UiPath Orchestrator and Its Advanced Features

Himadeep Movva

movvahimadeep@gmail.com
Independent Researcher

Abstract

This research paper will explore the advanced features of UiPath Orchestrator. These features help schedule business processes that can be run in attended or unattended mode. Installation and infrastructure required for UiPath Orchestrator are mentioned. This research paper also mentions roles, database, AD integration, different entities, and robots in Orchestrator.Differences between attended and unattended automation and robot configurations for each type are discussed. Orchestrator Entities such as assets, jobs, logs, queues, machines, and robots are mentioned.The importance of folder structure, reusable components, configuration of triggers, deployment of packages, and configuration of triggers are mentioned.

Keywords: Orchestrator, UiPath, Jobs, Assets, Queues, Machines, Robots, Triggers, Cron, database, API, Active Directory, Tenant, SLA, and Kibana

1. Introduction:

UiPath Orchestrator is a web application that schedules robots to run business processes. It manages the entire robot fleet. For the attended automation, the orchestrator ensures that correct packages are used and ensures centralized management for the setting related to the project. In Unattended automation, Orchestrator ensures proper mapping of packages with processes and lets the users schedule the processes. UiPath has transitioned from an on-prem Orchestrator server to a cloud-based server. This setup is available with steps provided by UiPath forAzure, AWS, and GCP.Virtual machines can be created in the cloud instances to make the machines available to the robot for processing. HTTP methods to the Orchestrator URL are used for all Orchestrator API calls. The format for the Orchestrator URL is https://{yourDomain}. You should use the HTTPS protocol to encrypt the data you submit via API requests.[1]This research paper discusses the following best practices: organization in UiPath Orchestrator, automation best practice recommendations, and attended and unattended automation setup. Also, tenant context and entities in a tenant are discussed, with details on their functioning. The requirements for updating processes, managing them, sending arguments to start the process, and recording a job run are mentioned.

2. Installation of Orchestrator:

OrchestratorCloud can be configured using any leading cloud-based environment: Amazon Web Services (AWS), Microsoft Azure, or Google Cloud Platform (GCP).Hardware configuration requirements must be handled based on the environment's size and build. While similar or the same configuration can be established for development, test, and prod tenants, this would incur additional and unnecessary costs for small and medium-scale deployments. Hence, proper due diligence must be taken while assessing the hardware requirements for tenants.[2] For the Production instance, it is recommended that the following applications be on dedicated servers: Orchestrator web application, SQL Server Database Engine, Elasticsearch, and Kibana. There are three types of deployment. The first one is a Web Server on a Single Machine. This setting is relatively easy to configure as all web servers, orchestrators, and databasesreside on

the same machine.While it is easy to deploy and manage with no additional infrastructure, it is not suited for handling enterprise-level requirements for automation. The drawback is the single point of failure: when the server goes down, Orchestrator will not be accessible. Also, it has limited performance and scalability, is not suited to handle high loads, and has no high availability. Unlike multi-node setup, there's no failover mechanism. However, multi-node deployment in UiPath Orchestrator ensures high availability and load balancing by distributing load across the servers.Disaster recovery setup can also be made using either an active/passive setup or two active data centers. There'll not be any downtime with high availability configuration as there'll be three high availability nodes, two orchestrator servers, a load balancer, and an SQL server listener that runs continuously and is hosted on the SQL server.

During deployment and configuration, it is essential to assign roles and permissions for the users or developers diligently. To prevent anyone from inadvertently or intentionally maligning the system, the roles are recommended to be minimal to require accomplishing a task. Special care must be taken not to grant administrative rights in case of attended automation. Limiting robot permissions and not granting changing machines or robots is recommended. Implementing a strong password policy is crucial, and the default password length, which is eight characters, can be increased owing to security. Encrypting iPath.Orchestrator.dll.config adds another layer of protection. The recommendation is to disable the autocomplete feature to prevent unintended password changes and disable the remember me option during login for better security. Transparent Data Encryption (TDE) is recommended for SQL servers, especially for organizations that need to be HIPAA compliant.When the data is not being accessed, it will always be decrypted so that if someone tries to access the files, logs, or backups, it'll not be allowed as it requires using the decryption key. However, when Orchestrator users log in and do some action, it automatically retrieves the data as decryption happens in the background.Keeping the database clean and tidy during daily operations and maintenance is important. Hence, cleaning up the old data and running regular backups daily and weekly is recommended.

