CTX-Manual-Intervention User Guide

Contents

Preface

1.1 About this Manual

This document provides an overview of how to use the Manual Intervention module, along with details of the components.

1.2 Related Material

- CTX-Manual-Intervention Deployment Plan
- Example flows contained within the CTX-Manual-Intervention module studiopkg file

Commented [AK1]: An acronyms section would be helpful at the start. Not sure if it goes after this or after Section 2

Commented [JL2R1]: Add: DB, UI, JSON

Formatted: Innovise Heading 2

1.3 Abbreviations used in this Document

DB	Database
UI	User Interaction (usually referring to a flow containing a LivePortal block)
JSON	JavaScript Object Notation

2 Versions

2.1 Document Revisions

The following revisions have been made to this document

Date	Revision	Notes
27/07/2021	1.0	First Full Release

2.2 Module Versions

The following updates have been made to this module, starting with the most recent

Module Version	Release Date	Comments
1.0	27/07/2021	**Module Flows and Subtasks** • GMIC-Get-Manual-Intervention-Cfg • MI-UAD-User-Access-Dashboard • MI-LPUI-Launch-Process-UI • MI-WFU-Wait-for-User • MI-LUD-Log-User-Data • SPAG-Set-Process-AD-Groups **Example Flows and Subtasks** • MI-Example-Onboarding • Check-Add-Asset • Generate-Username • Random-List-Selection • MI-UI-Onboarding-Department • MI-UI-Onboarding-Assets • MI-UI-Onboarding-Approval **Additional Subtasks** • CSL-GAG-Gather-AD-Groups • LP-BD-Build-Dashboard

Formatted Table

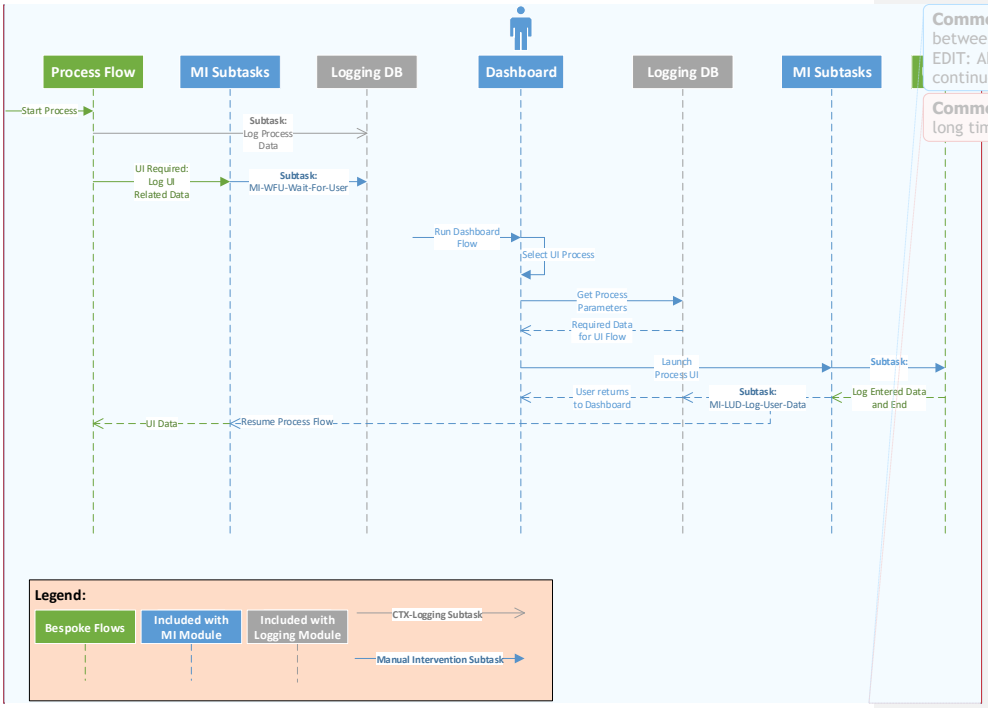
3 Requirements

The CTX-Manual-Intervention module requires the following:

- CTX-Logging Module
  - CTX-Logging Database Deployed on DB Server
- Manual Intervention Deployment
  - Configuration File created as per CTX-Manual-Intervention Deployment Plan
  - LivePortal Components saved in LivePortal directory /Custom/CTX-Manual-Intervention as per CTX-Manual-Intervention Deployment Plan

4 Manual Intervention Overview

4.1 Module Overview - UI



This diagram shows the interactions between the main Process Flow (with no associated UI), the relevant subtasks, the database and the UI Dashboard and UI Flows.

