**FieldServices ServiceReport Dispatcher**

**Process Code Review**

**Videojet**

**Revision History**

|  |  |  |  |
| --- | --- | --- | --- |
| **Date** | **Version** | **Author** | **Description** |
| May 13, 2019 | 1 | Mahesh Kumar | We can use this template for code review. |
| July 03, 2019 | 1.4 | Mahesh Kumar | In “II.1 Detailed Results” section, added column to have recommendations per item. |
| Sep 11, 2019 | 1.5 | Cyril Kuhns | Tied detail “areas” to categories from Scope and Review Result. Added Framework Usage and Program Design detail sections. |
| Sep 17, 2019 | 1.6 | Luiza Surdu-Bob | Review according to the online documentation on best practices. |
| Aug 14, 2020 | 1.7 | Kristopher Kim | Added 3.3 Dependencies section. |

**Table of Contents**

[1. Introduction 3](#_Toc25066858)

[1.1 Purpose of the Document 3](#_Toc25066859)

[1.2 Objectives 3](#_Toc25066860)

[1.3 Scope 3](#_Toc25066861)

[1.4 Pre-Engagement Checklist 4](#_Toc25066862)

[2. Process Description 4](#_Toc25066863)

[2.1 Detailed Results 5](#_Toc25066864)

[2.1.1 Workflow Design 5](#_Toc25066865)

[2.1.2 UI Automation 10](#_Toc25066866)

[2.1.3 Project Organization 13](#_Toc25066867)

[3. Summary 16](#_Toc25066868)

[3.1 Modules considered for review 16](#_Toc25066869)

[3.2 Review Result 17](#_Toc25066870)

[3.3 Concluding Remarks 17](#_Toc25066871)

# Introduction

## Purpose of the Document

The purpose of this document is to discuss the findings of a process review for coding best practices conducted by UiPath for ***Videojet.*** The process Code Review was requested by *Videojet* to be done for the **Process(es)**, as listed below:

* FieldServices ServiceReport Dispatcher

Many of the issues indicate areas of improvements as per programming and design rules described in the UiPath leading practices guidelines (please refer to <https://docs.uipath.com/studio/>), which are UiPath’s prescriptive guidance for developing more robust and stable processes, concentrating on several areas:

* Workflow Design - <https://docs.uipath.com/studio/docs/workflow-design>
* UI Automation - <https://docs.uipath.com/studio/docs/ui-automation>
* Project Organization - <https://docs.uipath.com/studio/docs/project-organization>
* Automation Lifecycle - <https://docs.uipath.com/studio/docs/automation-lifecycle>

The code review focuses on leading practices in the following categories: Modularity, Maintainability, Readability, Flexibility, Reliability, Extensibility, Performance and Security.

In this code review we have gathered metrics, identified, measured, evaluated the issues, suggested approach for improvement and concluded on the current situation, by providing feedback on the efficiency, robustness, maintainability, and design, overall process code quality.

## Objectives

Code reviews are an integral part of the development process and facilitate the creation of applications that are well-designed, well-written and supportable once deployed. The code review documentation contained herein addresses static source code only and makes comments and/or recommendations where applicable. It is important to reiterate that a code review is no substitute for adequate functional and performance testing of an application.

## Scope

The scope of the project is as follows:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **Application name** | FieldServices ServiceReport Dispatcher |  | **Review start** |  | 2021-01-26 |  |
| **Code review owner (Client)** | William Mercier |  | **Review end** |  | 2021-01-26 |  |
| **Number of files (xaml)** | 33 |  | **Code Version** |  | 1.0.1 |  |
| **Code Review Mode** | Remote |  |  |  |  |  |
| **Categories** | Modularity |  | Flexibility |  | Performance |  |
|  | Maintainability |  | Reliability |  | Security |  |
|  | Readability |  | Extensibility |  |  |  |
| **Comments** |  |  |  |  |  |  |

