ServiceNow Best Practices

University of Rhode Island — Information Technology Services

Explore best practices, recommendations, and practical tips to help you get the support and information you need, work smarter, reduce challenges, and achieve better overall results. Click through below to view our recommendations.

For End Users

Submitting requests, tracking tickets, and getting help efficiently

1	**Email helpdesk@uri.edu — it's the fastest way to get help** Sending an email to helpdesk@uri.edu automatically opens a support ticket on your behalf. You'll receive a confirmation right away, and you can reply to that email at any time to add more information. No account login needed to get started.
2	**Write a clear, specific subject line** The subject line is the first thing the help desk sees. Something like "Cannot access Banner — login error since Tuesday morning" tells the team what's wrong, what system is affected, and roughly when it started. That context gets your ticket to the right person faster. Avoid vague subjects like "Issue" or "Help needed."
3	**Include the details that matter** In your message, describe what you were trying to do, what happened instead, and any error messages you saw. Let us know what device and operating system you're using, and whether anyone else you know is experiencing the same thing. The more context you provide upfront, the fewer back-and-forth questions are needed before we can help.
4	**Reply to your ticket email — don't start a new one** When you receive an update on your ticket, reply directly to that email rather than sending a new one. This keeps everything in one thread, ensures your response is linked to the right ticket, and prevents the help desk from accidentally creating a duplicate case for the same issue.
5	**Check the portal to see where things stand** You can log in to the ITS ServiceNow portal at any time to view the status of your open tickets, see the latest updates, and review past requests. You don't need to call or email to check in — the information is there when you need it.
6	**One issue, one ticket** If you're experiencing multiple unrelated problems, open a separate ticket for each one. Bundling several issues into one request makes it harder to route, track, and resolve each item. It also means one lingering issue could hold up the others from being marked complete.

For Fulfillers — Help Desk & IT Staff

Working tickets well, keeping records accurate, and supporting your team

1	**Link a CI to every ticket before you close it** Attaching a Configuration Item (CI) to every incident and change request is one of the most impactful habits a fulfiller can build. It tells ITS what was affected, where it lives, and who owns it — and it's how the CMDB stays accurate over time. If a CI doesn't exist yet for what you're working on, flag it so one can be created. No CI linked means no trail.
2	**Write work notes as you go — not just at the end** Work notes are the institutional memory of a ticket. Add a note each time you take a meaningful action: a call made, a test run, a fix attempted. If someone else needs to pick up the ticket, or a similar issue comes in six months from now, those notes are the whole record. A ticket with one closing note and nothing else tells almost no story.
3	**Assign to the group first, then to yourself** Always route a ticket to the correct assignment group before claiming it personally. Assigning only to your own name removes the ticket from the team's visibility — if you're out sick or change roles, the ticket can sit untouched without anyone knowing. Groups keep work visible and accountable.

4	**Keep the requester informed** If a ticket is going to take longer than expected, say so. A brief update in the portal — even just "still investigating, expecting to have an answer by end of week" — goes a long way. Silence is frustrating for end users, and most frustration comes from not knowing what's happening, not from the wait itself.
5	**Use "Pending" — don't abandon or prematurely close** If you're waiting on a response from the end user or a third party, set the ticket to Pending status and add a clear note explaining what you're waiting for. Closing a ticket before the issue is actually confirmed resolved means it often comes back — and the end user has to start from scratch.
6	**Check the Knowledge Base before you start troubleshooting** If an issue has come up before, there may already be a documented resolution. Search the Knowledge Base before spending time on diagnostics. And when you resolve something new that's likely to repeat, take five minutes to write it up. Every article you add saves your colleagues time the next time the same question walks through the door.

For Administrators

Managing the platform responsibly, safely, and sustainably

1	**Never make changes directly in Production** Every configuration change — no matter how small it seems — should be built and tested in a non-production environment (Dev or Sandbox) before it goes anywhere near Production. A single untested change can break functionality for hundreds of users. The rule is simple: if it hasn't been tested, it doesn't go to Production.
2	**Use Update Sets to move changes between environments** Don't manually recreate configuration changes across environments — you will miss something. Capture all changes in an Update Set in Dev, test it in QA, and then promote it to Production. Update Sets make changes trackable, auditable, and reversible. If something breaks, you can identify exactly what changed and roll it back.
3	**Document everything — in the platform and outside it** Add a clear description to every business rule, script include, flow, widget, and field you create or modify. Describe what it does, why it exists, and when it was last changed. Future administrators — and future you — should never have to reverse-engineer something you built. Undocumented customizations are technical debt that compounds over time.
4	**Resist the urge to over-customize** ServiceNow ships with a lot of well-designed out-of-the-box functionality. Every customization you add is something you own — through every upgrade, every patch, every platform change. Before building something custom, honestly ask whether the out-of-the-box option is good enough. The answer is often yes, and saying so is a form of discipline, not a compromise.
5	**Give people only the access they actually need** Assign roles and permissions based on what a user genuinely needs to do their job — not on what seems convenient or what someone might need someday. Over-permissioned accounts are a security risk and a data quality risk. Review role assignments regularly, especially when staff change positions or leave the university.
6	**Monitor CMDB Health — don't wait for problems to surface** Use the out-of-the-box CMDB Health Dashboard to track data quality on a regular cadence. Stale records, missing owners, and duplicate CIs don't announce themselves — they quietly accumulate until they cause real problems during an incident or audit. Catching issues early, when they're small, is far less painful than a large-scale cleanup.
7	**Test before and after every upgrade** Establish a set of critical test cases and run them before each environment upgrade. Run them again immediately after. The difference between those two results tells you exactly what the upgrade changed in your specific environment. Without a pre-upgrade baseline, you're flying blind on what broke and what didn't.