply to calculate these figures in the sheet.
    - Rounding - how closely to round. - 0.25 or 0.1 Leave as s.
    - Hours per day - uses this to calculate the dev activity costs, but also the rates. Leave as is.
    - Contingency - be led by your PM. Assume 10 or 20% uplift as required.

Notes:

- If complicated, I always include a contingency line in the notes, adding a little more to the estimate on top of any global contingency added to the estimate by the calculations. This is to account for something difficult taking longer than expected. Be led by R&D advice for this.

7.2.3 Using the BRD template

The purpose is to help the process of taking notes and translating them to changes when doing a larger analysis.

It's also a working document to refine and capture multiple meetings and notes, all the way through to fully-qualified and quantified SCRs.

- Notes tab.
  - You will have taken notes - give them a reference (e.g. TW/20251211-1) and enter this and a brief description in the Refs tab. Refer to any supporting docs in this note like the ER(s) you gene rated the notes from.
  - Enter your notes here in the Description column. Break them down to small pieces/statements, in general the smaller the better.
  - Then for each note, categorise by Area. Note if more complicated than this, add a column for sub-area.
  - Add a section title - Number those sections by logical area.
    - Use two decimal places for IDs. For WMS that might be
    - 1.00 - General Information
    - 2.00 - Config
    - 3.00 - Receipt
    - 4.00 - Putaway
    - 5.00 - Movements
    - 6.00 - Stock Take
    - 7.00 - Orders
    - 8.00 - Picking
    - etc.
  - Make sure every note is then IDd under the section e.g. 1.01, 1.02, etc. This will be useful later.
  - Mark every note you have entered has Source column marked with the Ref for that meeting note.
  - Mark every note as to which System this applies to (WMS, PORTAL, WCS, etc) in the System column.
  - If a note refers to or expands on a previous note line, use XREF to show the note to which this cross-references. Don't just type in the ID, type in "=" then selet the ID field of the note you are cross-referencing.

- Do this for all your meetings and notes, from as many people who provide them to you.
- Now use SCR column t decide whether you think this may be a change. Y, N, ? (unsure). The column will colour code, so anything red/yellow is something you have to deal with.
- Enter any clarifications you want with the customer in the Notes column.
- Run through and MoSCoW each change. By convention:
    - M(ust) - critical to do, phase 1.
    - S(hould) - will be included in the estimates phase 1
    - C(ould) - will not be included phase 1, could do it later. Will highlight yellow
    - W(ould) - will not be done. Will highlight yellow
    - ? - Unsure. Will highlight yellow

- SCRs tab.
    - Now go through each note you have marked as SCR = "Y" or "?". Basically, if you need an SCR, add it to the list in this SCRs tab.
    - Enter the SCR number.
        - By convention, anything chargeable to the customer starts at 01, anything that might be implementation or product dev starts at 99 and counts down.
        - By convention, if you want to split an SCR into smaller components (perhaps for multiple systems development, add an A, B, C etc to the end).
    - Cross-ref the IDs in the BR (optional)
    - Enter System, Area, Sub-area and a short description.
    - Categorise however you like, but expect that PROD would be product development, you might use this for phases, or leave blank and work with it later. Basically it's usually a good column to sort and format stuff later. Consider adding conditional formatting on NA to make it red and strikethrough, adding why this change is no longer applicable in the Notes column.
    - Enter a basic Hours or Days (whatever your favoured convention is here). Don't get too bogged down in details just yet - this is tshirt sizing at this time (2/5/10/25, etc). The total column will calculate for you based on some basic uplift settings in the Summary sheet.
    - Hide the following columns initially - they are only used when we start to make further refinements later with the customer
        - Initially provisioned
        - Expected
        - Delta
        - Delta Notes
    - Use the cross-referenced BR ID to get more information.
    - Paste any tech or clarification notes into the Notes column.
    - When the SCR is created, enter that in the SCR column in the Notes sheet. Again, use "=" and then select the ID column on the SCR tab to get it as an updating cross-reference - if you change SCR numbers later, the references will update on the prior sheet and vice versa. Enter that ID against every BR note that the SCR could reference. Extremely useful for discussions later.
- Summary sheet
    - Maintain the uplifts here, similar to the EST sheet.
    - You also have summary stats of changes/days per category, very useful for phasing, e.g.

Now you have a list of SCRs and t-shirt sizes, ballpark costings and criticality to start discussing with the customer.

As discussions go on, more notes can be generated - follow the same process to put them in the BR sheet. Amend any SCRs.

When SCRs have been initially discussed and categorised, the customer will likely want you to firm up the estimates.

Use the SCR numbers and fully cross referenced notes you have to put together discussion notes for use with R&D.

The best way to go about this is to use a variant of the EST sheet above.

You may also move forward from this point by creating a change document, SOW or SDD. In any case, you can use this sheet to keep track of where you are up to in documenting and ensurign all requirements are met.

- Notes tab:
    - Resolved % - has this been resolved 100% by the SCRs?
    - Section - what section is this populating the final doc (number, text or just a Y to say you've included it/copied the descriptive text into the final doc.)
- SCRs tab:
    - Doc - have you added this into the final doc or created a doc for it - y.

7.2.4 Using the EST template for multiple SCRs.

We will use the sheet as before, but now we will make a copy of Cost tab and call it SCR-01.

We will then use this to collate discussion notes from he BRD into this tab.

We will do this for each SCR.