The main flow pauses when the MI-WFU-Wait-For-User subtask is called. This is now waiting for a UI, which can be triggered from the Dashboard flow.
This in turn opens a new tab to the associated UI. Once the user action has been completed, the LUD-Log-User-Data sends the request to continue the process flow ~~to continue~~.

Commented [AK3]: Not sure what the difference is between solid and dashed arrows
EDIT: Ah, it's when information is being passed back to continue running a flow?

Commented [JL4R3]: I think so? I made this diagram a very long time ago!

Commented [AK5]: Is it standard to have both the acronym and name within the flow?

Commented [JL6R5]: I think it used to be

4.2 Module Details

This module is designed to be used for case management using Cortex flows which may require a UI action (either a manual action or some additional information / decision making).

This allows flows to run unattended, but still gives control to the user - meaning an operator can login to the Dashboard flow and view all the processes requiring their attention and deal with them from a centralised UI flow.

Some examples of what can be achieved using this module are provided below:

- View all items pending User Interaction and see the logs from each execution.
- Easily open the UI in a new tab and interact with it.
- Allow a break in a process where a user can manually update a system, for example if there is no API available.
- Allow a break in a process where a user can find some information and update the UI to return this to the automation, for example if there is no API available.
- Select an action for the automation to perform where a human decision is required.

This means that a single process using this module would consist of the following components:

- **Process Flow**

  This is the main automated flow created to manage the case. These flows are developed as part of a solution which is using the Manual Intervention module and can be triggered via schedule, external call or another trigger mechanism. If the process or branch the process takes requires a UI, the main Process Flow will pause, using a subtask defined in the manual intervention module, until the User Action has been taken.

  This architecture means that the token does not have to wait on a LivePortal UI block (which may then timeout) until a user opens the page.

  An example (**MI-Example-Onboarding**) is included with the module.

  Please note that the Execution Reference (e.g. Order ID / Case ID)~~,~~ within the Process flow~~,~~ should be set as appropriate at the start of the flow using the 'Set Reference' block. This reference should be added as the External Reference when logging the process through the Module Intervention module and will be used to manage executions.

  This is used to refer to and identify this particular process execution and link the Cortex flow with the Process entry in the database.

- **Dashboard Flow**

  This flow allows users to view the logs associated with all the running processes and trigger any UIs, which open in a new tab.

  The 1st UI page is a dashboard which allows users to view different queues such as items requiring a User Action, In Progress processes, or queues based on a particular process.

Commented [AK7]: A summary or examples of what a user can do/achieve with this for their flows be useful? I.E is it things like simply start/stop or more complex things? At a glance/skim-read, aside from monitoring flows, I'm not sure what exactly this is for.

Commented [JL8R7]: Do the bullet points below help?

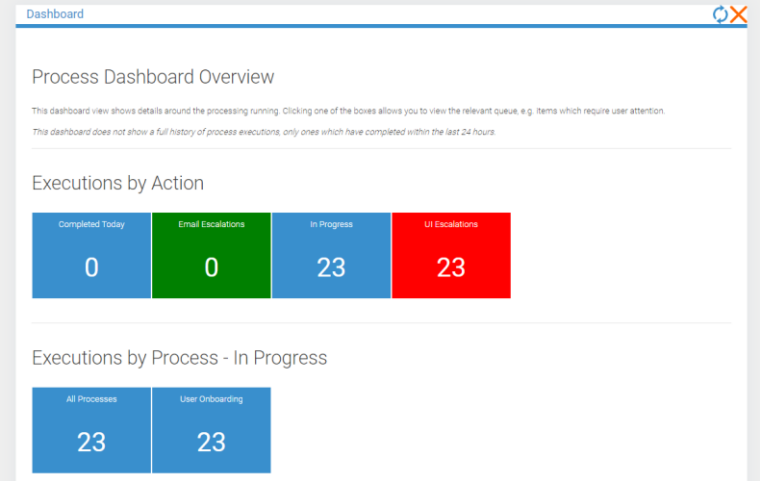
Formatted: No bullets or numbering

Formatted: Bulleted + Level: 1 + Aligned at: 0.63 cm + Indent at: 1.27 cm

Commented [RB9]: We should mention that this is created as part of the solution that the user is doing and not part of the MI module