3. Best Practicesfor the Orchestrator:

Orchestration provides multiple features for modeling your deployment, efficiently using assets, including isolation, regardless of size. A singleOrchestrator instance can be split into varioustenants, and nothing will be shared across the tenants. Each tenant can be created for Development, Testing, and Production.Each tenant can be further subdivided into Folders, and there can be as many folders as required to set up the structure. Each folder has settings for configuring and managing entities to manage the automation projects across the firm.Tenants can also be defined for each region of office internationally for the enterprise, as business users may have tailored requirements for automation based on local laws. However, a single robot can only connect one tenant at a time. For ease of development and management, Active Directory integration is enforced, managing users' access to robots and control of access management. Anyone in the Active Directory group (AD) can sign in to the orchestrator without needing to be added manually, as instead of assigning users individually, we grant permission for the AD group. If a user is removed from an AD group, as the data refresh occurs every hour, the user will be removed from the access. However, instead of going through the AD group to assign user permissions, if a user is granted access, even if a user is removed from the AD group, the manually assigned access for the user persists. The manual permissions remain until removed, ensuring that manually assigned user roles are not lost due to changes in the AD group.As discussed earlier, a multi-tenancy option is possible,and all the tenants share the same orchestrator database, enabling the isolation of desired resources from the rest of the organization and accessibility of automation resources from only within that tenant. Folders enable users to limit automation administration to a corresponding project folder. Every time a robot is provisioned, administrators must choose the type of

robot: Unattended or attended robots. While unattended robots need Windows credentials to run unattended jobs on them, attended robots don't need credentials because the jobs are triggered manually by human agents directly on the machine where the Robots are installed. After registering the Robot, the next step is to check whether the Robot is availableon the Robots page. In the process, any old version should be deleted. This can be achieved by using the Delete button that deletes a certain version selected or the Delete inactive button that deletes all unused versions. Keeping at least one previous version in case the current version fails is recommended. A robot can be triggered on multiple processes, and while a job is in progress, other jobs will be in the queue and will get picked up after the robot finishes the first job. Queues are used to process each transaction, and at the end of processing, it is mandatory to set the status of the processed transaction item as success, Business exception, or System Exception; otherwise, transactions with NEW status are abandoned after 24 hours.

Unattended automation is performed by Unattended Robots, which are deployed from Orchestrator and set to run on a certain configuration. They run in the back end, so users never need to check or trigger them manually.[3] As mentioned in the Figure 1, once the project is developed in Studio and published to the orchestrator, create a process and map robot for this process. The orchestrator sends the job information to an unattended robot. The robot continuously communicates with the Orchestrator using a heartbeat mechanism. Robot accounts are needed to run unattended processes that are not the responsibility of any user. These are known as service accounts that can meet specific requirements for credentialing without depending on a user to run background processes.Assigning roles for a robot in a folder lets the robot account automatically get access, and a new folder can be added as a subfolder. While the permissions can be customized for the robot to the subfolder, the permissions are also inherited from the parent folder. It is possible for a robot account to have greater access at the folder level than at the parent folder level. If a robot needs to access only a subfolder, then assign the robot only at the subfolder level, making the robot invisible to the parent folder. Also, please note that this assignment doesn't make the robot available to other child folders unless the robot is assigned to all of them. If a process requires access to another folderbesides the one where the robot is defined, the robot account must also have access to that folder. Hence, in this case, assign the robot account to those folders and provide the necessary permissions.One option is to group the robots under a specific group so that when a configuration for a few robots needs to be changed, the group setting changes accomplish the change instead of changing configurations/permissions for each robot account. Service mode is preferred for unattended automation as it doesn't require user login, supports automatic session management, allows robots to start and stop without human intervention, and works well for concurrent jobs on Windowsservers. Instead of defining a machine for each robot, an efficient way is to define a machine template, and once it is configured, multiple host machines can connect to the Orchestrator using that template. Simply put, a machine template acts as a blueprint for a group of machines in the same group.When running unattended automation, Orchestrator chooses any available machine assigned to the machine template. Runtimes are licenses assigned to machines to run unattended jobs,and they're assigned at tenant level and pooled for machine usage. Once a job is completed, the machine disconnects, allowing the runtimes to return to the pool for other machines.