## Pre-Engagement Checklist

The following prerequisites need to be in place before a Process Code Review can start:

|  |  |  |
| --- | --- | --- |
| Prerequisite | Responsibility | Check |
| Supporting documentation (PDD) | Customer |  |
| Customer team allocated with correct skills and training in place (RPA DEV available) | Customer |  |
| Process code shared | Customer |  |
| Sensitive information to be removed or replaced with dummy value(s) | Customer |  |
| Describe the business purpose of use case | Customer |  |

# Process Description

The Process Code Review process consists in the following activities, divided into four principal phases: validation of information received (green), code analysis (blue), prepare report (brown) and final PCR review session (gray):

The issues found are divided into three categories: High, Medium, and Low. Typically, the priority for fixing them should be set according to these categories. The following is a prioritized list of the issues found during the code review. Note, something that is deemed an error by UiPath may not be considered as critical based on specific criteria or goals for the specific process. However, choosing to dismiss this guidance should be evaluated on a case by case basis and a conscious decision should be made to neglect specific issues that have been raised.

## Detailed Results

### Workflow Design

|  |  |
| --- | --- |
| **Hard Coded Values** | |
| **Description** | Do not use hard coded values. |
| **Area** | Flexibility / Maintainability |
| **Impact** | High |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | Avoid using hardcoded values - use config & get environment activity |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
| *SendingForeCastReport -> Forecast Daily report* | *The log message is hardcoded to 2020. Eventually this will be updated to 2021 and that change should not require a code change.* |
|  |  |

|  |  |
| --- | --- |
| **Detailed Logging** | |
| **Description** | Proper Logging helps in analysing the issues. |
| **Area** | Reliability / Readability |
| **Impact** | High |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | * Add logs before and after each module. * If the page load is slow, add logs to confirm the page got loaded in given timeout or not. * Log messages should be detailed enough that a user can easily identify key execution checkpoints and/or failures. |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
| *Login\_Salesforce* | *This workflow has an open browser that is marked as private / the browser is opened as a new private window. What is the reason for this since it will prevent proper logging within this container?* |
|  |  |

|  |  |
| --- | --- |
| **Unused Screenshots** | |
| **Description** | Screenshots of the activities and exceptions. |
| **Area** | Performance |
| **Impact** | Medium |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | Remove unused screenshots. |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Application Specific activities** | |
| **Description** | We should use recommended activities. |
| **Area** | Performance / Reliability |
| **Impact** | Medium |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | We should use application specific activities (for example – Use Open Browser to open browser window, use Excel related activities instead of GUI activities to deal with excel data etc). |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
| *BILogin -> Assign - Active Contracts Download file Path* | *Use available activities instead of hard coding paths. (e.g. Get Environment Folder vs “c:\users”…)* |
| *BILogin -> If-Active Contracts File* | *Use Flow Decisions instead of If activities within a Flowchart.* |
| *Salesforce\_DailyUnitAdds* | *Use Flowcharts when it makes sense to do so. When the steps within a Flowchart are sequential with no decisions, the activities belong within a sequence.* |
| *SendingForeCastReport -> Do -Inside Forecast File*  *ProcessServiceRevenue -> Do – Inside Sheet 3 in Service Dashboard file* | *Use a Maximize Window activity.* |

|  |  |
| --- | --- |
| **Comments/Annotations** | |
| **Description** | Provide comments/Annotations. |
| **Area** | Readability/ Maintainability |
| **Impact** | Low |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | For clear understanding of processing context, describe the processing outline of the workflow at the top of each workflow. Also, add annotations for the activities wherever required.  Use annotations for better readability -> annotations are added for non-obvious behaviors. Should have annotations on all workflows, to better understand their usage and expected values. Describe annotations for lower-level containers (sequences and flowcharts) and activities for others to understand the purpose of each block and assumptions that were made before creating it. |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Activity Names** | |
| **Description** | Provide meaningful names to the activities. |
| **Area** | Readability / Maintainability |
| **Impact** | Low |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | All the activities should have their names personalized (including If activities/action conditions). This would be good from at least several perspectives:   * better understand of what is done * easy to follow and check a certain activity in case of code checks / observations / reviews * in case of errors the stack trace is mentioning the activity name. |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
| *Various workflow files* | *Use meaningful names instead of default activity names. This helps with log readability.* |
|  |  |

