servicenow

Support DevOps in the enterprise with ServiceNow

What's in this Success Playbook

This Success Playbook will teach you how to get started with ServiceNow to support DevOps. It will help you answer the following questions:

- How do I learn what DevOps teams need so I can figure out how to use ServiceNow to support them?
- What use cases are most relevant to supporting DevOps at my organization?
- What steps do I need to take to implement solutions on ServiceNow that will deliver value to DevOps teams while supporting our operational goals?

Key takeaways

The most important things to know

Through customer interviews, we've identified the two most important steps to support DevOps with ServiceNow:

1. First, do your homework. Learn how DevOps is working at your organization—how many teams there are, how they work, what tools they use, etc.—so you know how to support your DevOps teams effectively.
2. Second, identify and implement an initial use case (typically change automation) that can deliver value to many DevOps teams in your organization by integrating ServiceNow with DevOps tools (such as CI/CD pipeline tools) and improving and/or modifying operations on ServiceNow as needed.

The payoff of getting this right

- You'll help your organization realize the expected value from DevOps by providing an effective system of record that is responsive enough to support the operational needs of continuous integration and continuous deployment.
- You'll help maximize the value provided by operations supported on ServiceNow by ensuring that they get operationally relevant information from DevOps tools.
- You'll prepare operations supported by ServiceNow to keep pace with DevOps demand, for example, increasing your rate of change.
- You'll ensure that IT operations supported by ServiceNow don't risk being relegated to supporting "old world" operations as your organization increasingly adopts a DevOps approach.

What you need to get started

- Support from your executive sponsor for investing in identifying and deploying functionality on ServiceNow to support DevOps teams
- Someone to champion the discovery, planning, and implementation steps needed to deliver solutions to DevOps teams
- Teams using DevOps and Agile methods for development because the use cases detailed in this playbook won't deliver expected value to teams using waterfall methods
- Continuous integration/continuous delivery (CI/CD) toolchains in use at your organization that ServiceNow can integrate into to support DevOps work

When you should start this activity

If your organization is shifting towards Agile and DevOps as your primary development method, you need to start this as soon as possible. Ideally, you will be working on this at the same time as DevOps practices are emerging at your organization, so you can co-create how ServiceNow fits into that new ecosystem.

servicenow

Playbook overview

ServiceNow recommends four steps to support DevOps* for the enterprise using ServiceNow.

Step		Outcome
Start At the beginning, you'll build your initial or foundational capability. This includes setting up initial frameworks, defining roles, and clarifying your objectives.	**Step 1 –** Get visibility into how DevOps works at your organization	You have an understanding of how DevOps works at your organization. You're ready to start working with DevOps teams and leaders to add value to their work using ServiceNow.
Improve As you improve, you'll take steps that help you reach your objectives and see value fast.	**Step 2 –** Identify where ServiceNow can deliver early value to DevOps	You've figured out where to start, with a use case that is widely demanded by DevOps teams and that you can implement relatively quickly.
	Step 3 – Design and pilot initial DevOps use case	You've demonstrated the value of your first use case with a pilot team(s), have optimized it to suit their needs, and are starting to expand access to additional teams.
Optimize Last, you'll refine and expand your capabilities so you can scale as you grow and continuously get more from using ServiceNow.	**Step 4 –** Roadmap your plan to expand support for DevOps with ServiceNow	Your initial use case is widely adopted and you're ready to explore opportunities to expand how you use the Now Platform to support DevOps teams.

These guidelines will help determine where you should start in this document:

- If you have little insight and/or understanding of how DevOps works at your organization and you aren't well networked with DevOps teams and leadership, start with Step 1.
- If you understand DevOps at your organization and what DevOps teams need but you aren't sure where exactly to begin, start with Step 2.
- If you know what you want to enable but you aren't sure how, start with Step 3.

- If you've already delivered value to DevOps teams at your organization with ServiceNow and are looking to expand that value by enabling new use cases, start with Step 4.

Note: In this playbook, we focus on how DevOps teams can use ServiceNow to help them when they're working on projects other than ServiceNow application development. We anticipate creating other resources to support using DevOps for ServiceNow application development in the future.

Terms and definitions

DevOps – DevOps is the combination of cultural philosophies, practices, and tools that increases an organization's ability to deliver applications and services at high velocity: evolving and improving products at a faster pace than organizations using traditional software development and infrastructure management processes.

DevOps Change Velocity – DevOps change velocity is the capability within ServiceNow that can be used to automate change management leveraging integrations with your organization's DevOps toolchain.

DevSecOps – DevSecOps (development, security, and operations) automates the integration of security at every phase of the software development lifecycle, from initial design through integration, testing, deployment, and software delivery. Embracing this shift-left mentality requires organizations to bridge the gap that usually exists between development and security teams to the point where many of the security processes are automated and handled by the development team itself.

ServiceNow DevOps – This ServiceNow product is designed to expand DevOps success across the enterprise and reduce the friction between IT operations and development, primarily by enabling change management to operate at DevOps speed.