We will either estimate ourselves and make technical notes, or we will discuss with R&D and make technical notes in here, deciding on the cost for each line, if any.

Then we can use the total to revise/refine the days back in the BRD.

We should also fill in the EST sheet in exactly the same way as before, including release and imp totals - although these are not used right now, these may be useful later if the customer chooses to break these SCRs down into other phases, or do SCRs singly.

7.2.5 Using the Cost Sheet Template.

This can be used right from the start as a full project cost sheet, or can be used after the stages above to start to pull together full project costs.

note that, due to rounding, a fully qualified BRD list of SCRs will probably not exactly match the costs in a cost sheet - agree with the PM which is the cost mechanism and use accordingly. It's up the PM to manage this with the customer.

Note also that this template includes some of the tabs from the BRD template (Notes, References) - if not going from BRD initially, you can use the same process here on these tabs to do a similar process of creating notes and cross-referencing SCRs, but this time the cross-referenced SCRs go in the Configuration and Development Tab.

- Profile tab
  - ♦ Enter the olive columns. Specifically:
  - ♦ Enter your rates.
  - ♦ "Fixed Cost" will apply an uplift like the Estimate sheet.
  - ♦ "Inc Optional" will include phase 1 and optional (phase X) changes in the total costs produced. Typically "No".
  - ♦ "Full Project" - applies system testing rates, typically the same as unit testing for all the changes, in the C&D tab. Typically "Yes".
  - ♦ "Hourly" - hourly rather than daily estimates - typically "Yes"
  - ♦ Enter the total sites and users for each Calidus product - this is typically used by the team producing SOWs to determine licence costs.
  - ♦ The rest of the sections should be populated if we don't have format RFQ information (which we should reference in the RFQ tab) or SDD/BRD notes (which should be referenced here or in other documents.)
- Assumptions & Comments - typically this is a restatement of general SOW comments. But you can add specifics here that the SOW team will use to help qualify the SOW.
- Configuration and Development tab
  - ♦ You will determine before or during this process a list of SCRs that you want to develop as part of this project, either now, pre-known requests from the customer, or from a BRD or BRD-style process. Enter them in the Configuration and Developments tab in a similar way to the BRD process above.
    - ◊ Phases can be shown in the Phase 1 and Optional changes sections. Split them accordingly.
    - ◊ Req column references the Notes, as in BRD.
    - ◊ Enter the System, Area, Sub-area and Description as before.
    - ◊ Enter the olive columns - specifically:
      - · Dev - the days/hours you estimate (either tshirt or from an estimate or combined estimate sheet, as before.)
      - · Config/Spec - any CON, REQ, Pre or Post imp cost SPECIFICALLY FOR THIS SCR - we will count some global values for these later, so just specific to this change.
    - ◊ Enter solution notes and comments as per the BRD process.
- Cost Details tab
  - ♦ You can enter the Olive columns. Some calculate already, but over-type if you're confident.
  - ♦ Use the notes column to qualify and quantify the value here. For example, you may add 30 hours for Release under the implementations section. You may then want to quantify and qualify that e.g. "Assuming 2 major releases over 2 systems"
  - ♦ Technical assistance (Other Sites) - If you're trying to cost a major implementation project across multiple sites, you may want to enter a value here. Simply enter the number of hours or days you want in the Notes column - the sheet will then calculate the total amount for other sites (total sites from the Profile

sheet, minus 1)

- Calculations tab
  - ♦ Some cross-references and settings, much like EST and BRD. Importantly, also:
    - ◊ Rounding - use 0.1 for daily and 0.25 for hourly.
    - ◊ Imp Hours per SCR - this is a number of hours to globally add for implementation per SCR, for the project costs. This covers standard consultancy and implementation changes, not SCR specific ones. Also used to calculate Super User Training costs, testing assistance, go live assistance.
    - ◊ Support Transition Hours per SCR - handover to support, plus 7.5 hours.

The result of this is the Summary of Costs tab, which is essentially a copy/paste into an SOW.

Note that the year 1/2 costs are included here. This can be modified to include the WCS/EPOD year 1/2 costs, or can be left blank/deleted - typically the Sales team will do this licensing and additional costs separately, and may need to include SaaS costs as well.

8 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).

8.1 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.

