Downtime Planning

Background

The promise of automation is achieving more with less. As organizations build more automations, the impact on operations grows accordingly. The catch? When an automation breaks or undergoes maintenance, it becomes unavailable to operations. Downtime plans are required for every automation built, as operational teams have adjusted their operations with the expectation that the automation will be running. Automation leaders can overcome this challenge with robust downtime procedures, preparing for known upgrades, and even the unexpected.

Overview

This guide provides guidelines for automation teams establishing downtime procedures with operations teams, managing automatons in the face of application or EMR upgrades, and tips for organizing your team around the unexpected. Whether you're a program manager, solution architect, or developer, this guide empowers you with tactics to navigate and overcome the evergreen challenge of RPA downtime.

General Downtime Recommendations

Below is a list of activities that every organization should deploy regardless of whether the downtime is anticipated (e.g., application/EMR upgrade) or unanticipated (e.g., an insurance portal changes their Graphical User Interface). In either case, the automation team is going to have to inform operations, triage, and fix the issue.

- Automation teams should set expectations with operational teams that automations *will have downtime* and the process being automated is *still their responsibility*
    - Automations should not be built without this alignment
    - Organizations can consider having a statement/disclaimer incorporated into a Process Definition Document (PDD) that requires sign-off
- Operational teams should have access to the automation work queues so no administrative work is required for operational teams to intervene when needed
- Automation teams should establish automated emails to operations triggered when the automation doesn't run
    - We also recommend sharing the volume and percentage of accounts that the automation worked daily in case automation issues cause backlogs that need to be worked manually
- Automation teams should establish a triage level rating for the automation based on how long it will take to fix the issue, an example is as follows:

    - Level 1: We expect the automation to be down for 1-2 days
    - Level 2: We expect the automation to be down for 2-7 days
    - Level 3: We expect the automation to be down for 7-14 days
    - Level 4: We expect the automation to be down indefinitely
- The automation team should assign a triage level within 24 hours of an issue being identified
    - This requires the automation program to adjust their sprint plans with maintenance requests
- Operational leaders are responsible for writing clear downtime procedures for each automation that align with the triage level that gets assigned by the automation team
    - For example, Level 1 requires no action, while Level 2 requires staff to cover from a different workflow team
- Automation teams should send daily status updates to operational teams, specifically calling out any status changes or resolutions
- Automation teams should also talk to operational leaders personally to keep them informed and to maintain trust and transparency
    - Commonly, automation teams will need to work with the operational Subject Matter Experts to diagnose and triage the break
- Downtime plans should be documented in a place accessible to the automation and operational teams, such as an intranet site

Known application/EMR upgrade

Automation teams will have to navigate situations known to cause downtime. Application and EMR upgrades are the two most common scenarios, with the EMR upgrade commonly causing the largest amount of maintenance activity.

As a general policy, the automation team should be blocking the two weeks before and after an application / EMR upgrade to solely focus on reducing downtime. That means that new builds, enhancements, etc. should be deprioritized.

Some general asks to make of your IT department:

1. Have a representative from the automation team join application / EMR meetings
    a. Alternatively, have an application / EMR representative join a standing automation program meeting
2. Have the automation team included on the application / EMR release notes
3. Ask for the automation team to have early access to the application / EMR test environment (as early as allowed by the organization)
    a. This will allow the team to view, test, and understand (with precision) how upgrades will impact your automations as soon as possible

Below are the operational steps the automation program should take to transition existing automations to a new, upgraded application or EMR.

1. Isolate the automations that are going to be impacted
2. Prioritize the isolated automations based on their operational impact
    a. Tarpon recommends focusing on the automations with the highest financial and labor impact
3. Build a sprint based on the established priority and assign goals with your development team
    a. Identify and secure resources outside your team
4. Execute as much of the sprint ahead of the application / EMR upgrade as possible
5. Communicate anticipated downtime with operational teams (see above) as soon as possible so that operational/staffing adjustments can be made

Unanticipated Issues

Unanticipated issues are very common in automation for a myriad of reasons, and are generally out of the control of the automation team. Common examples include changes on the Graphical User Interface (GUI), expired credentials, external website throttling, and unannounced application/website downtime. Below is a list of activities automation teams can deploy for unanticipated issues.

- Proactively allocate time to enhancements/maintenance into every sprint
    - Developers can adjust the allocation as needed
- Development teams should review whether the automation has run daily
    - Most automation platforms have pre-built triggers indicating whether the automation has run
    - This can be an established norm and productivity expectation
- The development team should set aside time to review the automation dashboard to identify trends in skip rate, error rate, execution time, etc.
    - See our guide to automation metrics
- Staff should be educated on submitting tickets if they notice any issues or get any relevant communication from applications/websites that the automation is using