Use case – Use cases are specific situations or needs that a product, functionality, or solution can be used for to deliver value to your target customer.

CI/CD pipeline – A CI/CD (continuous integration/continuous delivery) pipeline is the set of tools used to automated steps in your software delivery process. It is "the backbone of the modern DevOps environment. It bridges the gap between development and operations teams by automating the building, testing, and deploying of applications."[1] See Appendix B for visual of a continuous integration/continuous delivery (CI/CD) pipeline.

[1] DevOps Zone, August 2018, **Learn How to Set Up a CI/CD Pipeline From Scratch**, https://dzone.com/articles/learn-how-to-setup-a-cicd-pipeline-from-scratch.

servicenow

Step 1 – Get visibility into how DevOps works at your organization

KEY INSIGHT

- Assign a champion to build a structured discovery process.

ServiceNow is well positioned to deliver high-quality operations to DevOps teams so they can work quickly with minimal downtime and without incurring risk. However, it's often not obvious how to best do this because the ServiceNow platform support team isn't typically directly involved in DevOps. You may lack a single point of reference or definition of how DevOps works and what DevOps teams need from them. This challenge is compounded when DevOps is approached differently across different DevOps teams in the same organization. In fact, it's common to have each DevOps team in your organization executing differently, often using different tools.

To overcome this, it's critical to start by gaining visibility into DevOps *as practiced* to identify how best to use ServiceNow to support DevOps teams.

Start by researching how different DevOps teams at your organization work, what they need, and what tools they use. This insight will prove critically valuable as you start to figure out specifically what you need to do, and when, to start using ServiceNow to help deliver a high-performing, enterprise approach to DevOps at your organization.

Note: Having an enterprise approach to DevOps refers to DevOps teams across your organization adhering to a consistent set of practices and tools. This centralized approach makes it easier to support and optimize how work gets done.

To get visibility into how DevOps works at your organization, complete the following action steps. Keep in mind these insights from our customers.

Action step 1: Assign a champion to lead discovery and objective setting

Before jumping to technical solutioning, assign someone to champion the effort who is required to identify and design the right opportunities to support DevOps at your organization.

We've found that the ServiceNow platform owner is generally best positioned to assume this responsibility at most organizations—most organizations already have a designated platform owner, that person typically understands how their ServiceNow implementation delivers value (and where it could be improved), and they have relationships with the right people to get things started.

If the platform owner is unable or unwilling to champion the work required to start supporting DevOps with ServiceNow, select someone else with similar seniority and a solid understanding of your current ServiceNow implementation (for example, your senior ServiceNow developer). What's most important is that *someone* assume responsibility for coordinating the people and process described in this playbook to identify opportunities and implement a solution.

The champion must:

- Familiarize themselves with the guidance in this playbook
- Discover how DevOps teams work at their organization specifically and where ServiceNow can be best positioned to support them
- Identify desired outcomes for any use case they think should be implemented
- Engage and coordinate relevant stakeholders to design, implement, and pilot initial use cases to deliver value to DevOps

INSIGHT 1:

Assign a champion with insight into what DevOps teams need, when possible

Some organizations start with a strong sense of what needs to be done to support and improve DevOps at their organization—either because demand from DevOps teams and leadership is obvious and/or because the ServiceNow platform support team members already have a keen sense of where they can add value with ServiceNow. This can (and should) influence who you select to champion your ServiceNow DevOps implementations. For example, if the need to speed up change management approvals is obvious, your process owner for IT change management or someone from your change advisory board (CAB) can also be an excellent candidate for the champion/coordinator role. In this case, the IT change management process owner will already have some insight into the stakeholders involved, what they need, and what can be delivered. Just make sure your champion is culturally adaptable to DevOps—in other words, they should not be so rigid with the process that they're disinterested in supporting developers' needs.

Action step 2: Build a structured discovery process

With discovery, you gain a better understanding of how many DevOps teams your organization has, how they work, what tools they use, what they need, and what their top pain points are.

This discovery may be the most important step to successfully identifying where and how ServiceNow fits into supporting and improving the work DevOps teams do. Effective discovery:

- Connects you with the people you need to work with
- Informs what you need to implement to resolve pain points and deliver value to DevOps teams

- Reveals what DevOps teams want, which should guide where you invest first and how to design a solution they *willingly* adopt (even if adoption requires them to change how they work and/or what tools they use)

Doing this well involves more than just asking DevOps leaders and team members what they want. What they want may tell you enough to get started, but you need a lot more insight into how DevOps works at your organization to inform a roadmap for how you will deliver value to the largest number of DevOps teams possible.

Overall, the information you collect in this step will help you identify:

- How common certain team structures and processes are, which helps you prioritize demand for functionality on ServiceNow (what to implement first)
- How often teams are using similar tools, which helps you know if the integrations you design will benefit enough teams to be worth your investment
- Where there are the gaps in those tools and/or functionality that teams use, which could identify a solution you could deliver to resolve the gaps
- Which use cases are most needed and/or which might be out of scope based on how DevOps teams are working (and how often they work the same way across teams)
- Where organizational change management (OCM) is going to be hardest to manage (for example, because certain DevOps teams are more established, larger, or set in their ways)
- Where you may struggle with adoption because it may require significant changes to how DevOps teams work

See Table 1 for a list of the different types of information we recommend that you collect for this step and why.

