THE WORLD'S #1 CONTINUOUS TESTING PLATFORM

ServiceNow upgrade checklist: A step-by-step guide to improving the release process

Tricentis

Table of Contents

Executive summary

ServiceNow updates can be time-consuming and risky. From system updates to extensions of the user interface, even seemingly minor changes can significantly impact system behavior. Due to the Now platform's extreme versatility, it is nearly impossible to test manually, especially in broad installations with equally broad testing requirements. Traditionally, developers and testers work independently to ensure the code is working as expected. Thorough testing takes time and patience. The back and forth, human errors, and constant effort to capture fundamental changes to the core system to avoid defects are very difficult to manage through manual testing.

Adopting test automation can provide consistency and peace of mind. Test automation provides immediate feedback on a breaking change and saves testing time over the long run. To control the risk of the next ServiceNow upgrade, establish a reliable safety net with an effective test automation strategy. Ideally, this strategy:

- Includes multiple levels of testing to be used at all stages of development
- Defines how requirements should be used to create tests and how those tests should be organized and maintained
- Outlines how and when tests should be executed and how defects should be logged from the results

Planning

Start with an overall strategy. Decide which applications or areas of ServiceNow is the most important to test with automation. This may mean areas of the system which are the most impactful to users or areas that take the longest time to test manually. Remember, automated testing can be phased in incrementally. There is no need to tackle everything at once. Often, investing the effort to do the most complicated areas first will provide quicker results later.

Testing can validate individual components of ServiceNow (unit testing) or entire flows of applications (acceptance testing). Usually, a test strategy will incorporate multiple levels of testing to ensure components work individually and as a whole. This is very important, as many defects only occur when two or more components interact or when a specific sequence of events occurs.

Pro tip:

When looking for a test automation solution, the ideal platform should allow for multiple levels of testing across the entirety of ServiceNow.

Planning checklist

Identify a list of applications and workflows to test

- ☐ Prioritize: Consider what has the most impact and decide from high to low the importance of each feature/process
- ☐ Identify levels of testing and goals: Decide if you need to test individual pieces of code or configuration (unit testing), and what applications will require testing an entire workflow (acceptance testing)
- ☐ Assign roles and tasks to your team (process owners, project leads, developers, business analysts) – clearly identify who will be responsible for what
- ☐ Communicate schedule: Share the timeline with the team so everyone is aware of upcoming activities
- ☐ Communicate guidelines (level of detail, test frequency, success criteria) for the test strategy

Development

Automated tests can be developed at different times. This can be as early as when requirements are defined, after development, or at many times in between. Clear rules should be defined about when tests should be created and by whom, and these rules should be enforced as part of the development process.

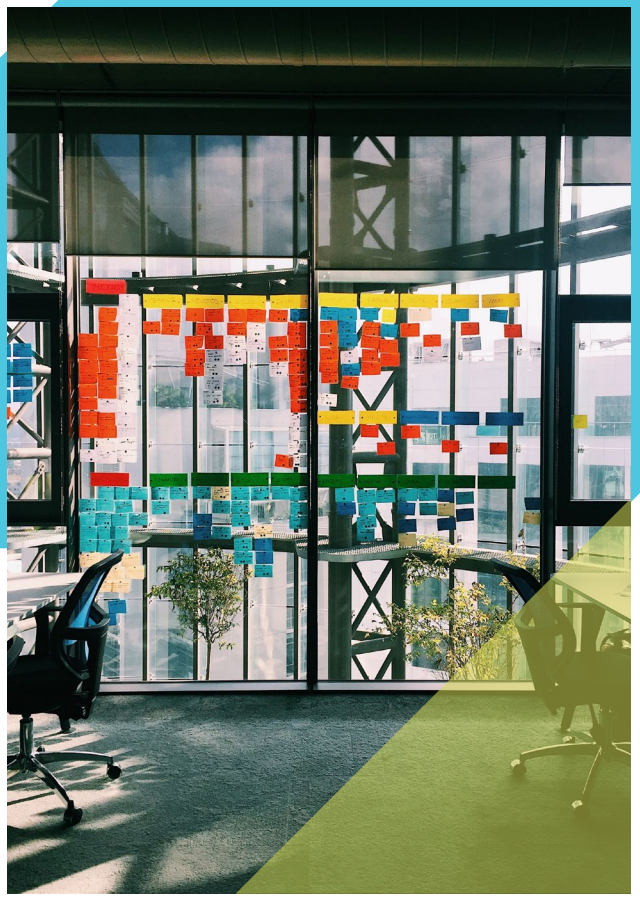
Having a well-organized set of tests will significantly decrease the time to develop while increasing the code coverage.

Pro tip:

Your testing platform should offer features that allow you to easily organize tests. It should also provide multiple paths to create the tests, allowing developers and non-developers alike to contribute to the process.