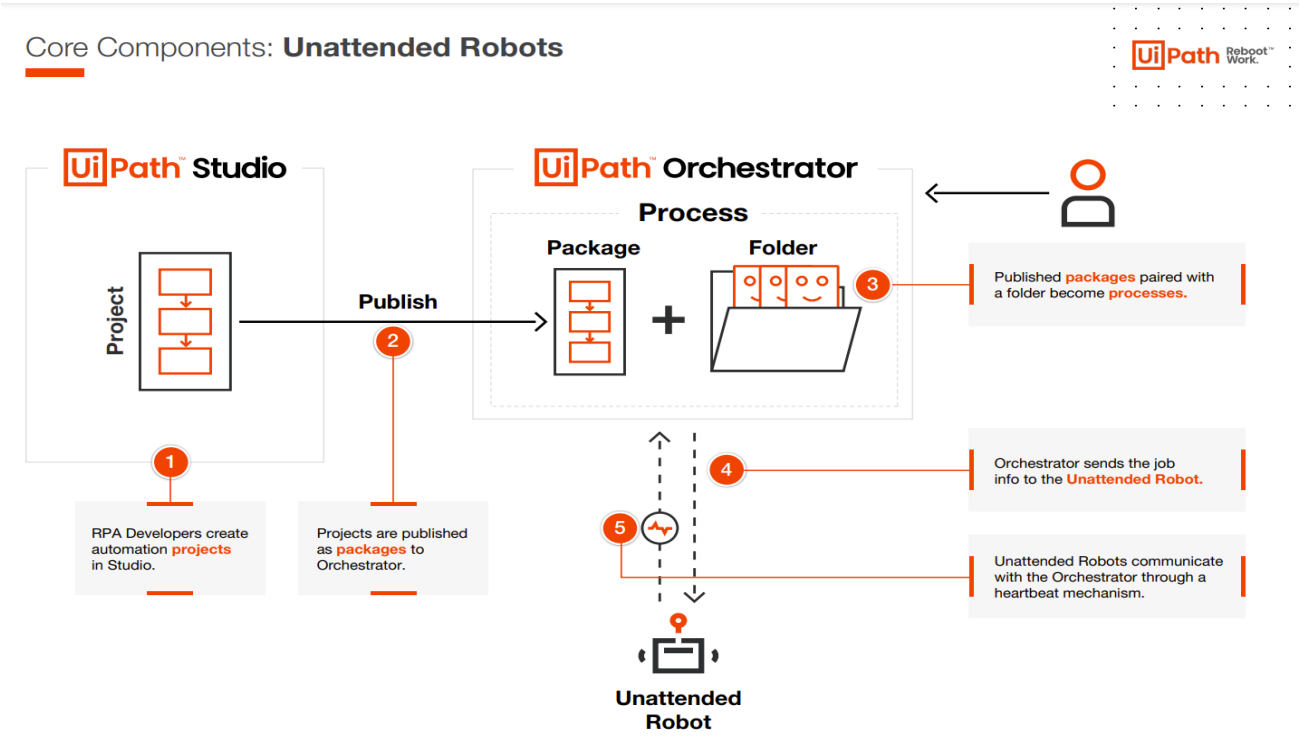

Figure 1- Unattended Robot Communication with orchestrator

4. Entities and features available at the Tenant level:

Here are some of the entities in the Orchestrator tenant: Robots, Folders, Manage Access, Machines, Monitoring, Packages, Audit, Credential Stores, Webhooks, and Settings.We can search for objects in the tenant by clicking the search icon on the upper right side of any page or using the keyboard shortcut Ctrl+\ in Windows. Clicking search lists all the resources available for the folder and tenant. Objects can be filtered for view using the type filter and, based on edit accesses, can be modified. Also, filters can be applied based on created labels.Administrators can automatically create robots for users or groups in the UiPath Orchestrator from the Manage Access page. They need to turn on the attended or unattended robot option for a user or group, enabling the creation of a floating robot with configured settings. For users, the automatic creation of attended robots happens when they log in to the orchestrator. However, unattended robots cannot be auto assigned and must be manually assigned. For attended setup, createa machine template, connect UiPath Assistant to Orchestrator using the key generated by the machine entity, and start the job using UiPath Assistant.First, create a Machine Template for unattended setup and assign a certain number of runtimes to the template. A template with n number of runtimes sets aside n licenses for each machine connected to the Orchestrator using that template. Using this setting, it is possible to run concurrent processes on each defined machine using this Machine template. For example, certain production and nonproduction licenses exist in the tenant's pool of licenses. Let's say we assigned three production and two unattended licenses to a machine template, and three workstations are connected using that template. Each machine will consume three production licenses * 3 workstations = 9 unattended licenses and two unattended licenses * 3 workstations = 6 attended licenses from a pool of licenses in the tenant. This configuration enables us to run two unattended automations and three unattended automations on each machine concurrently. To optimize