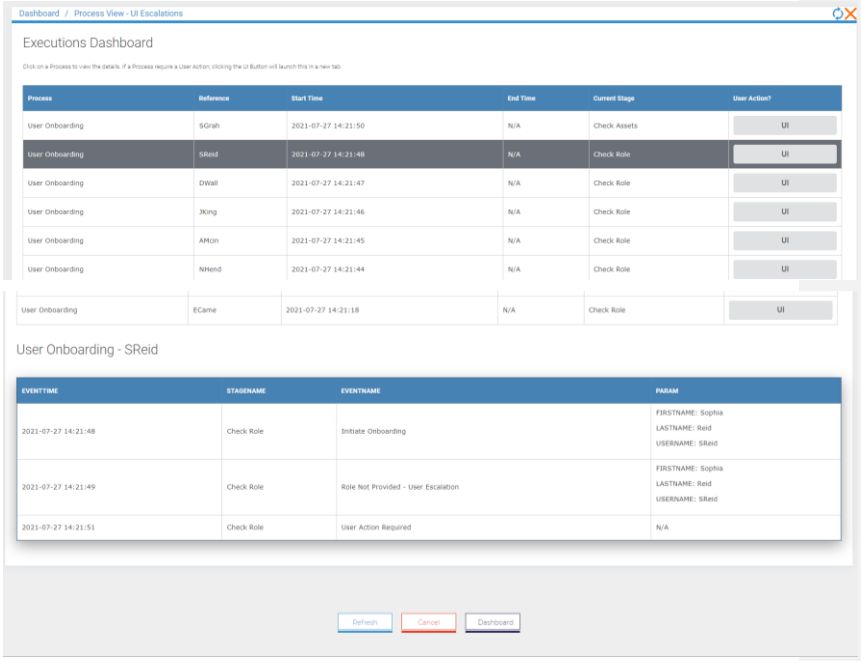
Commented [AK10]: What is an execution reference in this context?

Commented [JL11R10]: Hopefully makes it clearer



The 2nd page then shows all items in that selection and allows the user to trigger a UI (if the process is currently waiting) by clicking on the UI button. The user can click on a row to open a 2nd table below, showing the Stage / Event / Parameter logs.

Commented [RB12]: Minor comment. I would suggest to add a screenshot of the UIs



The flow (**MI-UAD-User-Access-Dashboard**) is included with the module as an example and can be customised.

- **UI Flow(s)**

  These are individual flows UI created to handle any User Interactions. These flows are part of the solution which is using the Manual Intervention module and will be triggered from the Dashboard Flow which will pass in any required inputs to use in the UI.

  Once a user has taken an action, the UI flow will log the event along with any parameters (such as data entered). The Process Flow will then continue and retrieve the parameters to use.

  The following example UI flows are included with the module:

  - MI-UI-Onboarding-Department
    - As part of the onboarding process, if the Department is not passed in to the main flow, then the user must select the relevant department from a dropdown list.
  - MI-UI-Onboarding-Assets
    - If a list of required assets is not provided, the user must select any required assets using the checkboxes
  - MI-UI-Onboarding-Approval
    - Once both previous steps are validated (with the data passed into the flow or with a User Escalation), the final step requires approval using checkboxes to select Approve or Deny.

4.3 Using the Module

This section details how the module should be used. An example flow (**MI-Example-Onboarding**) is included with the module for reference, along with 3 example UI flows.

The diagram in **Module Overview - UI** shows how the module will function, including the subtasks to be called.

For a simple guide to using the module, see Appendix A - Manual Intervention Cheat Sheet. That guide provides the details of each subtask required to use the module. Please bear in mind that it is aimed at users who already have an understanding on how the module works.

4.3.1 Manual Intervention Configuration Items

4.3.1.1 GMIC-Get-Manual-Intervention-Cfg

For any solution flow using Manual Intervention, the configuration should be loaded in at the start and stored in a Global Variable Structure. The configuration retrieved by this subtask comes from the Config file (C:\Cortex\Config\Manual-Intervention.cfg).

Any separate UI flows should also retrieve the config in the same way.

There are no inputs to this subtask.

The output for the subtask is:

- GMIC_o_Config
  - A structure variable containing the retrieved Configuration Items

Commented [RB13]: Again, we should mention that this is part of the solution on top of MI. Something like: "The UI flows are the ones created to handle the user interactions. These flows are create as part of the solution that uses the Manual Intervention module"

Commented [AK14]: Captioning and then including the cross-reference to the diagram here would be helpful (for lazy readers like me)

Commented [JL15R14]: Cross Reference should be done already. I've made it bold in case that makes it clearer

Formatted: Font: Bold

Formatted: Font: Bold

Commented [RB16]: Review how this is written, it is a little bit confusing 😉

Commented [AK17]: The name of the config file would be helpful for me as a 1st time user (thought it might be obvious if I had all the files in front of me)

Commented [JL18R17]: Covered in the Deployment Guide, but people may be reading this doc who haven't deployed it - good point

Once the configuration has been retrieved, the process should continue as normal (logging data as it goes).