Collect information on:	This information helps you:
What each DevOps team looks like (team size, composition, etc.)	Understand the teams that you want to support, how mature they are in how they do work (with regard to Agile/DevOps) and how different they are from one another. Also identify if teams are structured to truly manage development and operations. For example, are the DevOps teams really comprised of dev and ops or are they more like dev teams that are separate from ops?
The applications the team supports	Identify how critical their work is to support your organization. For example, what happens if their applications experience downtime?
The tools the team uses (and how)	Identify which tools are most commonly used across DevOps teams in your organization and how. Are the same tools being used in different ways? Or are teams using different functionalities within the same tools?
The KPIs the team tracks	Understand what's most important to the DevOps teams you want to support with ServiceNow functionality.
How the team currently uses IT operations and participates in post-deployment production issues	Identify where these teams already interact with your ServiceNow implementation and if they're happy or unhappy with it. Also, find out if they're using other tools to do things that you can support with ServiceNow, like submitting tickets.
The teams' top pain points	Identify where these teams currently struggle, especially if that difficulty specifically involves ServiceNow. Ask about the team's regulatory requirements to identify when/how pain points are linked to audits.
Whether the team would prefer investment in: • Automating change management • Automating audit reporting • Reporting and analytics providing business insights across your DevOps lifecycle	Identify a solid lead for the first use case you should implement. These three options are the most common use cases we've seen and are most likely to deliver early value to your organization (see Step 2).

Table 1: Information to collect to create a structured discovery process

"DevOps teams across our organization use a lot of separate processes and tools that my ServiceNow team doesn't even know about…so we can't have a complete vision of what we need to do with ServiceNow to support DevOps."

– ServiceNow platform owner at a Fortune 500 insurance company

Collecting this information is an investment, so it's important to collect everything you'll need in your first attempt, when possible. We recommend this four-step process:

1. **Assemble a list of DevOps teams in your organization.**

   If you have a designated DevOps leader, start by asking that person for a list of DevOps teams you can survey. Otherwise, we recommend a grassroots approach to assembling a list of teams. To do this, start by reaching out to the lead of one DevOps team you know, and ask that person to connect you with any others they work with. Branch out from there until you're confident that you've identified most of these teams.

2. **Build a consistent approach to learn what you need to know about each team.**

   In many cases, there will be too many groups to informally approach and learn from. A more systematic approach using a common set of questions will provide better insight. Your assigned champion (possibly your platform owner) should take the lead on developing a consistent set of questions to learn what you need to from each DevOps team. See Table 2 for a set of questions you can modify and use. Note the different survey sections—you can use different questions as needed but make sure that you include questions that will collect the right *types* of information, such as information on teams, the apps they support, their top pain points, etc.

   The right questionnaire will keep you from skipping directly to investigating the tools without considering the people, process, and data involved in the DevOps approaches at your organization. Without this full picture, it's easy to overlook how teams might be using the same tools differently, which could factor significantly into how you design integration and other functionality.

3. **Collect information from each team.**

   Send each team lead the questions as a survey or meet with them in person (in a business alignment session, interview, process scoping workshop, etc.) or both. Aim for at least 50% participation (ideally 90%+).

4. **Analyze the results that you collect.**

   Look for patterns in the information you collect. At best, you'll see a clear, dominant pattern that will help you identify what the average DevOps teams looks like at your organization. You'll be able to use this insight to learn precisely how that average team works, what's important to it, and where they struggle—all of which will inform how you use ServiceNow to help.

Questions	Additional considerations and specific questions
1. What does your team look like? How does it work?	• How many people are on the team? • What part of the organization does your team reside in? • Is your team using Agile methodology? Scrum? For how long? • Do you have all skillsets needed to manage both development and operations in your team? • What does your code release process look like?
2. What apps do you support?	• Who are your customers (that is, your internal or external users)? • What type of solutions are they developing (revenue generating, internal for the enterprise, etc.)
3. What tools do you use?	Specify whether you use a tool in a specific configuration, for example, Jenkins parrallel pipeline. (See Appendix C for a more detailed list of DevOps tools you can include in your questionnaire.) • Planning: (Azure DevOps Boards, Jira, CA Rally, NOW ITBM, other?) • Coding: (Bitbucket, GitHub, GitLab, other?) • Testing: (JUnit, Selenium, NUnit, other?) • Orchestration: (Jenkins, Azure DevOps, GitLab, other?) • Security/Software Quality Scanning: (SonarCube, Veracode, Fortify, CheckMark, Black Duck, other?) • Deployment: (Urbancode, Ansible, Kubernetes, other?)
4. What KPIs or performance metrics do you track?	
5. What is your team's frequency and volume of code releases?	
6. How do you currently submit tickets/requests?	• What do you commonly request? • What tool do you use?
7. What are your top pain points?	• How much time and manual effort is spent on creating change requests? • How much time and effort do you spend getting approvals for change requests? • Do you currently have a large number of fragmented DevOps tools with no way of collecting data and metrics across the entire life cycle?
8. Are you under regulatory compliance?	• How often is code release process audited? • How much time and effort do you spend collecting data to pass internal or external audits?
9. Is implementing DevSecOps a priority?	• Do you have an ongoing DevSecOps initiative? If not, is one planned in the near future? • How important is the use of security scanning data in your risk calculations as part of approving a production deployment?
10. Which of these three use cases would be most valuable to you?	• Automated change management/change approval • Automated audit reporting • Reporting and analytics providing business insights across your DevOps lifecycle