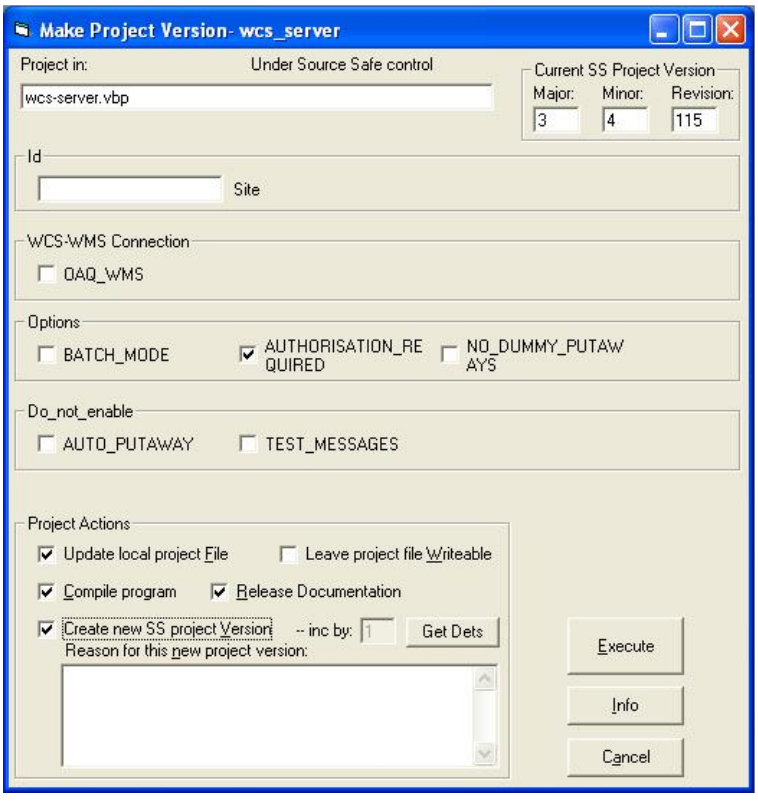
8.1.1 Responsibility

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

8.2 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:

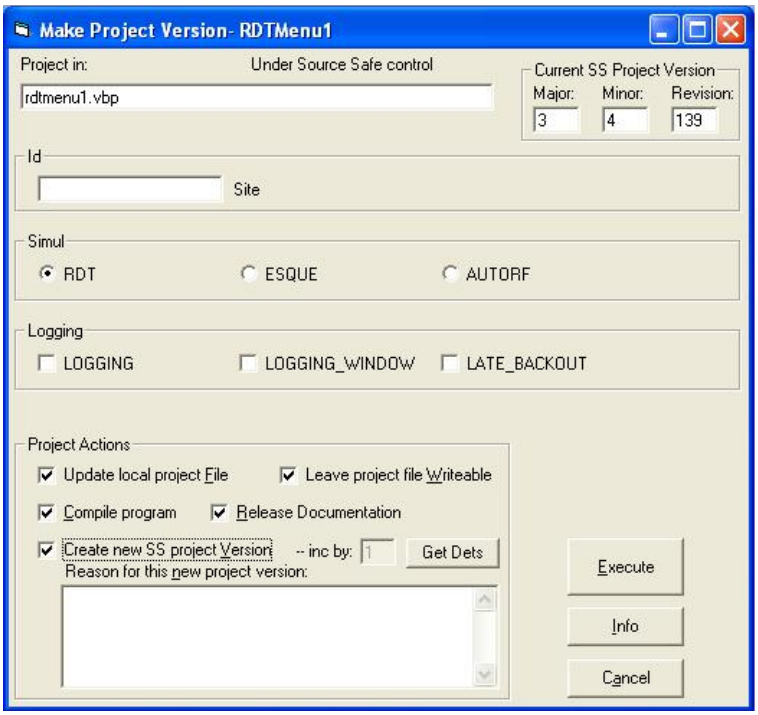


The screen shot shows the settings needed for a build of WCS-Server.exe. To build WCS-Server_OAQ repeat the call with OAQ_WMS ticked (after saving version number change to the .vbp file). Normally the same version number is

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.

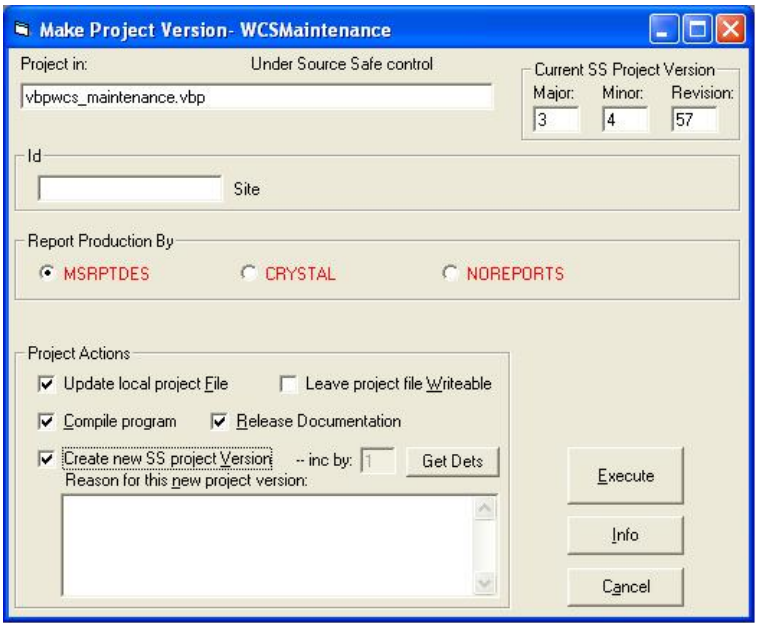
RDT:



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.

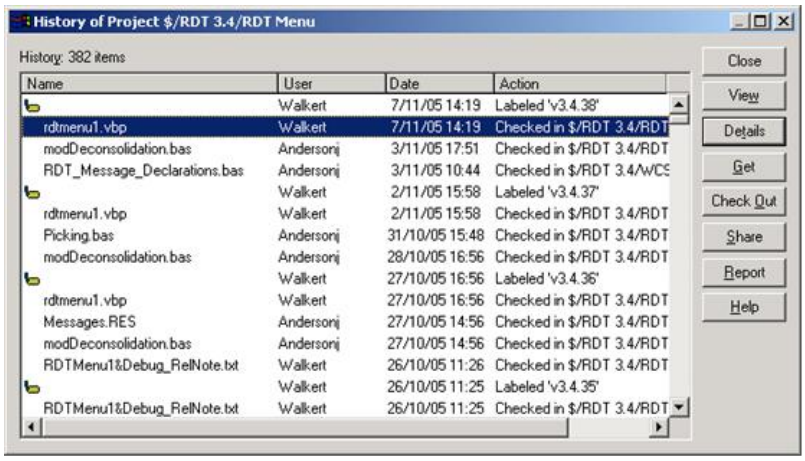
The build is normally preceded by a ?Get Dets? (see button above). This extracts from VSS details of the changes included in the build. This information is automatically written to the release documentation.