4.3.2 UI Intervention

The Subtask details are covered in the relevant section under Module Subtasks - UI Actions.

4.3.2.1 MI-WFU-Wait-For-User Subtask (Main Process Flow)

If User Interaction is required at any stage in the process, the process will pause while calling this subtask which logs the relevant data and waits for the UI to be dealt with.

The inputs for this subtask are:

- WFU_i_Log-Structure (required)
  - Logging structure, usually named g_log-str or similar. This stores information about the logging ID fields and any logs to be written throughout the process flow.
  - Note that this variable should be an output of the first logging subtask where the ProcessLog entry is initialised, and should then be both an input and output of every subsequent logging subtask.
- WFU_i_Connection-String (required)
  - Connection string for Logging DB
- WFU_i_UI-Flow-Name (required)
  - Flow name for the end UI flow which will be triggered
- WFU_i_UI-Params (optional)
  - Any parameters in a structure format which should be passed into the UI flow

The outputs are:

- WFU_o_log-str (required)
  - The updated logging structure, to be used in subsequent logging subtasks
- WFU_o_User-Response (optional)
  - Response structure from the UI flow - Required if the UI should return information to the main flow.

4.3.2.2 MI-LUD-Log-User-Data Subtask (Separate UI Flow)

This subtask will be called by the flows that handle the user interaction once it has been completed. This subtask logs the data gathered from the user, and signals to the process flow that the execution can continue. A structure is passed into this subtask to return parameters from the UI flow to the main flow. This can be set using a Create Structure block with data from the UI.

The inputs to this subtask are:

- LUD_i_ConnectionString (required)
  - Connection string for Logging DB
- LUD_i_process-details (required)

Commented [RB19]: The concept of the logging structure is not dealt with before and I find it a little confusing

Commented [JL20R19]: Should this not be handled in the Logging User Guide?

Commented [JL21R19]: Misunderstood - I thought this meant the logging schema but I see you mean that variable

Commented [JL22]: Is this too much information for this guide?

Commented [AK23R22]: No, I think it's useful. Just a little confused by the multiple subtasks - is it the same logging subtask being used multiple times within the MI flow?

Commented [JL24R22]: The logging subtask is used within the main flows, and also within some of the MI subtasks

Commented [AK25]: When is it required and when is it not?

Commented [JL26R25]: Reworded - is that better?

Commented [AK27]: I'd be confused with "usually" when trying to use this, and overthink about when is it required and when not from reading this.

Commented [JL28]: Comment from Asfand:
I'd be confused with "usually" when trying to use this, and overthink about when is it required and when not from reading this.

Should read better now

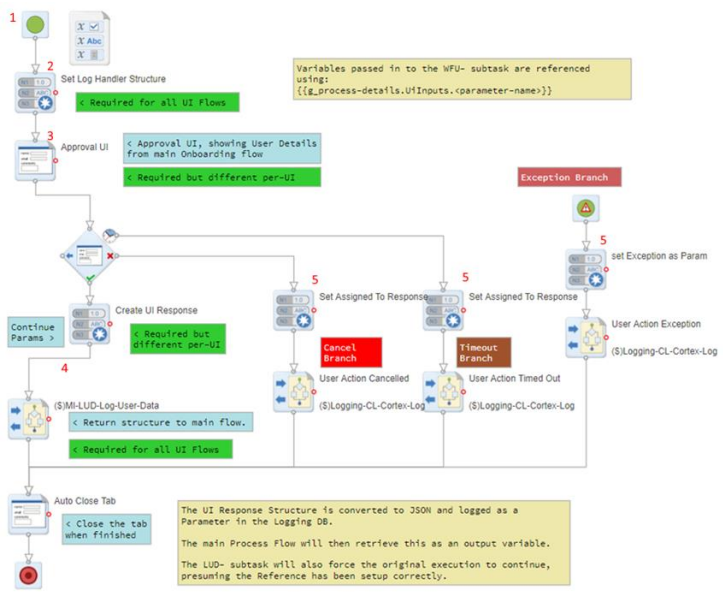
- Structure passed in from the Dashboard flow, usually named **g_process-details**

- LUD_i_ui-data-struc (required)
  - Structure containing any parameters or information returned from UI flow

4.3.3 Configuration of a separate UI Flow

The screenshot below shows an example of the main state of a UI Flow, also included in the studiopkg. This is broken down in the list underneath the image.