infrastructure and maximize cost efficiency, UiPath provides Process Type and Process Complexity settings during the configuration of unattended environments. The two key types in Process Type are foreground automation and background automation. Foreground automation requires interaction with the UI layer and can run only on Windows-based environments. Background

automation runs without requiring UI interaction and can run both on Windows and Linux machines. Process complexity defines where a process can be executed. Different options are Windows-legacy and Windows, which can only run on Windows machines, and Cross Platform, which supports execution on both Windows and Linux. The recommendation is to assign cheaper Linux licenses for cross-platform background processes, windows machines for foreground automation, and high-density Windows servers for multiple concurrent automation. Your UiPath unattended automation architecture can be optimized by properly selecting Process Type & Process Compatibility, which will balance performance and cost while guaranteeing that the right processes run on the right machines. Unattended jobs in UiPath are automatically assigned to machines and accounts. If the account is mentioned, only the machine will be assigned, and if no account is mentioned, both the account and machine are automatically paired.[4] Each Windows user account must be provisioned on the machine where the robot will run.As each user corresponds to an account in Orchestrator, the account must be provisioned both in Orchestrator and on the Windows machine to execute jobs in a folder.For example, a server is connected to Orchestrator using Machine Template and is assigned to two folders: admin payroll and employee payroll. Six user accounts are created in the orchestrator and are assigned to these two folders. To allow these six accounts to execute automation, each of them must exist as a user on a Windows machine, with 6 separate Windows user profiles on the server, matching the accounts assigned in Orchestrator. To run unattended automation efficiently, every user that is assigned to folders must also have accounts on the host machine. Graphical representation of modern unattended setup is provided in Figure 2- Modern Unattended Setup.