Table 2: DevOps discovery questionnaire

INSIGHT 2:

Identify what the average DevOps team looks like first

The first thing you should do after collecting this information is to identify a dominant pattern that will represent what the average DevOps team at your organization looks like.

First, identify the most common responses to each of the questions in your questionnaire. Then answer these questions to further analyze your data:

1. Is there significant diversity in the responses you collected? If so, why? For example, do DevOps teams from different parts of the organization (question 1) respond differently? Do teams that use different tools (question 3) respond significantly different needs? This will help you identify if there are distinct cohorts, which you need to consider when you define what "average" looks like overall.
2. Does the requested use case (question 9) match the most commonly cited top pain point (question 5)? If they don't match, is there a clear demand for another use case? This will help you verify if the average team would benefit from one of the three use cases we recommend starting with (Step 2).
3. Are the most commonly used tools far more prevalent than others (question 3)? If not, how common are these other tools across teams? For example, do 51% of teams use one build tool and 49% another? This will keep you from over simplifying what tools the average DevOps team uses and will help inform what integrations you need to enable a use case later on.

What to do next

Now that you've learned about how DevOps is working at your organization, you're ready to figure out where to start by supporting common DevOps teams' needs using ServiceNow.

servicenow

Step 2 – Identify where and how ServiceNow can deliver early value to DevOps

KEY INSIGHTS

- Most organizations should start with one of three use cases: automating change management, automating audit reporting, or providing insights across your DevOps lifecycle on ServiceNow.
- Information from your discovery process (Step 1) is key to selecting the right initial use case and finding the right team to pilot with.

Your findings from Step 1 should have given you a strong sense of what groups are involved in DevOps, how they work, and what they need. This insight is critical to identifying opportunities for using ServiceNow to improve how DevOps is done at your organization. More importantly for now, it tells you where you need to start to alleviate the top pain points and show value quickly to build support for ServiceNow's role in supporting DevOps at your organization.

To identify where and how ServiceNow can deliver early value to DevOps teams, complete the following action steps.

Action step 1: Decide on where to start

In collaboration with customers who have already started using ServiceNow to support DevOps, we've identified the top 10 use cases that we think have the most potential over the next few years to deliver big value to DevOps teams at customer organizations. (See Table 3 and, for a more detailed version of Table 3, Appendix A.)

Review each use case in Table 3 and consider which of them might be most valuable to DevOps teams in your organization based on what you learned in Step 1. Then pick one to start with. Whenever possible, we recommend that you start with one of three use cases that we've found are needed most at customer organizations:

- Automated change management
- Automated audit reporting
- Reporting and analytics to provide business insights across your DevOps lifecycle

We recommend these three because we've found that they enable the functionality that's most commonly requested by DevOps teams—especially automated change management. These three use cases are also more likely to deliver obvious, trackable value to DevOps teams

Initial ServiceNow DevOps use cases	Description	Availability in ServiceNow DevOps
Automated DevOps change management	Automate change management process by connecting to the existing CI/CD tool set to collect the data required to create change tickets, calculate risk, and automate approvals. This cuts the time needed for DevOps teams to obtain approval for changes from days to minutes and maintains governance by avoiding the need for developers to leave their dev tools to log change tickets.	Available in the current product release
Automated audit reporting	Maintain an end-to-end audit trail that correlates the pipeline, releases, and incidents to eliminate manual audit data collection time. Effectively reduce the time needed for "reactive" audit data gathering from weeks to minutes by integrating with the tools that generate the data needed by audits, aggregating it in ServiceNow, and enabling automated report generation. Automatically managed data collection means that audit data is consistent and less prone to incorrect entry.	Available in the current product release
Reporting and analytics to provide business insights across your DevOps lifecycle	Integrate across tools in the CI/CD pipeline to gain visibility and tracking of end-to-end CI/CD pipeline to dashboard performance and identify opportunities for improvement. Eliminate manual tracking mechanisms like Excel spreadsheets from the process.	Available in the current product release
Pipeline orchestration	Create a path to production and gate promotion policies.	Available in a future release
Value stream mapping	Visualize the end-to-end delivery process and optimize the throughput of the value stream.	Available in a future release
Release readiness and orchestration	Orchestrate multi-app releases across teams and manage dependencies.	Available in a future release
Configuration Data Management (DevOps Config)	Manage configuration data through the CI/CD lifecycle	Available in a future release
Incident/problem management integration	Integrating problem management with development planning tools to introduce development work that needs to be done to resolve recurring incidents directly into the development lifecycle. Operationally, incident and problem management become easier to manage because managers have more information (pulled from development tools via integration) in tickets.	Requires services and custom work
Predictive operations and code reverts	Tie DevOps changes to related infrastructure (by tying all information in the change record to relevant CIs) so that if there is a problem with the infrastructure (e.g., if there's an incident) you have more information to identify the issue than in the past, help with root cause analysis, and revert faster.	Requires services and custom work
Developer onboarding	Reduce the time to productivity for new developers by enabling them to onboard quickly and become productive on day one and by automating access control to development environments and IDE/PaaS access. This helps attract and retain developer talent.	Requires services and custom work