|  |  |
| --- | --- |
| **Variable/Argument Names** | |
| **Description** | Provide meaningful names to the variables and arguments. |
| **Area** | Readability / Maintainability |
| **Impact** | Low |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | Variable and Arguments names should be meaningful. |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
| *Various workflow files* | *Use meaningful variable and argument names. Consistency with variable name formats is also important.* |
|  |  |

|  |  |
| --- | --- |
| **Unused Activities** | |
| **Description** | Remove all the unused and commented out activities. |
| **Area** | Readability / Performance |
| **Impact** | Low |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | Activities not connected, in flowcharts (leftovers).  Keep your workflow clean, remove junk. |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Default Values for the variables** | |
| **Description** | Provide default values for the variables. |
| **Area** | Reliability |
| **Impact** | Low |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | Default values should be specified for variables (in order to avoid possible issues with null value variables). |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
| *Various workflow files* | *Whenever possible, add default values to in arguments for unit testing.* |
|  |  |

|  |  |
| --- | --- |
| **Unnecessary Sequences** | |
| **Description** | Keep only the required sequences. |
| **Area** | Readability / Performance |
| **Impact** | Low |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | Remove empty/nested containers (Eg. inside sequence A there is sequence B and nothing else). |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
| *Load* | *Remove nested sequences that do not add any value to the workflow.* |
|  |  |

|  |  |
| --- | --- |
| **Nested if-else** | |
| **Description** | Avoid Nested If-else |
| **Area** | Readability/ Maintainability / Performance |
| **Impact** | Low |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | Avoid nested IFs to improve readability. No more than 2 nested IF activities in sequence - if complex logic is used, use switch or add Flowchart inside the sequence. |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Unreferenced Variables & Arguments** | |
| **Description** | Remove Unreferenced Variables and Arguments. |
| **Area** | Performance / Maintainability |
| **Impact** | Low |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | Remove unreferenced variables/Arguments. |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
| *Various workflow files (e.g. KillAllProcesses -> in\_Config)* | *Remove unused variables and arguments (use Remove Unused Variables button)* |
| *Main -> Process Batch Data -> out\_RevenueFile & in\_RevenueFile* | *Is there a reason why an i/o argument is not used?* |

|  |  |
| --- | --- |
| **Unused Flows** | |
| **Description** | Remove Unused Flows. |
| **Area** | Performance / Readability / Maintainability |
| **Impact** | Low |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | Remove unused flows.  Keep code clean, delete unused content. |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
| *Main -> End Process -> End.xaml -> SendingForeCastReport.xaml* | *End does nothing besides calling SendingForeCastReport. Move the steps from this file into End or vice versa to remove unnecessary workflow files.* |
|  |  |

### UI Automation

|  |  |
| --- | --- |
| **Selectors** | |
| **Description** | Use static and reliable selectors |
| **Area** | Reliability |
| **Impact** | High |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | * Use partial selectors where several activities are executed in the same window (performance, reduces risk of different activities occurring in different windows). If an individual activity is being executed, then it is acceptable to use full selectors. * Make sure that selectors do not reference user/date-specific items or similar which can change. Address using variables or wildcards such as \* or ? * Avoid using index in selectors - try to add more attributes or utilize nested selectors. * Avoid including random values in selectors (selector contains random dynamic values, like ID: 98IGHuiclM81jNC or case specific values etc.). * If an attribute’s value is all wildcard (e.g. name=’\*’) then the attribute should be removed. * If editing attributes doesn’t help, try adding more intermediary containers. |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
| SalesforceOpenWOBilling -> Find Total records Element | Use partial selectors when inside a container. |
| Salesforce\_DailyUnitAdds -> Find Total Unit Adds Element  Salesforce\_FSLContractTermination -> Find Total Records Element | Avoid indexes within selectors whenever possible. (e.g. SalesforceOpenWOBilling also interacts with Salesforce, but does not require an idx) |