For the majority of UI Flows, there will be common elements such as the first state (get config) and some of the blocks and subtasks within the main state.



1. (Previous State) Retrieve Manual Intervention Configuration Items and set Connection String
   a. This is common for all UI flows and can be copied when creating a new UI flow.
2. Use the Create Structure block to initialise the Logging Structure based on the UI Flow inputs.
   a. This is common for all UI flows.
3. User Interaction – configured as required.
   a. This is bespoke for the relevant UI
   b. This example shows a simple User Interaction consisting of 1 UI block before returning the data to the flow, however the UI flow can be made up of

Commented [RB29]: Would it make sense to add circled numbers referring to the list items below?

Commented [AK30]: Missing the "1"
Obvious what it refers to from the list below but for clarity purposes, it'd be useful.

Commented [JL31R30]: How should I show this? Include the State level diagram too?

Commented [AK32R30]: #

Commented [RB33]: Could they create a subtask and call it? We go against our own re-usability best practice?

Commented [JL34R33]: This is mostly a subtask (Get MI Config), and 1 Set Variable block for Connection String. Maybe it would make sense to set the Connection String in the subtask too.

multiple UIs or other processing blocks before returning the data.

4. For 'Continue' branch, set the relevant params in a structure and call the MI-LUD-Log-User-Data subtask
    a. The data returned will be specific for each UI, however the process remains the same.
    b. Most of this can be copied to a new UI flow and modified accordingly.
5. For Cancel, Timeout, and Exception branch, configure the Logging subtask as required.
    a. This is usually generic and can be copied to new UI flows.

Commented [RB35]: Don't understand what this means?

Commented [JL36R35]: A UI flow can have more than 1 User Interaction or perform other checks / subtasks within it, i.e. it does not have to be a simple UI and response. Does it need to be re-worded?

Commented [AK37R35]: What you said above explains it better. I'm not sure what "escalation" means here though.

Commented [JL38R35]: Updated - is that better?

5 Module Subtasks - UI Actions

5.1 GMIC-Get-Manual-Intervention-Cfg

This subtask gets the Manual Intervention config from a file.

5.1.1 Overview

This subtask should be used in any Process Flows, UI Flows, or Dashboard Flows which require the Manual Intervention Config. The default location is C:\Cortex\Config\Manual-Intervention.cfg – if for some reason this does not work, the subtask can be modified to point to a different location.

5.1.2 Input variables

N/A

5.1.3 Output variables

Name	Type	Comments
GMIC_o_Config	Structure	Config Data retrieved from the file

Commented [RB39]: Should we put the parameters retrieved and what they mean? Or is this explained in any other place?

Commented [JL40R39]: Explained in Deployment Guide

5.2 MI-WFU-Wait-for-User

This subtask should be used in the main process flow when a UI action is required.

5.2.1 Overview

This subtask connects to the Cortex Logging Solution to track the requirement of a user interaction. To achieve this, it adds an event to the current stage, marking it as 'Awaiting User Action'. It also sets the ProcessData column to 'UI' to indicate it is pending User Intervention.

The execution of the process calling this subtask will then pause until the User Action is complete - this action is defined in a separate UI Flow, the name of which is added as a Parameter along with any inputs to the UI Flow.

The User Action flow will then log any data as parameters, and then continue the main process flow.

5.2.2 Input variables

Name	Type	Comments
WFU_i_Log-Structure	Structure	The existing log handler ~~so~~ to ensure Event Logs are tied to the correct stage.
WFU_i_UI-Flow-Name	Text	UI Flow Name to be called while the original process execution is waiting.
WFU_i_UI-Params	Structure	Any Parameters for the UI Flow (in the format: 'Attribute: Value').
WFU_i_Connection-String	Text	Connection String for the Cortex Audit Logging DB.

5.2.3 Output variables

Name	Type	Comments
WFU_o_User-Response	Structure	UI data extracted from the DB.
WFU_o_log-str	Structure	Logging DB Handle to be returned to main flow.

5.3 MI-LUD-Log-User-Data

This subtask will be called by the flows that handle the user interaction once it has been completed. It will log the data gathered from the user, and signal to the process flow that the execution can continue.

~~5.3.2~~5.3.1 Overview

This subtask is designed to be used in the standalone UI Flows as part of the Manual Intervention solution.

It should be used after the User has interacted with the UI, and will mark the Action as completed, log any data, and continue the original execution.