DevOps self-service portal	Empower developers to self-serve by providing a single system of engagement for development teams to collaborate and communicate requests and delivery through workflows for shared service, and a curated catalog of developer tools and libraries. This reduces the "swivel chair" effort for development teams to work across different tools. This also helps ServiceNow teams get greater visibility into DevOps processes.	Requires services and custom work

Table 3: Description, benefits, and level of effort required to implement different use cases to support DevOps with ServiceNow

INSIGHT:

Use the information you collected in Step 1 to decide where to start

We specifically designed the DevOps discovery questionnaire (Table 2) to identify which of the top three recommended initial use cases was requested by the most survey participants (refer to Table 2, question 9). Start by delivering the most demanded of those three use cases if you agree that the average DevOps team at your organization will benefit from that use case, based on your analysis of questionnaire responses (Step 1).

If the results are close (for example, 42% asked for audit and 45% asked for automated change) seek more granular information by reviewing your findings. First, sort your data into three groups, divided by which initial use case they requested (See Table 2, question 9). As you review, consider the questions in Figure 1.

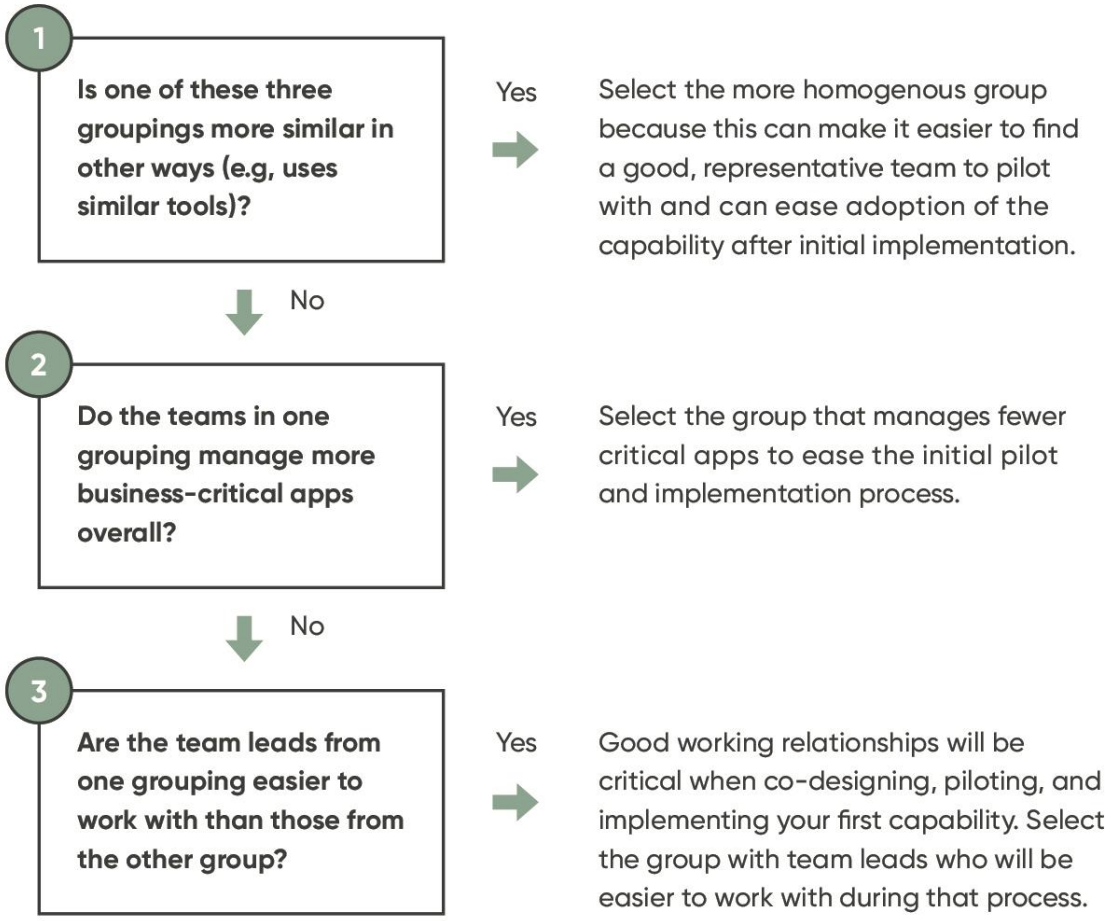


Figure 1: Decision tree to decide between use cases if there are multiple groupings that your discovery process identified would deliver similar value