|  |  |
| --- | --- |
| **Input methods** | |
| **Description** | Use Simulate Type/Simulate Click Where Possible |
| **Area** | Reliability |
| **Impact** | High |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | * Try using **SimulateClick**, **SendWindowMessages** or **hardware events** (both these properties are unchecked) in the order presented, as the properties are faster and work in the background, but they depend mostly on the technology behind the application. * For background automation, use the **SimulateType**, **SimulateClick**, and **SendWindowMessages** options for navigation and data entry via the **Click** and **Type Into** activities. * Recheck/test if using those properties create any issues. |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Use Containers** | |
| **Description** | Use containers (OpenBrowser/AttachBrowser, OpenApplication, AttachWindow) for several actions in the same screen. |
| **Area** | Performance / Readability / Maintainability |
| **Impact** | Low |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | There are several advantages to using containers with partial selectors instead of full selectors:   * Visually groups activities that work on the same application. * Is slightly faster, not seeking for the top window every time. * Makes it easier to manage top-level selectors in case manual updates are necessary. * Essential when working on two instances of the same application. |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Image Automation** | |
| **Description** | Image recognition is the last approach to automating applications if nothing else works to identify UI elements on the screen. |
| **Area** | Reliability |
| **Impact** | High |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | * Should be the last approach. * Should check if all the environments have same resolution. If not, then how are we handling the scenario. * The resolution should be kept greater or equal than when recording the workflows. * If the automation supports different resolutions, parallel recordings can be placed inside a **PickBranch** activity and the Robot uses either match. * Should use best OCR Engine. * Should prefer short-cut keys wherever possible. |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Background Automation** | |
| **Description** | If an automation is intended to share the desktop with a human user, all UI interaction must be implemented in the background. |
| **Area** | Reliability |
| **Impact** | High |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | * Should be the last approach. * Should check if all the environments have same resolution. If not, then how are we handling the scenario. * The resolution should be kept greater or equal than when recording the workflows. * If the automation supports different resolutions, parallel recordings can be placed inside a **PickBranch** activity and the Robot uses either match. * Should use best OCR Engine. * Should prefer short-cut keys wherever possible. |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **UI Synchronization** | |
| **Description** | There are situations when an application’s state must be validated before proceeding with certain steps in a process. Measures may include using extra activities that wait for the desired application state before other interactions.  Avoid using hardcoded delays (the **DelayMS** property). Use **TimeoutMS** property or **On Element Appear** / **Element Exists** activtities instead of hardcoded delays. |
| **Area** | Performance / Flexibility |
| **Impact** | Medium |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | * Avoid Delay activity if possible and use On Element Appear or Element Exists or similar. * Set correct Timeout MS on screens that take a while to load. * For multi-page Extract Structured Data activity investigate using reduced Timeout MS value. |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
| *BILogin -> Attach Browser -Active Contracts BI -> If- BI Report Export Complete* | *The sequence title does not match the actions. The Else says a 10 second wait, but the config value is 5 seconds. Also, the wait is after the save is clicked vs before like the sequence name suggests.* |
| *ExecuteMacro* | *Instead of a hardcoded 20 second delay, the process should check Excel to see if it has completed the macro and is interactive again. This allows for a longer timeout, but a shorter response time if the macro completes quicker.* |

### Project Organization

|  |  |
| --- | --- |
| **Framework Usage** | |
| **Description** | Starting from a generic (and process agnostic) framework ensures working in a consistent and structured way with any process. It can help starting with the high-level view, then going deeper into the specific details of each process. |
| **Area** | Readability / Maintainability / Reliability / Extensibility |
| **Impact** | Low |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | * Make use of framework-level state machine for error handling and segregation of duties. * Propagate exceptions to Main and handle there. * Separate process workflow code from data extraction and reporting. |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
| *CloseAllApplications* | *CloseAllApplications should attempt to cleanly close open applications. If this attempt fails, then it should fall back to the KillAllProcessess workflow via the framework. The in argument is not utilized.* |
| *Main -> Init* | *If an error occurs with a transaction, there are approximately 3 KillAllProcesses actions taken – once in the RetryInit, once in the Init, and once in the Error. Let the framework handle this.* |