WCS Maintenance:

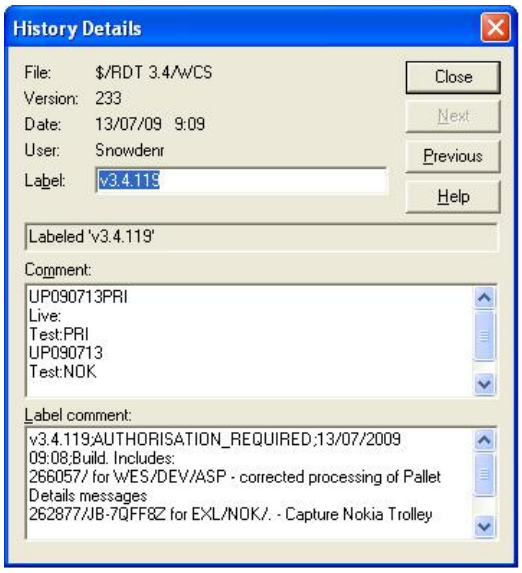


The build is normally preceded by a ?Get Dets? (see button above). This extracts from VSS details of the changes included in the build. This information is automatically written to the release documentation

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.

8.3 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.

8.4 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

8.5 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

9 WCS Release Procedure

9.1 Introduction

The purpose of this document is to document the release process for WCS

9.2 Release admin

9.2.1 Release Spreadsheet

The currently released versions of the WCS system can be found in:

WCS Release List.xlsm

The spreadsheet contains a list of all the packages available for release, the versions of the RDT, WCS and Maintenance EXE's and the rdt1 (RDB) and log/log1 (LDB) Database versions within each zip:

Release	RDT	WCS	MNT	RDB	LDB	DEP
UP190712	3.4.414	3.4.402	3.4.110	305	3	
						414
UP190718	3.4.415	3.4.403	3.4.110	306	3	
UP190724	3.4.416	3.4.404	3.4.110	306	3	
UP190806	3.4.417	3.4.405	3.4.110	307	3	Oracle
UP190815	3.4.418	3.4.405	3.4.110	307	3	

It also contains a list of each log - with a description and functional area - contained within the zip/release:

Release	RDT	WCS	MNT	RDB	LDB	DEP	Client	Ref	Desc
							Hallets Retail	358288	Receipt Printing Issue
UP190712	3.4.414	3.4.402	3.4.110	305	3		Moran Logistics	358352	WCS disconnection error
							Century Logistics	358359	Re-Logon Issue
						414	Cert	358110	Receipt Issues
UP190718	3.4.415	3.4.403	3.4.110	306	3		Century Logistics	357245	Pick Order From Pick Wave
							DHL Arthouse	358577	Blind Receipt changes
							Cert	357560	WCS Group Setting Issue
UP190724	3.4.416	3.4.404	3.4.110	306	3		Cert	358673	Inbound receipt Max Intake Life
UP190806	3.4.417	3.4.405	3.4.110	307	3	Oracle	Century Logistics	359143	Add UOM display on Picking
UP190815	3.4.418	3.4.405	3.4.110	307	3			358314	Carriage Return in barcode
								359341	Change DAMAGES Func to HOLD Func

For each version, an "L" (Live) and "T" (Test) will be in the appropriate row/column for the client:

Release	RDT	WCS	MNT	RDB	LDB	DEP	Client	Ref	Desc
							Hallets Retail	358288	Receipt Printing Issue
UP190712	3.4.414	3.4.402	3.4.110	305	3		Moran Logistics	358352	WCS disconnection error
							Century Logistics	358359	Re-Logon Issue
						414	Cert	358110	Receipt Issues
UP190718	3.4.415	3.4.403	3.4.110	306	3		Century Logistics	357245	Pick Order From Pick Wave
							DHL Arthouse	358577	Blind Receipt changes
							Cert	357560	WCS Group Setting Issue
UP190724	3.4.416	3.4.404	3.4.110	306	3		Cert	358673	Inbound receipt Max Intake Life
UP190806	3.4.417	3.4.405	3.4.110	307	3	Oracle	Century Logistics	359143	Add UOM display on Picking
UP190815	3.4.418	3.4.405	3.4.110	307	3			358314	Carriage Return in barcode
								359341	Change DAMAGES Func to HOLD Func

Each time a release is made, the L/T will be moved to the appropriate row.

9.2.2 Release Directories

The following directories are relevant to a WCS release:

projects\Releases\WCS\WMS\Release%20ZIPs|\\spekefs2012\projects\Releases\WCS\WMS\Release ZIPs

Contains the ZIP files which are copied to the sites and released.

projects\Releases\WCS\WMS\Release%20Notes|\\spekefs2012\projects\Releases\WCS\WMS\Release Notes

Contains release notes with details of the changes made inside each release.

projects\Releases\WCS\WMS|\\spekefs2012\projects\Releases\WCS\WMS

Contains sent emails of Release Notes generated from the XLS

projects\Releases\WCS\WMS\Release%20DBs|\\spekefs2012\projects\Releases\WCS\WMS\Release DBs

Contains backup versions of the rdt1 and log databases

Note: These Dir's also exist for TMS, replacing WMS with TMS.