EXPERT TIP

In some cases it's possible (and efficient) to implement more than one use case at a time, especially when the same integration can enable multiple use cases. For example, the very integration needed to automate change management (such as integrating Jenkins with ServiceNow Change Management) can also be used to pull the data needed to automate audit reporting and/or provide DevOps insights. However, we still recommend leading with one use case to make sure you deliver value as early as possible before following through with delivering other use cases (even if they all use the same primary integration).

Action step 2: Connect with DevOps team leads to vet where you should start

After selecting an initial use case, connect with the DevOps team leads who originally asked for that use case in your questionnaire (Step 1). If you were correct on what use case to start with, this step gives you an opportunity to learn more about their teams' specific needs, how a potential solution could fit into how they work, and to connect with teams to co-design and pilot an initial implementation. If you misread the information you collected (or if its needs have changed since), this vetting step gives you a critical chance to change course before investing in a specific solution.

You can convene team leads in a meeting to review a proposal all at once, but we recommend going back and talking to each individually first. This way, you'll hear from each team and you have a chance to build individual relationships that will make it much easier to implement and expand your solution's use later.

Accomplish the following as you meet with these leaders:

- **Build rapport and show that you've done your homework –** Start by reporting back your findings from the questionnaire. Describe what you learned about the diversity of DevOps teams/approaches in your organization and what teams described as pain points.
- **Vet and earn support for your direction –** Explain your decision to start with the use case you've selected (automating and integrating change management, for example). Focus on how questionnaire findings informed this (see below for guidance on how to use numbers to support this). Reassure them that you don't intend to replace the tools they use with a ServiceNow solution, only to integrate and collect data from them to provide new support. Ask if your decision still makes sense to them or if anything has changed since they filled out your questionnaire.
- **Collect more information to guide your next steps –** Ask questions to learn more about why they answered the questionnaire like they did. Focus on learning more about why they prioritized the use case you want to enable first.
- **Seek partnership –** Ask if the team would be interested in collaborating with you to design, pilot and implement the integrations needed to enable your initial use case. Mention that this would get them first access to the resulting functionality and benefits (such as eliminating their need to manually enter change requests).

Productive and positive engagements with these team leads will not only help you inform your direction, but it will make it more likely that their teams will readily adopt the use case you deliver. Involving these teams early on will also help ease some of the typical tensions over organizational change that can derail other initiatives (such as a fear of change or of feeling uninvolved that can lead to them resisting the adoption of new capabilities).

INSIGHT 1:

Use hard figures to build credibility when meeting with DevOps team leads

When meeting with teams, support your explanation for where you want to start with hard figures from the information collected in your questionnaire. For example, "68% of the DevOps teams we surveyed are frustrated by how slow change management is currently, so we are considering starting by automating change management on ServiceNow and integrating it with CI/CD tooling to dramatically decrease the time to approve change requests without effort from developers."

Best practice

Familiarize yourself with the target KPIs for the DevOps team leads you meet with. You should have collected them with the questionnaire. Refer to them during your meeting, when appropriate, to better explain how your initial solution with ServiceNow can directly improve their KPI performance.

INSIGHT 2:

Connect with other stakeholders beyond just DevOps team leads

DevOps team leads are the obvious stakeholders to connect with because ultimately you want to deliver support to them and their teams. However, there are other stakeholders that are still important to learn from. You may also need to gain their support for your work. Don't forget to also engage:

- DevOps support teams (such as any centralized teams that manage the tooling used by DevOps teams)
- DevOps leader(s) (if this role exists in your organization)
- Your ServiceNow executive sponsor to make sure that other leaders are involved as appropriate

Of these stakeholders, the DevOps leader is the most important to engage and get support from—if they disagree with your current direction, ask them to follow up with the teams you've spoken with and work with you to set a direction that all can agree on.

Action step 3: Select a team to partner on design and to pilot your initial use case for a quick win

It's essential to learn from as many teams as practical but you need to pick only a small group to pilot with. Depending on the structure and/or scale of your organization, this group could be as small as a single DevOps team or as large as all DevOps teams in a line of business. Keep the

group as small as reasonable so you can move though design, pilot, and initial implementation quickly. Ideally you can pick a single team led by one of the team leads you met with in the last action step. After you pick a group to pilot with, work with that group to select a pilot app. Ideally this app should support a business need so you can clearly track the value of your solution to the organization, but it's not so important that your pilot will risk significant business value.

INSIGHT 1:

Pilot with teams that are enthusiastic about your solution

Select a pilot group that's led by someone who seemed especially keen on the initial use case you intend to deliver. It also helps when that lead has demonstrated strong support for ServiceNow in the past. Some customers have even found that DevOps teams led by former IT operations professionals make excellent candidates because they're more likely to understand and value what you're trying to deliver with ServiceNow.