|  |  |
| --- | --- |
| **Project Design** | |
| **Description** | Breaking the process in smaller workflows is paramount to good project design. Dedicated workflows allow independent testing of components while encouraging team collaboration by developing working on separate files.  Use concise expressions and activities.  Keep workflows clean and avoid spaghetti code. |
| **Area** | Readability / Maintainability |
| **Impact** | Low |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | * Choose the layout type wisely (flowcharts and sequences). Normally the logic of the process stays in flowcharts while the navigation and data processing are in sequences. * Use the fewest number of methods/activities needed. * Keep workflows and expressions clean/readable. |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
| *Multiple files in the main directory* | *Organize project files into logical folders where appropriate.* |
| *Various workflow files* | *Use AndAlso/OrElse instead of And/Or to skip checking other conditions when the first condition fails.* |
| *PopulateRunChartsAndForeCast -> For Each – NA\_SheetName -> Sheet Name is Present* | *Use a Break activity when no further processing is required for a particular loop.* |
| *PopulateRunChartsAndForeCast* | *Break down large workflows with a lot of activities into smaller, logical workflows that have singular purposes.* |
| *PopulateRunChartsAndForeCast* | *Use Multiple Assign when appropriate.* |

|  |  |
| --- | --- |
| **Handling Sensitive Data** | |
| **Description** | How are we handling sensitive information in the workflow? |
| **Area** | Security |
| **Impact** | High |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | * Avoid storing sensitive information in queues. * In case, it cannot be avoided, we should avoid logging this information. * Do not use Get Password activity. * Credentials should be stored in   + CyberArk   + Credential Assets in Orchestrator   + Windows Credential Manager |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Use Config File** | |
| **Description** | Keep environment settings in external configuration files or Orchestrator instances, making it easy to run automation in both testing and production environments.  Use config files to store the data/paths which might change. |
| **Area** | Flexibility / Maintainability |
| **Impact** | High |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | Use config files for the below:   * Environmental variables like paths. * Any data which might change. * Assets names. |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
| *SMTP\_Server*  *SMTP\_Port* | *What are these assets used for?* |
| *Jdrive* | *Avoid mapped drives whenever possible. Use UNC paths instead.* |

|  |  |
| --- | --- |
| **Accessibility of Config File** | |
| **Description** | Config file may have some sensitive information. Hence, we need to keep it secure. |
| **Area** | Security |
| **Impact** | High |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | Config file – preferably not accessible to users without permissions and should be deployed together with .xaml files |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Exception Handling** | |
| **Description** | Use Try catch block where the code might throw an exception |
| **Area** | Reliability |
| **Impact** | High |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | * Surround with Try/Catch the code which might throw an exception. * Use finally to close all necessary connections/files/applications. * Proper use of Throw activity. |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
| *KillAllProcesses* | *This does not account for processes started within the user’s session that belong to a different username (e.g. Internet Explorer owned by user SYSTEM). This will potentially cause issues with failing to kill processes appropriately.* |
| *RetryInit*  *End Process* | *If Init fails, these states will fail since Config is empty and calls to values within the Config will produce an error* |

|  |  |
| --- | --- |
| **Recovery/Retry** | |
| **Description** | If the process fails, how is it going to recover or retry or exit with proper logs. |
| **Area** | Reliability |
| **Impact** | High |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | Recovery/retry is planned for errors at different stages of process. |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
|  |  |
|  |  |

|  |  |
| --- | --- |
| **Code reusability** | |
| **Description** | Create and use reusable components or workflows wherever possible. |
| **Area** | Modularity |
| **Impact** | Medium |
| **Review Results** |  |
| **Evidence** | *<Screenshot of activity/file>* |
| **Best practice** | * If same code (Set of activities) is being used at multiple places, create a single component and refer to the same. * If the module can be used for multiple projects, keep it at a shared location. Have separate shared locations for different environments. * Use dedicated workflows to allow fine granular development and testing. |

|  |  |
| --- | --- |
| **Items** | **Recommendations** |
| *GettingFilesFrom\_ShareDrive*  *SalesforceOpenWOBilling & Salesforce\_DailyUnitAdds & Salesforce\_FSLContractTermination* | *Consider reusable code instead of copy/pasting the same actions over and over. (e.g. Path Exists? If…) (for the salesforce files, the steps in the browser are the same, only the selectors change slightly)* |
|  |  |