This is done by adding the attributes from the structure input variable to the logging database so that it can be read by the main processing flow. The subtask then continues the original execution so it can read this information.

Commented [RB41]: For me it would make more sense to go a little on to the detail of what the subtask does and logs, like in the previous subtask.

Commented [JL42R41]: Added 1 extra paragraph. Is this enough detail?

Formatted: Normal,01 Body Text (Innovise), Space Before: Auto, After: Auto

~~5.3.3~~5.3.2 Input variables

Name	Type	Comments
LUD_i_ui-data-struc	Structure	UI Data entered by the user
LUD_i_process-details	Structure	Details (log, flow name, reference) etc for the Process. This is usually passed into the UI flow in the correct format from the Dashboard flow.
LUD_i_ConnectionString	Text	Cortex Audit Logging DB Connection String.

5.4 MI-LPUI-Launch-Process-UI

5.4.1 Overview

This subtask launches a UI Flow, passing in all the required inputs. These are retrieved from the Logging DB in JSON form.

It can also pass in the username of the person initiating the UI, so the users can be stored in the database for reporting or access control.

- This is used in the Dashboard UI flow, but can also be used as part of another UI flow or to call the Manual Intervention UIs from a 3rd party UI.
- A possible example of this is documented in **Appendix C**.

Commented [RB43]: This is ok from my point of view but the question that now pops in my head is how do I know which processes are awaiting for a user action? If you want to open the door for other UIs, then this should be explained.

Commented [AK44R43]: Agreed - not sure how easy/complex it would be to interface it with a 3rd party UI. Some more info would lessen the uncertainty in trying to do so

Commented [JL45R43]: Maybe another Appendix with a diagram and some details?

5.4.2 Input variables

Name	Type	Comments
LPUI_i_ExternalReference	Text	The External Reference of the ProcessLog entry / the reference of the Cortex Execution
LPUI_i_Process-ID	Text	Process ID from Cortex-Logging DB
LPUI_i_Stage-ID	Text	Stage ID from Cortex-Logging DB
LPUI_i_MI-CFG	Structure	Manual Intervention Config
LPUI_i_FlowUser	Text	Flow API Username
LPUI_i_FlowPass	Text	Flow API Password
LPUI_i_AssignedTo	Text	Optional variable to pass in the Username
LPUI_i_Log-Handle	Structure	Log Handler structure with a connection to the Logging Database

5.4.3 Output variables

Name	Type	Comments
LPUI_o_LivePortal-URL	Text	The URL of the initiated UI – this is used to open the end UI in a separate tab

5.5 MI-SAS-Set-Action-Status

5.5.1 Overview

This subtask connects to the Cortex Logging Solution and updates the current process to set whether or not a User Intervention is required. This is stored in the 'ProcessData' field.

Note that this is already used within the MI-WFU-Wait-For-User and MI-LUD-Log-User-Data subtasks, however it can be used separately if required.

5.5.2 Input variables

Name	Type	Comments
SAS_i_Connection-String	Text	The External Reference of the ProcessLog entry / the reference of the Cortex Execution
SAS_i_Log-Str	Structure	Logging structure from main flow
SAS_i_Process-Data	Text	Action Status type: None, UI, or Email (to be added in a future release)

5.6 SPAG-Set-Process-AD-Groups

5.6.1 Overview

This subtask is used to set a Process to be accessible by members of a certain AD Group

5.6.2 Overview

This subtask connects to the Cortex Audit Logging Solution and updates the ProcessLog table, adding a value to the ProcessUser column. This should tie in with the relevant AD Group name for which users can view the process details and launch the UI.

5.6.3 Input variables

Name	Type	Comments
SPAG_i_SQL-Server	Text	The SQL Server which hosts the Cortex-Logging DB
SPAG_i_DB-Name	Text	The Cortex-Logging database name (usually Cortex-Logging)
SPAG_i_Process-ID	Text	The Process ID of the current process execution to update
SPAG_i_AD-Group	Text	The name of the Active Directory group which should have access to the Process

5.6.4 Output variables

N/A

6 Module Flows

6.1 MI-Example-Onboarding

6.1.1 Overview

This flow is an example for how to use the Manual Intervention. Currently it is a simple flow showing up to 3 UI interactions depending on the data passed in.

1. A random First Name and Last Name are selected, if not provided as an input. A username is then generated.
2. If the Department is not passed in, then the execution will go to a UI Escalation where the user can select the department.
3. If Assets are not provided as a list, then the execution will go to a UI Escalation to select the required assets.
4. Once done, an Approval UI is required.