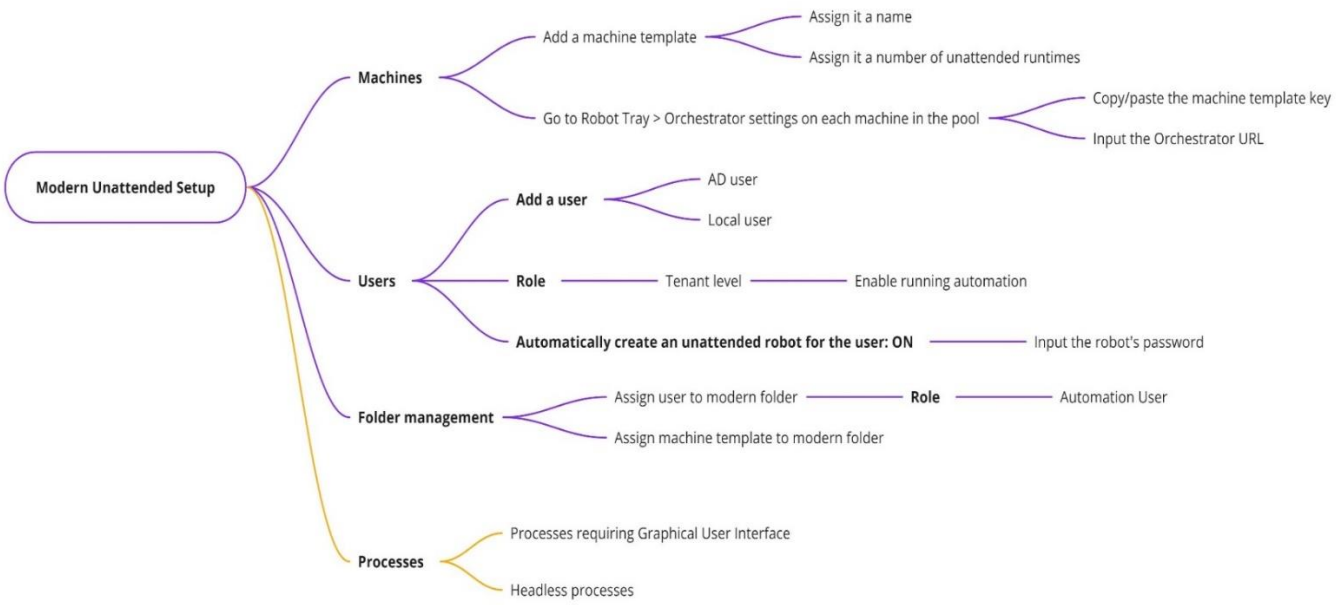

Figure 2- Modern Unattended Setup

In UiPath Orchestrator, administrators can remove disconnected or unresponsive sessions for better and more efficient maintenance of the environment.This practice declutters the environment, displaying only active sessions.A disconnected session occurs when a robot is gracefully stopped,and an unresponsive session happens when the robot fails to send four consecutive heartbeats. Permissions are needed to delete unattended sessions, and only sessions on which triggers are not set can be deleted. Go to Tenant > Monitoring > Unattended sessions and deleteindividual sessions or multiple sessions at once. For community users, disconnected or unresponsive sessions older than three months are automatically deleted, and for enterprise users, such sessions will be removed after six months.

Folders separate the projects and enable fine-grain control over the entities in a specific automation folder. Folders have their own Orchestrator resources.Global resources are created at the tenant level andare also available for folders. Folders with a dedicated package feed are marked with folder package labels. It may not be visible at first, but upon refreshing the orchestrator, it can be seen.Admin needs to assign the user access to one or several folders and based on folder-level permissions assigned for each folder, the resources can be accessed. The folder hierarchy can be up to 7 folders, with 6 sub-folders under the parent folder.[5] User access is inherited from the parent folder. For example, if a user is given access to the parent folder, all its sub-folders can be accessed by the user,too. Usingthe Orchestrator path in the Get Asset activity, resources from other folders can also be accessed by Robot. By giving '/' at the beginning, the path starting from the root folder in the hierarchy will be searched. The path starting with '.' starts from the ambient folder,and the path starting with ' ..' starts one level upthe tree structure. The expression '../../ ' searches two levels above the tree. Folder permissions to create, delete,or edit exist for both parent and subfolders. While creating a folder, there will be two options to send the feed: tenant feed and package folder feed. Packages published to the tenant will be available in all the folders, but packages published to a folder are available only in that folder. Moving folders can impact processes, package feeds, and machine access. Hence, it is always recommended to check dependencies before moving folders and move only if it's required in order not to miss out on any dependencies, making the process break. Also, be careful with deleting folders as the action may remove its resources and impact ongoing automation.Once the folders are configured, folder administrators manage the entities Orchestrator administrators may need to intervene only in case of creating new folders for new automation. Instead of individual users, access is suggested to be controlled using AD directory users or groups. This way, manually updating each user by the orchestrator admin can be eliminated whenever a user leaves the department or organization. Personal workspaces are folders used by individual users to maintain attended automation. You can provide particular people or groups access to their own workspace so they can operate attended robots. The packages within personal workspaces are available to only a particular folder and can't be accessed elsewhere as they're defined outside of Orchestrator tenant feed. Any package that is added to the workspace automatically gets added as a process in the workspace.With no Orchestrator access or actions required, the user can start their automation nearly immediately after publication because each package submitted automatically creates a matching process or updates existing packages and processes[6].