# Summary

## Modules considered for review

The focus of this code review was on the FieldServices ServiceReport Dispatcher process. To carry out the review, the application files were divided into groups called batches, following the libraries and modules already defined by the application itself.

|  |  |  |  |
| --- | --- | --- | --- |
| **Batch #** | **Batch name** | **# of Files** | **# of Activities** |
| 1 | FieldServices ServiceReport Dispatcher | 33 | 1000 |

## Review Result

This section defines at a high level each of the tests carried out, including the checks performed, the results obtained and any relevant evidence. This is a summary of the detail above, based on the impact of the issues.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Category** | **Findings** | **High** | **Medium** | **Low** |
| Workflow design | Avoid hardcoded values, use private windows only when required (pass UiBrowser variables around to track multiple tabs), use available/correct activities when available, use meaningful activity names, use meaningful and consistent variable and argument names, add default values when possible to allow for unit testing, clean-up nested sequences, remove unused variables and arguments, and remove nested flows that are unnecessary/only call another workflow. | 2 | 4 | 7 |
| UI Automation | Use partial selectors when inside a container, avoid indexes whenever possible, avoid hardcoded delays, and position delays in the correct positions if absolutely necessary (e.g. before or after an activity, but prefer to use a delay activity instead of delays that are hidden inside an activity’s properties). | 2 | 2 |  |
| Project Organization | Use CloseAllApplications and KillAllProcesses appropriately within the framework, use folders to organize workflow files, use AndAlso / OrElse / Break / Multiple Assign when appropriate, breakdown large workflows, remove unused assets from the config file, avoid mapped drives whenever possible (use UNC paths), adjust KillAllProcesses to account for session owned processes that cannot be killed by the user (e.g. system), fix transitions and states that would fail if the config file is not read properly, and create reusable workflows instead of copy/pasting code into multiple files with slightly different inputs/outputs. | 4 | 1 | 7 |

## Dependencies

This table lists the dependencies used by this process and the potential for risk. Red represents critical issues. Orange represents major issues. Yellow represents minor issues.

|  |  |  |  |
| --- | --- | --- | --- |
| **Dependency Name** | **Version** | **Owner** | **Comments** |
| UiPath.Excel.Activites | 2.8.6 | UiPath | Newer version available 2.9.5 |
| UiPath.Mail.Activities | 1.9.3 | UiPath | Newer version available 1.9.5 |
| UiPath.System.Activites | 20.4.4 | UiPath | Newer version available 20.10.3 |
| UiPath.UIAutomation.Activities | 20.10.6 | UiPath | Newer version available 20.10.8 |

## Concluding Remarks

Several highly rated items that should be remedied prior to the next iteration of this process were found. The items that were of highest concern were the hardcoded year within a log message, the use of a private chrome session which prevents proper logging within that container, not using partial selectors within a container, the presence of indexes within a selector, unused assets within the config file, using mapped drives instead of UNC paths, the KillAllProcesses not accounting for processes that the current session user may not have permissions to terminate, and the transition and state that will fail if the config file is not read correctly.

Several Medium and low risk items were found that can be remedied fairly easily. Future enhancements should include changes to resolve the medium and most of the low-risk items, such as improving and being more consistent with the use of comments, annotations, variables, arguments, and activity names, and breaking down larger workflows into smaller ones for readability, maintainability, and potential reusability.

Using helper methods such as Path.Combine should be consistent throughout the project since it is implemented in some areas, but not in others (e.g. BILogin -> Move File Active Contracts to Bot Input Folder; SalesforceOpenWOBilling -> Assign- Download Open Wo Billing File; Salesforce\_DailyUnitAdds -> Assign- Downloaded Unit Adds File; Salesforce\_FSLContractTermination -> Assign- Downloaded FSL Contract File, MovingForeCastFilesToShareDrive, etc.)

The macros were not reviewed, since they were not present within the xlsm files provided.

To conclude, the overall flow of the process is ok, but correcting the items stated above should allow for a cleaner process that is easier to understand and maintain going forward.