This is linked with the below flows:

- MI-UI-Onboarding-Department
- MI-UI-Onboarding-Assets
- MI-UI-Onboarding-Approval

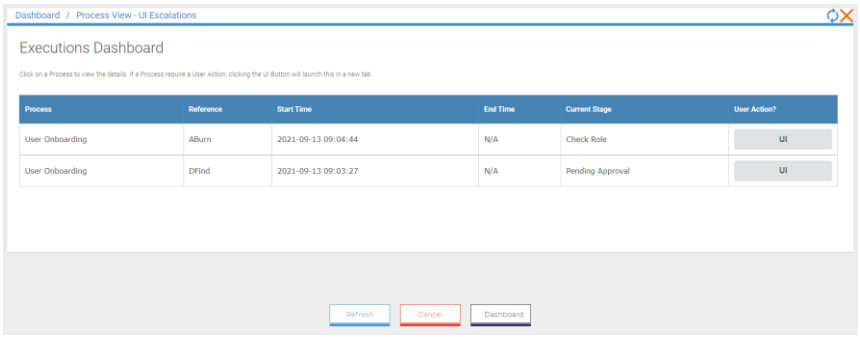
6.2 MI-UAD-User-Access-Dashboard

6.2.1 Overview

This flow is both useable for Manual Intervention and an example of how a Manual Intervention dashboard can be operated. It allows a user to view the processes running in the Cortex-Logging DB. If a User Interaction is required then a button will be shown next to the Process (in the row) which allows the UI to be quickly opened, and the related logs and parameters can be viewed by clicking a row in the relevant grid to drill down.

Dashboard / Process Selection > Process Logs (click to view more) > Trigger UI

If a User Action is required, this can be triggered from main Process view directly. This opens the relevant flow in a new tab.



Clicking on a row in the table will drill down into the Stage / Event / Parameter logs.

Commented [AK46]: Maybe a picture here?

Commented [AK47]: Open the logging DB itself?

Commented [JL48R47]: UI - updated

The Breadcrumbs view at the top left can be used to navigate to a previous page, and the Cancel / Refresh button in the top right can be used to exit the UI or refresh the current view.

7 Appendix A - Manual Intervention Cheat Sheet

What are you trying to do?

Action	Subtask Name	Inputs	Outputs
I want to pause a Process that requires UI Escalation, storing any data to be used in the UI Flow	MI-WFU-Wait-For-User	• Logging Structure • Connection String (Logging DB) • UI Flow Name • UI Params Structure (optional, if data needs to be sent to the UI flow)	Logging Structure
I want to resume the Process Flow from the UI Flow once the user has entered the required data	MI-LUD-Log-User-Data	• UI Data Structure (data to store) • Process Details (passed in to the UI Flow) • Connection String (Logging DB)	N/A
I want to check the process executions and interact with any required User Interactions	MI-UAD-User-Access-Dashboard	This flow can be run directly from LivePortal in the current state. Note that the LPUI-Launch-Process-User-Data subtask can be used to make a custom dashboard flow or to trigger the end User Interactions from a 3rd party UI	

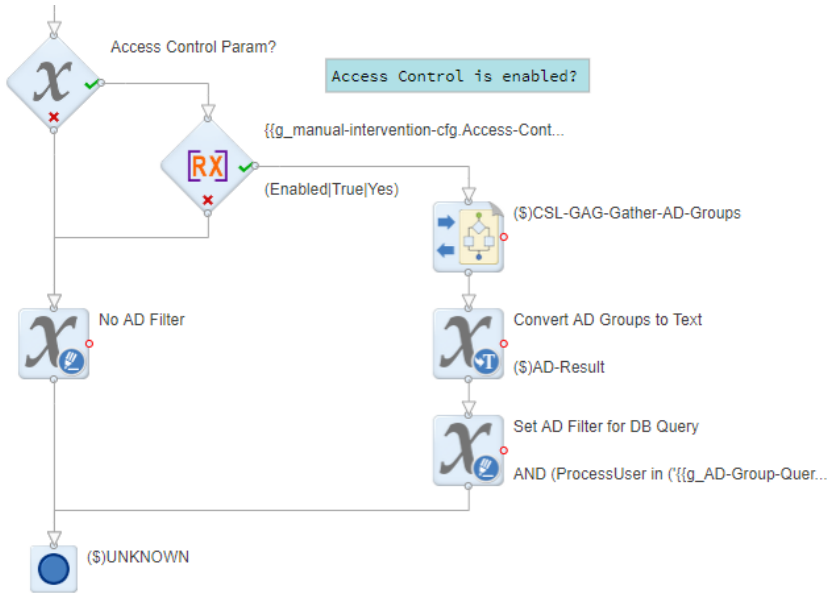
8 Appendix B – Access Control

Access control can be integrated with the Process Logs inside the Cortex-Logging database.