Tenant-level monitoring is available for users to understand tenant-level performance overview of jobs. The options available for the dashboarding are Overview, Machines, Processes, Queues, SLA, License (for admin view only), unattended sessions, and user sessions. The permissions can be customized even to view per object. If permission is not given, the view option for a specific object will not be available for the user. Ideally, the view option needs to be enabled for monitoring machines, queues, jobs, and queues. The overview ribbon has Job Status information: Number of jobs that are running, pending, stopping, terminating, suspended and resumed. It also has a Job History that depicts successful, faulted, and stopped jobs.- The overview page has Transaction status at the bottom tile, mentioning a number of business exceptions, Application Exceptions, and Successful records inQueue.The machines ribbon under monitoring has two sections, real-time and historical, letting the users view available machines in a chart, and errors are displayed for each machine in a chart. Processes ribbon has information about completed jobs, failure reason, and processing time of jobs. Queues ribbon also has real time data and historical data about the status of queues in each process. SLA ribbon depicts any processes that didn't meet SLA in case if its configured. Unattended sessions ribbon has machines and a number of runtimes that are assigned for each machine.

The packages page displays allpackages uploaded to Orchestrator either from the studio or manually.Packages have a published date, which signifies the date on which publishing happened.However, if packages are published to the user's workspace or a specific folder, they'd not be visible on the tenant packages page. Packages in the tenant feed can be accessed in all the folders. Feed settings and configuration through settings->Deployment applies to tenant feed only. Only root folders can be configured with package feeds at the folder level, and access can be controlled by folder-specific permissions set. Folder feeds can be only for processes and not for libraries. A personal workspace has a dedicated personal feed available to only that workspace.As mentioned, packages added to personal workspaces are automatically deployed as processes in that workspace context. A package has two statuses: Active, which means it is currently to a process or a folder, and Inactive, which means it is not currently deployed to a process or folder.Packages can be inspected from the Packages ribbon, and all the available versions will be displayed in package versions. There's also an option to compare different package versions by clicking compare packages, selecting versions that need to be compared, and clicking compare.This comparison lets the users know what the modifications are in different package versions. Thisscreen also has variables, arguments, and imports, just like in the studio. In this screen, flowcharts are displayed similarly to standard workflow diagrams. Tags can be added to the package versions during design time, and they'll appear when uploaded to the orchestrator. However, we cannot modify tags in Orchestrator. Tags can only be modified at the design stage itself.

Library components can be created to reuse the workflows in different automations. There can be multiple ways to create and reuse components. One such way that is recommended is creating a xaml and publishing it as a library to Orchestrator tenant. This makes the developers connected to that orchestrator instance to install that nuget package and use it in the workflows. The NuGet package will be uploaded from the host feed to the tenant feed. Publishing it to the host feed makes it available to all the tenants in the orchestrator and publishing it to the tenant feed makes the package only available to that tenant. Library components can also be published to the UiPath marketplace.

Job represents the execution of a process on UiPath Robot. A job can be launched either in attended mode or unattended mode. While unattended jobs can be started from Orchestrator, attended jobs need to be manually triggered or triggered based on certain user triggers. To start a job on a process, Click Start job, select process in the folder from the dropdown, select job priority by choosing among various options available, select runtime, select execution target, mention the number of times this process needs to run, select the robot account and select t the machine in which the Job runs. There are certain options while creating a job. Scheduling the end of the execution option is useful if a job needs to be stopped after a certain time due to SLA. It gives the option to stop after certain days, hours, minutes, and seconds.When starting a job or setting a trigger, you can assign specific accounts to machines, ensuring that unattended automation runs on designated machines for better control. Whiletenant-based mapping works across all folders, folder-based mapping is restricted to a specific folder. When starting a job, if the user doesn't specify the machine or account, Orchestrator dynamically assigns execution to the first available resource, maximizing resource allocation, handling high-volume jobs, and requiring no manual assignment. This is especially useful in environmentswith many robots and parallel needs to happen. You can decide which job runs first when multiple jobs are waiting by setting Job Priority. Priority can be set when: deploying a process or configuring a job or trigger. The following sequence of steps is followed while allocating Orchestrator Jobs: a robot becomes available, Orchestrator identifies the machine template assigned to that robot, Orchestrator evaluates all pending jobs in the folder, and Orchestrator assigns the job with the highest priority to the robot. If multiple jobs have equal priority, Orchestrator assigns them to robots based on when they were created - the oldest job gets processed first, followed by newer ones in time order. If you start a