INSIGHT 2:

Set realistic expectations with you partner team upfront

Set realistic expectations with the partner team(s) upfront about what the pilot process will look like and what support you'll need from them to implement your solution and get to value. If multiple teams are involved, make sure that people from all teams meet and understand the need to work across teams to meet timelines and goals. Involve a project manager to help coordinate across multiple teams and timelines if needed.

What to do next

With a decision on initial your use case, you're ready to pilot the solution with your selected pilot group.

Step 3 – Plan and pilot your initial DevOps use case, then expand adoption

KEY INSIGHTS

- Consider what data you need to share and how existing processes need to change before jumping to technical implementation.
- Plan how you'll onboard new DevOps teams into your initial use case after the pilot.

Now that you know where to start and who to work with, build a plan to enable your initial use case with your pilot team(s). When it's implemented and delivering value, then you can start to think about how to onboard additional DevOps teams into the solution.

To do this, complete these action steps. Keep in mind our customers' insights—they performed these actions.

Action step 1: Enact your initial use case with your pilot group

Before you can implement and pilot, you must think through what you need from a data and process standpoint for your solution to work. From a technical standpoint, you'll likely need to integrate your initial use case with ServiceNow using the appropriate development tool (such as tool(s) in the CI/CD pipeline). For integration to add value, though, you first need to figure out what information needs to be shared via integration and how to use that information once it's in ServiceNow to deliver value.

If you're starting with one of the three use cases that we recommend (like automated change, automated audit reporting, or developed pipeline dashboarding), ServiceNow has out-of-the-box (OOTB) integrations (available in the ServiceNow DevOps product) that you can use. If you use these integrations, you don't need to design and customize your own— you *only* need to figure out how to use the prepared integration to share the data you need to automate and improve processes.

To do this well, we recommend using the "recipes" in Figure 2 as your guide as you move through the high-level steps of figuring this out.

Steps to start automating and integrating change management approvals

1. **Review the current change process.**

   Review your current change record creation and approval process, including data inputs and decision logic. Review existing inputs and logic for potential improvements and update as needed to meet business requirements.

2. **Automate risk assessment.**

   Improve the efficiency of your change management processes by expediting change risk categorization. By doing this, you can expedite more standard, lower-risk changes when you automate change approvals. Complete the following prerequisite steps to prepare to automate risk assessment:

   a. Define the criteria you want to use for risk assessment going forward. We recommend:
      - **Functionality –** Does the code you're releasing perform the desired function. For example, are unit and regression tests passing at an acceptable level, etc.?
      - **Compliance –** Is the code in compliance with corporate security guidelines. For example, are security scans passing with no high-risk issues, etc.?
      - **Business impact –** What services are impacted? Are there any conflicts with blackout periods, maintenance windows, etc.?

   b. Identify the following for each criteria:
      - What information you need to capture from users submitting change requests
      - What thresholds are required to pass defined criteria
      - How you will weigh different criteria

   c. Review all risk assessment factors (defined above) with the change management process owner and change advisory board (CAB) for approval before you implement them.

   d. Use Change Management – Risk assessment and/or the Activate Best Practice – Change Risk Calculator plugin to automatically calculate the risk of changes using your defined risk criteria.

   Related resource: Automating incident and change management (playbook, page 12)

3. **Automate change record creation.**

   Identify the tools that provide the data needed to assess criteria defined in Step 2. Implement integrations with those tools to collect data for automated change record creation using the ServiceNow Change Management API.

   The Change Management API provides REST APIs that enable third-party, CI/CD pipeline application integration with the ServiceNow® Change Management process. By integrating your applications with the ServiceNow Change Management process, all change requests, regardless of where they're initiated, have a single source of truth and provide a single audit source.

4. **Activate the process.**

   Implement a "gating" two-way integration with a building or orchestration tool (Jenkins, etc.) to stop all code deployments and wait for change record approval in ServiceNow. If a change record is low risk, it will be automatically approved, there will be no wait time, and the code will be deployed instantly. If

a change is high risk, the code deploy will be delayed until the change record is manually approved or stopped completely if a change record is rejected.

Steps to enable automated audit reporting

1. Identify what information needs to be collected for audits most frequently or what regulatory policies you need to comply with and map to your collected data (e.g., what data does your auditor care about?).
2. Identify where that information is currently created and stored (which tools, etc.).
3. Implement integrations with those tools to pull and aggregate information in ServiceNow to generate audit reports.

Steps to start delivering reporting and analytics to provide business insights across your DevOps lifecycle

1. Identify the most standard pipeline your organization uses—in other words, the tools used for planning through operation.
2. Identify what the teams using these standard tools want to track, such as the KPIs they're interested in tracking across their pipeline.
3. Identify what specific information needs to be pulled from each tool to enable dashboarding the metrics these teams want to track.
4. Implement integrations to pull that information into ServiceNow.
5. Build custom dashboards to present the aggregated data back to DevOps teams.