8.1 Access Control in Dashboard Flows

To enable Access Control, the parameters need to be setup as shown in **Appendix A** from the **Deployment Guide**. Access Control needs to be set to 'True' and credentials must be provided for an account which can access the Active Directory.

The AD Groups of the user executing the Dashboard flow is then used to generate a WHERE clause when querying the available processes.



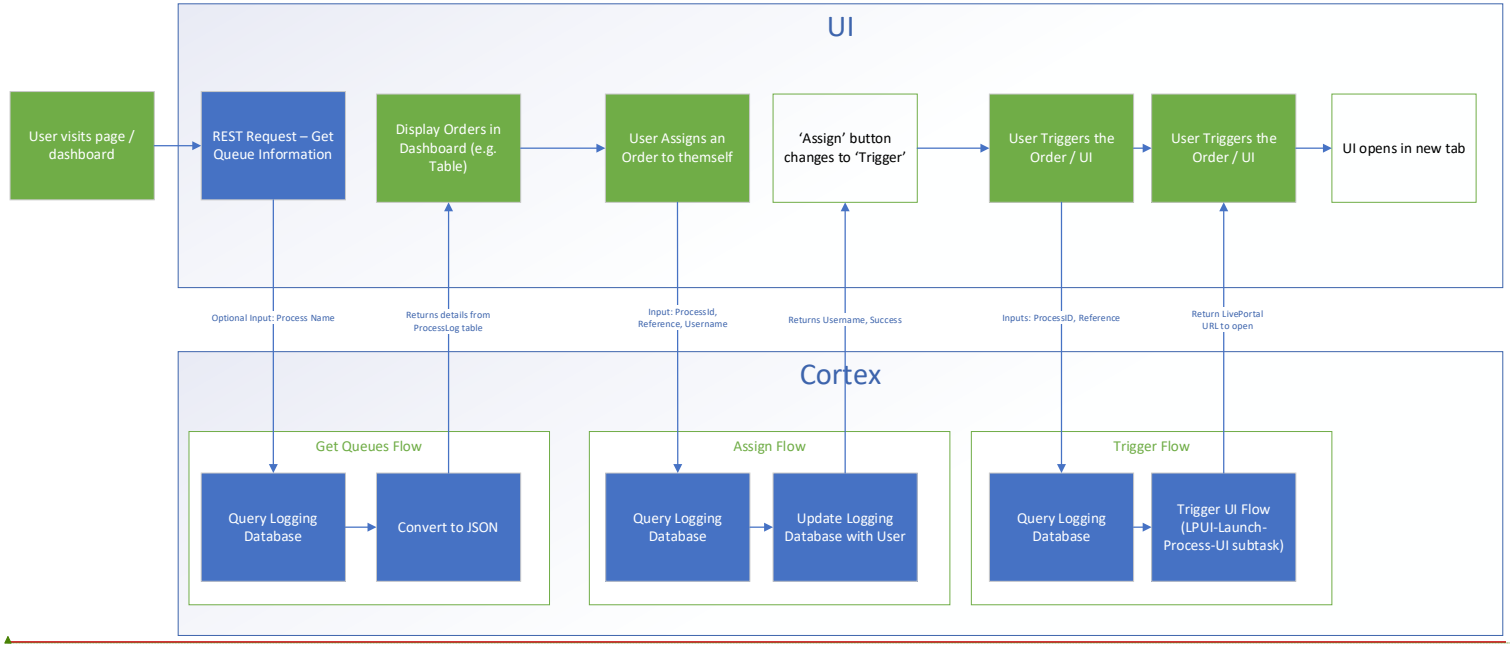
8.2 Access Control in Process Flows

Once Access Control is enabled, the Process Flows can be set as viewable by certain groups. This is achieved by updating the ProcessLog table and setting the ProcessUser column to be the name of the allowed AD Group.

A subtask has been created to handle this – SPAG-Set-Process-AD-Groups.

9 Appendix C – Possible 3rd Party UI Architecture

It is possible to use a 3rd Party UI to host the Dashboard component of Manual Intervention and only relying on LivePortal for the UI Escalations. The image below shows a possible architecture to support this. As the implementation can vary based on the solution, this is not documented further.



Field Code Changed