9.3 Release Procedure

9.3.1 Identify the required zip/package

Check the Release XLS to find the requested Supimix references and cross reference this with the current T/L row for the client to see if the client requires a release. Check all Supimix references and make a note of the highest i.e. the latest package that contains all the required references.

NOTE: If the UPxxx package is in red text in the spreadsheet, then it means a significant bug has been found in that release and it should not be released - in this instance, use the next non-red package where this would have been fixed.

Make a note of both the current ZIP version and the version to be released.

9.3.2 Check if Log database release required.

projects\\product\\WCS\\All Projects\\Support|\\spekefs2012\projects\RDT\All Projects\Support]]

Note that this document is password protected.

For most clients with OBS hosted systems, copying the ZIP to site will simply be a copy/paste onto an RDP session. The RDP's can be found in:

[[file:///\\spekefs2012\projects\Releases\WCS\RDP's|\\spekefs2012\projects\Releases\WCS\RDP's

For some clients the process is more complex. The process for these sites is also contained within the Home Support document.

The file should be copied into the \Database\Upgrades directory found under the base install directory in the support document, e.g.:

Install Dir
E:\Program Files\CENWPRD Warehouse Control Server

Would mean to copy the ZIP file into:

E:\Program Files\CENWPRD Warehouse Control Server\Database\Upgrades

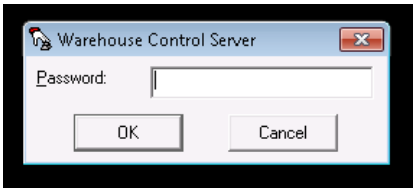
NOTE: You can shortcut the above procedure if the release is a Test To Live. In this instance you can copy the '.done' file from the Test system into Live and rename it back to '.zip'

\\spekefs2012\projects\Releases\WCS\WMS\Release DBs into the <base>\Database directory

9.3.3 Stop the system

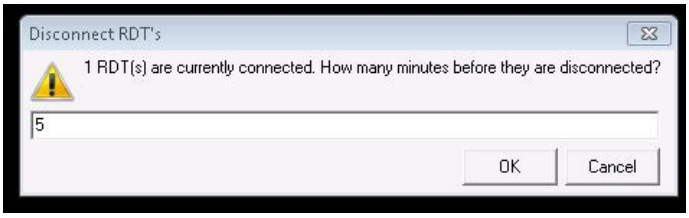
The WCS will be running in an application window, e.g.:

Close the WCS (check you're closing the right one!) by clicking on the X in the corner. You should get prompted for a password:



Enter "password" and click OK, then OK again at the 'Are You Sure' prompt.

If the WCS says 'Connected(x)' as in the above example, it means RDT devices are still connected and you will get an additional prompt:



This is a timer before the system will be taken down. If you want to delay this, enter a figure in minutes; otherwise enter 0 to stop the WCS immediately.

9.3.4 Check for running processes

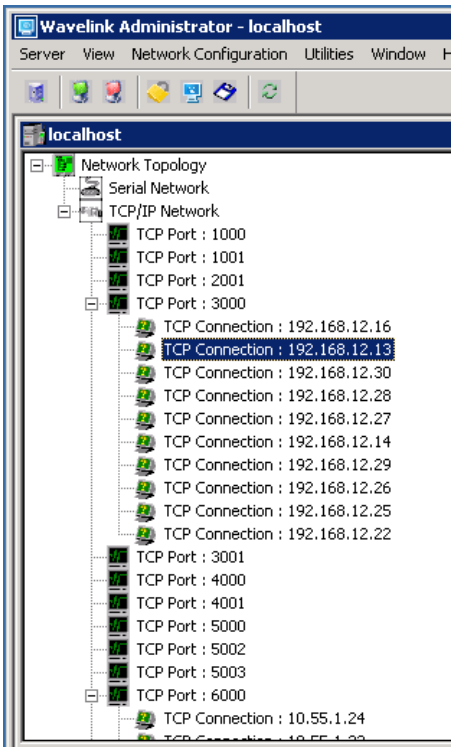
The quick way to check if processes are still running after shutdown is to check for the existence of an ".ldb" file against either the log (or log1) or rdt1 databases. These can be found in the Database directory underneath the base directory:

If no "ldb" files exist, then it is probably OK to proceed with the release, though it is advised to also do the checks below. If

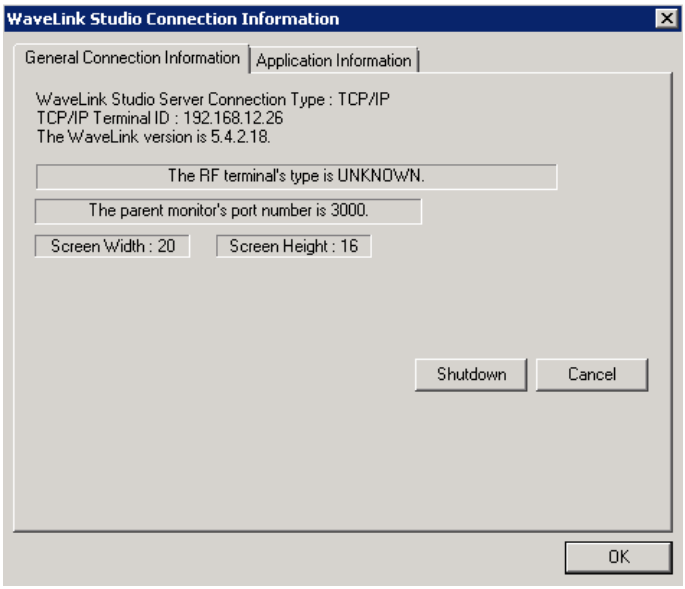
an "ldb" file exists, then the below checks must be done to release the lock - do not just delete the ldb file - it won't work.

