Process - Updating Documentation

Aptean Ltd

Contents

1 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.

1.1 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.

1.2 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.

1.3 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.

1.3.1 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.

1.3.2 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.

1.4 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 reducing fragmentation and reusability, and 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.