job on multiple high-density robots on a Windows server, several jobs are created that run on the same machine. This distributes the workload, and processing can be made faster instead of job running on a single robot. If high-density robots are used, RDP must be enabled; otherwise, the following error is displayed: “A specified logon session does not exist. It may already have been terminated.” If the recording is enabled during the last minutes of failure, the bot process will be recorded. Recording can also be downloaded and analyzed.By giving a stop command, the job will stop gracefully instead of abruptly killing. The time when a job should stop after giving the stop command can be configured. There is an option to make the job stop after a certain time or kill the job after a certain time. Job alerts can be set in Orchestrator for pending or resumed jobs and for jobs that are running too long.For example, if a job remains in pending or resume status beyond a certain time or a job doesn’t complete execution within a set time, the alert will be triggered, helping monitor and troubleshoot stuck or long-running jobs efficiently. Different states of a job are pending that convey that the job is queued to start, running, successful,faulted, stopping,terminating, suspended, resumed, or stopped.

Triggers, which are like cron jobs in Linux, enable the jobs to run at the scheduled time. The triggers page enables users to create future triggers or launch a job on an existing process.[7] There are two types of triggers, namely time triggers and queue triggers. Triggers can be created outside of the workflow in the Orchestrator using two types of triggers mentioned or inside the automation by thedevelopers during design time, using trigger activities. There’s a limit of only one trigger activity per workflow. If a trigger is designed during design time while publishing to the orchestrator and creating the trigger, it should be configured. If it’s skipped while adding it at process creation time, the trigger will not work properly even if triggered in the workflow while running. Also, in the case of Queue-based triggers, If the trigger is not added at process creation, the automation will not start automatically when new queue items arrive. In the tenant, it is possible to define nonbusiness days so that triggers will not run on those days. Also, the calendar can be defined for non-working days, and there’s an option to define only one calendar. Editing a calendar will affect the triggers that are already based on that calendar. Per process, the type of trigger launches a certain number of jobs per job. For example, if a trigger with 2 jobs is launched on a process and a second trigger with 3 jobs is launched at the same time, as 2 jobs are already running, 1 job will be launched by the second trigger. The second type of trigger is based on per trigger, which is based on number number of jobs that need to be started off on a process. For example, when a trigger is set off to launch 7 jobs, and let’s say 3 jobs have been just completed another trigger. In this case, the trigger will launch 3 jobs in addition to the 4 jobs. The third option is setting no limit on the number of jobs launched. If a trigger is set off to launch 5 jobs,irrespective of any checks, including a check of how many jobs are running on the process, 5 jobs will be initiated. When using time triggers, the exact second at which the trigger was created affects its future executions. This is the result of Cron value. For example, the trigger created at 10:30:22, with Cron expression: 5 * * ? * * runs every minute at the 5^{th} second. For example, to run a process every 4 hours, Cron needs to be set as 0 0 */4 * * ? and the trigger of run happens at 00:00 AM, 04:00 AM, 8:00 AM, 12:00 PM, 04:00 PM, 08:00 PM, 12:00 AM(next day). By default, if a trigger failed 10 times consecutively and it wasn’t successful even once in the past 24 hours, the trigger will automatically be disabled. In case of long running workflows, try setting the trigger to run less often.Orchestrator checks every 30 minutes for unprocessed items in the queue in case of queue triggers and can be modified by using Queue.ProcessActivationSchedule parameter.