Whilst an "ldb" file exists, you will NOT be able to do a successful release.

To ensure none of the files are locked, check WaveLink Studio Administrator (run from the Windows button or desktop shortcut) and check there are no active connections on the relevant Wavelink port - which can be found in the Support Document, e.g. :



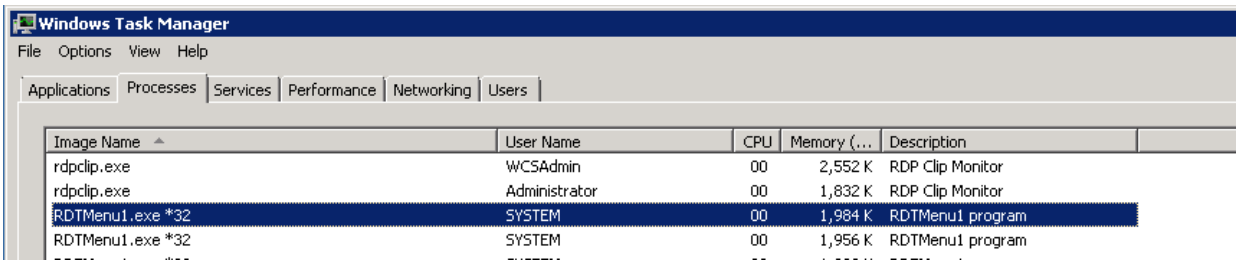
Double click on the TCP Connection lines (NOT the TCP Port) and click on Shutdown:



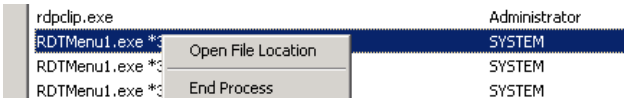
You may find these immediately start again, which could be a problem.

The Windows Task manager also needs to be checked for running instances of the Maintenance or RDT processes. On an RDP session the Windows Task manager can be accessed with the hotkey CTRL-ATL-END and selecting Start Task Manager

In the "Processes" tab, click on the 'Show Processes for All Users' then check the list for RTD processes:

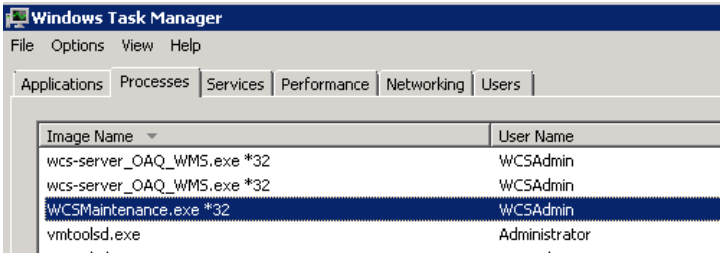


If there are ANY other WCS's running e.g. Live when you have taken down Test for a release, or other clients systems, then you **must** check which system the RDT is attached to BEFORE killing it. This can be done by right-clicking on each RDT process:



Then checking the File Explorer window that opens - it will open in the base directory of the running RDT, so you will then know if it belongs to the area where you are currently doing the release. If it does, then it can be killed by right-clicking again on the process and selecting End Process. **ONLY KILL THE PROCESSES FOR THE SYSTEM YOU ARE UPGRADING**

A similar process is also required to look for WCS Maintenance sessions which may also be locking the database:



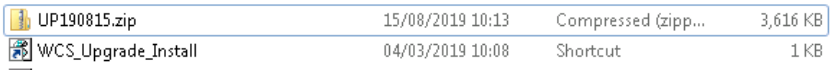
9.3.5 Release process

BEFORE running the release process, backup the rdt1.mdb database file in the format:

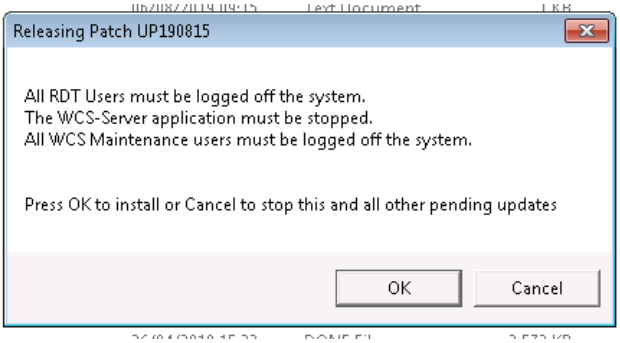
rdt1_<SITE><AREA>_YYYYMMDD.mdb, e.g.:

rdt1_MORL_20190101.mdb

The release process is run by double clicking the WCS_Upgrade_Install SHORTCUT:

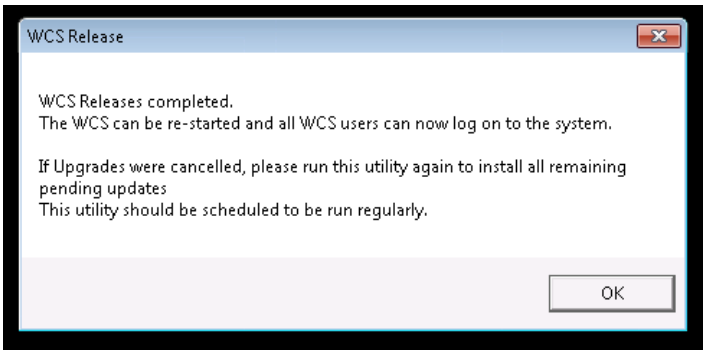


Once double-clicked, the process will be as follows - you will get a prompt:



Click OK.