Development checklist

- ☐ Identify the test scenarios that need to be tested
- ☐ Fill in the details to include in your test plan and be as clear as possible. Include details such as prerequisites, test type, test data, execution settings, list of test steps, expected results, and test users (if necessary)
- ☐ Define test cases based on requirements and creating automated scripts – try dividing into two phases:
  Case definition: Defining test cases based on software requirements
  Automation: Creating automated scripts based on test cases
- ☐ Create tests by leveraging test data that are repeatable and importable



Execution checklist

Your test strategy should define how, when, and where tests are executed and how successful and failing tests affect the overall life cycle. This may include running tests at specific times of day, when events occur, and across several instances.

Having a clear and efficient execution strategy will help you find defects as early as possible, giving you the opportunity to fix them with the least impact.

- ☐ Determine if each test plan should be executed on a schedule or following some other trigger
- ☐ Prepare the test environment with appropriate data
- ☐ Confirm how you'll handle passes, failures, and defects

Pro tip:

Your testing platform should have the flexibility to execute tests on a schedule that works for your team and in a manner that does not disrupt individuals. When testing is complete, results should be well-organized, and users should be alerted to failures as soon as possible.

Maintenance checklist

Automation is not done when test cases are created. Automated tests will evolve with your system, and your test strategy should define how test upkeep occurs.

This allows you to maintain a high level of confidence in your tests and reduce the cost of upkeep.

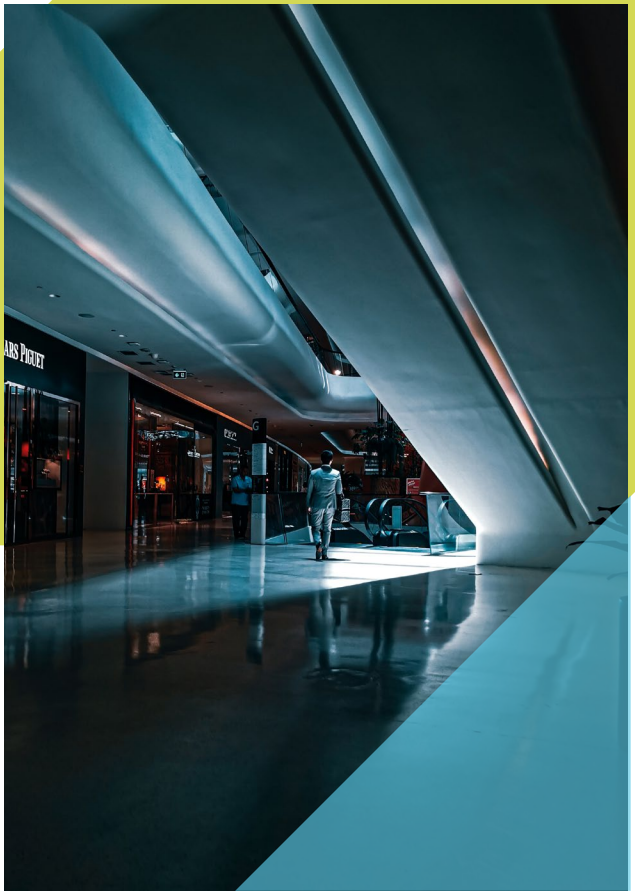
- ☐ Track and document your progress and defects
- ☐ Plan how you'll review configuration changes to keep tests up-to-date

Pro tip:

Any automated testing platform should offer a way to keep tests from becoming stale and possibly even link test cases to their source documents and configurations. An ideal system would even track those changes for you, giving you the ability to easily track and review tests associated with changes.

Upgrades

Now that we've walked through the strategy behind automation, let's discuss how to apply the foundation towards an upgrade. The above steps will allow you to create the baseline of your testing. When it comes to upgrades, the idea is to compare your baseline with the changes in the latest version. In addition to the checklist above, here are a few extra precautions to take when it comes to upgrades.



Upgrade checklist

- ☐ Planning: Allocate your resources and discuss the upgrade details (timeline, objectives, etc.)
- ☐ Planning: Review release notes for the target release and note the changes that may impact the upgrade
- ☐ Planning: Communicate the details with stakeholders and end users that an upgrade is happening
- ☐ Development: Ensure that all regression tests have been created prior to upgrade for baseline
- ☐ Execution: Run your baseline tests to ensure everything is passing before running against the target version
- ☐ Execution: Run your tests against the target version and review changes that need to be updated

Additional tips

- [ ] ServiceNow preview is available 4-6 weeks before general availability. Use this time to test in your dev instance to compare updated processes and features and note the difference
- [ ] Create process flow diagrams for applications if they do not exist and store these as controlled documents
- [ ] Create a story task that checks to validate a new feature does not impact pre-existing workflows for an application
- [ ] Test on a regular basis to track if tests are current (nightly/weekly)
- [ ] Keep your change process in mind and work around accordingly
- [ ] Keep communication open with your team to ensure the priorities are in order and that blockers are remediated as smoothly as possible

Final thoughts

Test automation removes the chance for human error and allows your tests to work 24/7, giving you better quality and increased speed to market with less resources. Having a test strategy will allow your organization to realize the benefits of planning and automating your ServiceNow platform with effectiveness and efficiency in mind while initially investing less.

2570 W El Camino Real, Suite #540
Mountain View, CA 94040
+1 650 383 8329
sales@tricentis.com