Logs are created for every job that’s been run and they’re helpful to understand if the process is run correctly or where failure occurred.[8] The following descriptors exist for the logs: time, level, process, hostname, hot identity, and message. Time is the time stamp on which it is logged, and logs can be sorted and filtered using time.The level determines the severity level, with options trace, debug, info, warn, error,

and fatal. The process would be the name of the process, the hostname would be the name of the workstation used for execution, and the host identity is the identity under which the execution takes place. When the Orchestrator is down, the status of the job is stored only in the UiPath Robot service and cannot be synced with Orchestrator. However, once the Orchestrator becomes available, the status will be synced. However, if the Robot service is restarted while the Orchestrator is down, when the Orchestrator is up, the job will be restarted. Hence, to avoid duplication of running jobs, it is recommended not to restart the Robot service when the Orchestrator is down. Logs can be stored either in the database, elastic search or both. The performance of the SQL server degrades once 2 million robot logs are reached. Depending on the size of the database server this limit could vary. The degradation leads to slow log searches and impacts the performance of automation. Hence, it is needed to regularly clean up the logs, freeing up the disk space. By default, logs of the previous day are only shown, and it is required to manually change the time filter to see past logs. Using elastic search enables users to search logs using name. By default, using multiple keyword searches, the OR method will be applied. [9] For example, when a search is made using the keyword 'sent file,' the work sent OR the word file will be searched. However, to do a search using and condition, the search must be made using 'search AND file'. Along with the default log fields, user-defined log fields can be defined using the Add Log Fields activity, generating the additional fields for logging after that activity is executed unless they're removed using the Remove Log Fields activity.

5. Conclusion:

In conclusion, Orchestrator provides user-defined functions, roles, and features to maintain and scale the automation in a reliable and sustainable way. It is a one-stop solution for every configuration related to RPA processes except the development. Required infrastructure setup, including attended and unattended automation, can be customized and defined in the Orchestrator, letting the administrators configure processes on the fly once developed. It also lets the users effectively use the entities such as Queues, Processes, Assets, Jobs, triggers, and logs. Reusable components can developed and published to Orchestrator tenant. Once the developers are connected to the tenant, they can utilize the component, saving a lot of time in development across the projects in a firm. Orchestrator plays a key role for the support team in monitoring jobs and job processes. In case of any failure, upon analyzing the log files, the support team could figure out the root cause for the failure and fix it. Automated scheduling of jobs can be made using triggers, eliminating the need for manual runs. Different automation can be maintained in a tenet using different folders. Each automation will be assigned a dedicated folder to define assets, queues, packages, triggers, and machines. Also, each department can be configured to have different automations in a firm. Using elastic search, logs can be indexed, making them searchable and filterable. This enables real-time insights for jobs. While a relational database takes some time to query, non-relational databases work very well for quick querying and visualization. Using Kibana, Splunk or any best visualization tool applicable, the results can be queried from elastic search, providing users with monitoring of robot performance, error tracking, and audits.

References

[1] [Online]. Available: https://docs.uipath.com/orchestrator/standalone/2023.10/user-guide/organization-modeling-in-orchestrator. [Accessed November 2023].

[2] [Online]. Available: https://docs.uipath.com/orchestrator/standalone/2023.10/installation-guide/orchestrator-hardware-requirements. [Accessed November 2023].

[3] [Online]. Available: https://docs.uipath.com/orchestrator/standalone/2023.10/user-guide/unattended-automation. [Accessed November 2023].

[4] [Online]. Available: https://docs.uipath.com/orchestrator/standalone/2023.10/user-guide/managing-robots-modern-folders. [Accessed November 2023].

[5] [Online]. Available: https://docs.uipath.com/orchestrator/standalone/2023.10/user-guide/folders#tenant-and-folder-resources. [Accessed November 2023].

[6] [Online]. Available: https://docs.uipath.com/orchestrator/standalone/2023.10/user-guide/job-states. [Accessed December 2023].

[7] [Online]. Available: https://docs.uipath.com/orchestrator/standalone/2023.10/user-guide/about-triggers. [Accessed December 2023].

[8] [Online]. Available: https://docs.uipath.com/orchestrator/standalone/2023.10/user-guide/about-logs. [Accessed December 2023].

[9] [Online]. Available: https://docs.uipath.com/orchestrator/standalone/2023.10/user-guide/managing-logs-in-orchestrator. [Accessed December 2023].