When the release is completed, you will get a second prompt:



If a log database release is required (infrequent) then a further process is required.

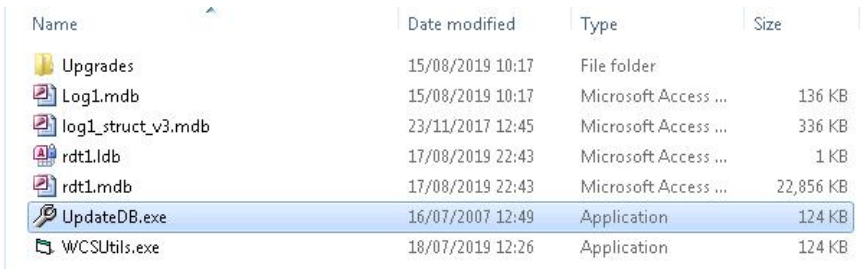
Check which log file is in use - it may be called either log or log1 - by looking at the most recent timestamp on the log databases

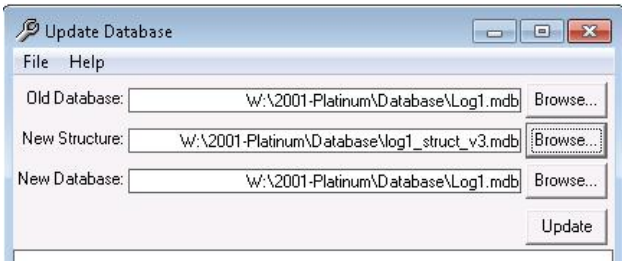
BEFORE running the database conversion process, backup the log1.mdb (or log.mdb) database file in the format:

log1_<SITE><AREA>_YYYYMMDD.mdb, e.g.:

log1_MORL_20190101.mdb

From the <base>\Database directory double-click the UpdateDB.exe:

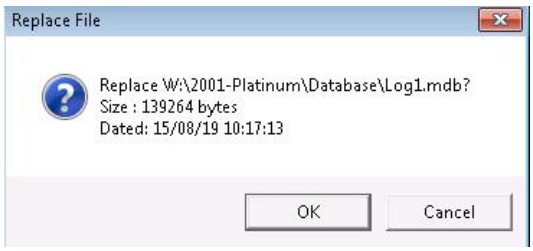




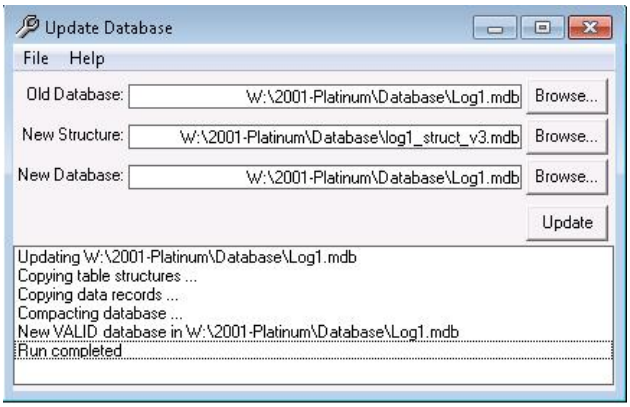
Enter the location of the existing log database (in both the Old Database and New Database fields.

Enter the location of the log struct file (copied as part of the 3.3 process) in the New Structure field.

Click on the Update button, and you will get:



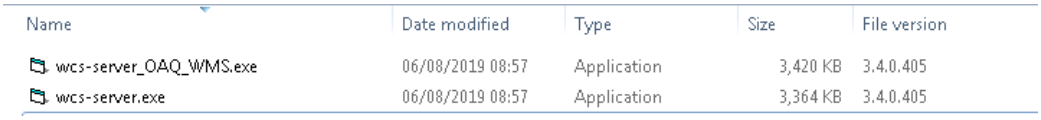
Click on OK, and the database will be converted:



When 'Run completed' then just close the Update Database box with the 'X' button.

9.3.6 Restart the System

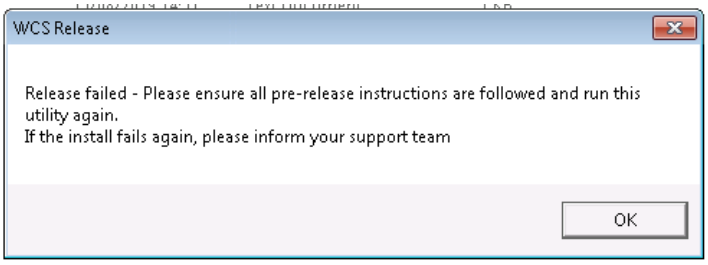
At this point the release is done and the WCS can be restarted by going into the base directory and double-clicking one of the following dependent on the system type:



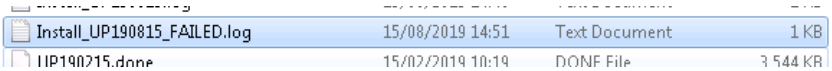
The OAQ exe is for Oracle systems, the non-OAQ is for powerhouse.

9.3.7 Release process failures

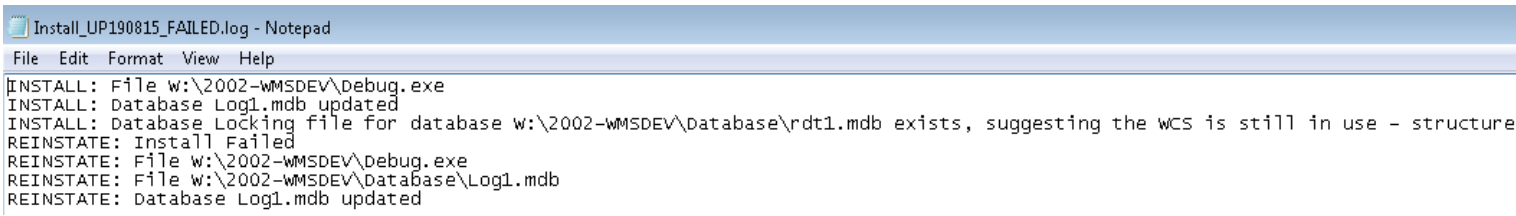
If the release process fails, then you will get the following prompt:



At this point check the relevant 'failed' log:



For details of the failure, e.g:

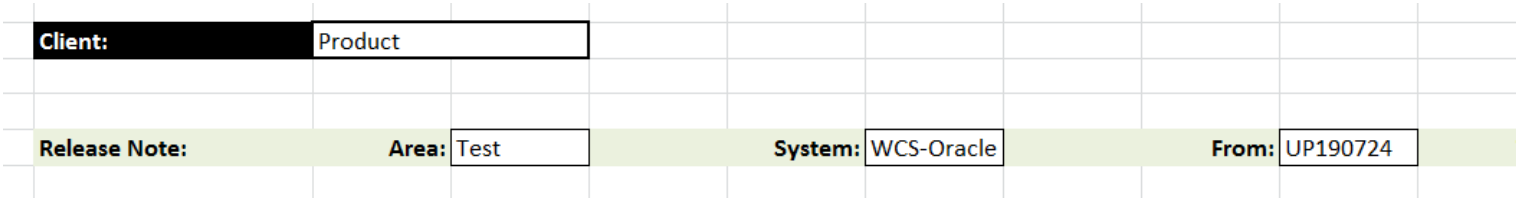


Indicating that a process is locking the rdt1 database.

If possible, fix the issue and re-run the release script.

9.3.8 Create Release Note

Once the release is complete, update the Release Spreadsheet for the client/system released, and produce a release note on the 'Reports' tab:

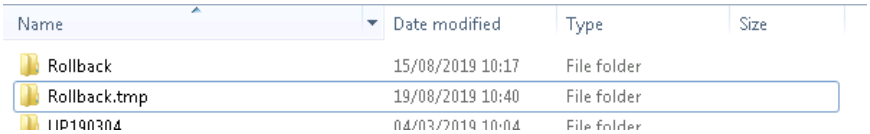


Enter the correct client, Area, System and the From/To zip versions then click on the Release button to create an email:

Send the email to the required recipients and drag/drop the sent email from your sent items into the release emails folder.

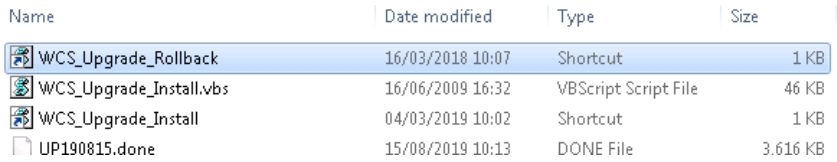
9.3.9 Release Rollback

Prior to a roll-back, check that a Rollback.tmp directory exists:

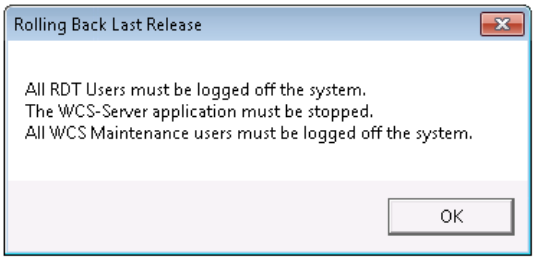


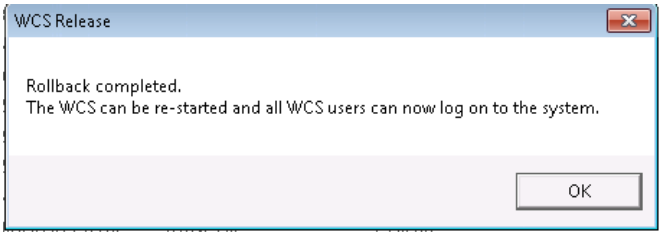
If not, create one.

To roll-back a release, double click on the WCS_Upgrade_Rollback script:



The process is similar to a release:





After the above process, if the log database was converted, then this will have to be reversed using the UpdateDatabase.exe selecting the previous log structure.