EXPERT TIP

The ServiceNow DevOps product is currently designed to support Jenkins, Azure DevOps, and GitLab, so we recommend using one of them as the build/automation platform for your CI/CD pipeline if you have not already chosen a different tool to function in that capacity.

If you chose to start with another use case, consider these questions:

- Related to the use case we want to enable, where specifically does our current process fail or cause delays to your DevOps team's work?
- What information would I need to share between ServiceNow and other tools in order to resolve these failures/delays and/or add value to how DevOps teams are doing work?
- If I had all the right information available in ServiceNow, how would I use it to improve the value that I can deliver to DevOps teams?

- How would existing processes on ServiceNow need to change for us to make effective use of the information pulled from other tools?

Answering these questions should give you a sense of what you need to integrate with and how you can use the information you share via integration to enable your desired use case.

> "Before designing how to technically make this work, we needed to come up with an idea of all the types of information we need from various development tools to be able to support DevOps with operations managed in ServiceNow."
> **– IT operations leader at a Fortune 500 entertainment company**

Given that DevOps is still an emerging practice, we recommend working with a certified ServiceNow partner that has prior DevOps market experiences to implement your use case—especially if you don't start with one of the three initial use cases we recommend. There are a number of partners with experience in using ServiceNow for DevOps. Please ask your account team if you would like help connecting with one.

INSIGHT:

Design a solution that can ultimately benefit all DevOps teams

Don't lose sight of your end goal: to expand access to this initial use case beyond your pilot group's use. Be careful to avoid designing a solution that's overly specific to your pilot team's needs. Use what you learned during discovery to ensure you design an enterprise approach—one that can ultimately benefit and unify all DevOps teams via one solution—by starting with a solution that benefits a majority of teams from the start rather than catering to the niche needs of a specific team.

Best practice

Before piloting, vet your design with other DevOps team leads outside of your pilot team. Deciding to work primarily with a smaller group to design and pilot your initial DevOps use case helps speed up the process but you can't lose sight of the fact that ultimately you want all other DevOps teams to leverage your solution as well. To ensure your solution will meet the requirements of a larger audience, take the time to continually check your design and intended direction with the other DevOps team leads you've met with.

Action step 2: Pilot your use case, implement all required integrations, and make minor optimizations

With a sense of how your use case will work, it's time to pilot. Implement the integrations and any required process changes and start testing how it works with your pilot team. Make modifications as needed. Follow any piloting protocol established at your organization.

Look for the following three things when piloting to inform how you expand:

- How long did the implementation take?
- How much did the pilot DevOps team need to change about how they used to work in order to use the new solution (the tools, process, etc.)?
- How did the implementation deliver value? Make note of KPI performance before implementation to baseline performance for comparison post implementation.

After a successful pilot, capture the lessons learned and data about how the solution added value to the pilot team's work. Communicate this broadly with DevOps teams across your organization to demonstrate the benefits of your use and spur interest in adopting it.

INSIGHT 1:

Keep your pilot solution as simple as is reasonable

Limit (or at least be very careful with) how much you modify and optimize your pilot solution. Remember that the solution was designed with the idea of delivering an enterprise approach that other teams could quickly adopt. Over customizing to cater to pilot group needs might hinder your adoption goals later on.

INSIGHT 2:

Work with your pilot team to champion your solution

Enlist your initial pilot team lead as a champion who can help communicate the benefits of your use case to other leads, convince them to implement it, and onboard other teams. Support from this trusted DevOps insider will drive adoption and help you identify and navigate any negative feedback or resistance to your use case.

Action step 3: Develop onboarding plans and start expanding access to other DevOps teams

When your initial use case is implemented and adding value to your pilot team's work, turn your attention to how you plan to expand adoption to other teams and apps with higher business value. You already have a list of teams that should be ready and excited to take advantage of the new functionality you can offer (for example, the teams you vetted the proposed solution with), but you shouldn't just open up access to your solution without restriction. Instead, you need to make a plan for how to effectively onboard new teams and apps, including requirements for how they'll need to:

- Learn to use the integrations and process
- Change how they work to conform to the standard approach designed, if needed (e.g. if the integration is designed for Jenkins, teams will need to start using Jenkins to make use of the new use case)

EXPERT TIP

Implementing use cases using the ServiceNow DevOps product itself should not require developers to change the tools they use—most often, it integrates and works in the background to support developer work. From a broader point of view, however, companies trying to scale an enterprise approach to DevOps may want developers to standardize the processes and tools they use. In that case, use cases implemented on ServiceNow can help guide behavior toward standardization by rewarding adoption with privileged access to new functionality. For example, teams using Jenkins, Azure DevOps, or GitLab can use automated change management but those using CircleCI must continue with the manual change approvals.

We recommend you build a consistent onboarding plan with set criteria to assess when teams are ready to adopt the new use case. For example, one of our customers used the process in Figure 2 to onboard DevOps teams to take advantage of the automated change management process they had enabled.

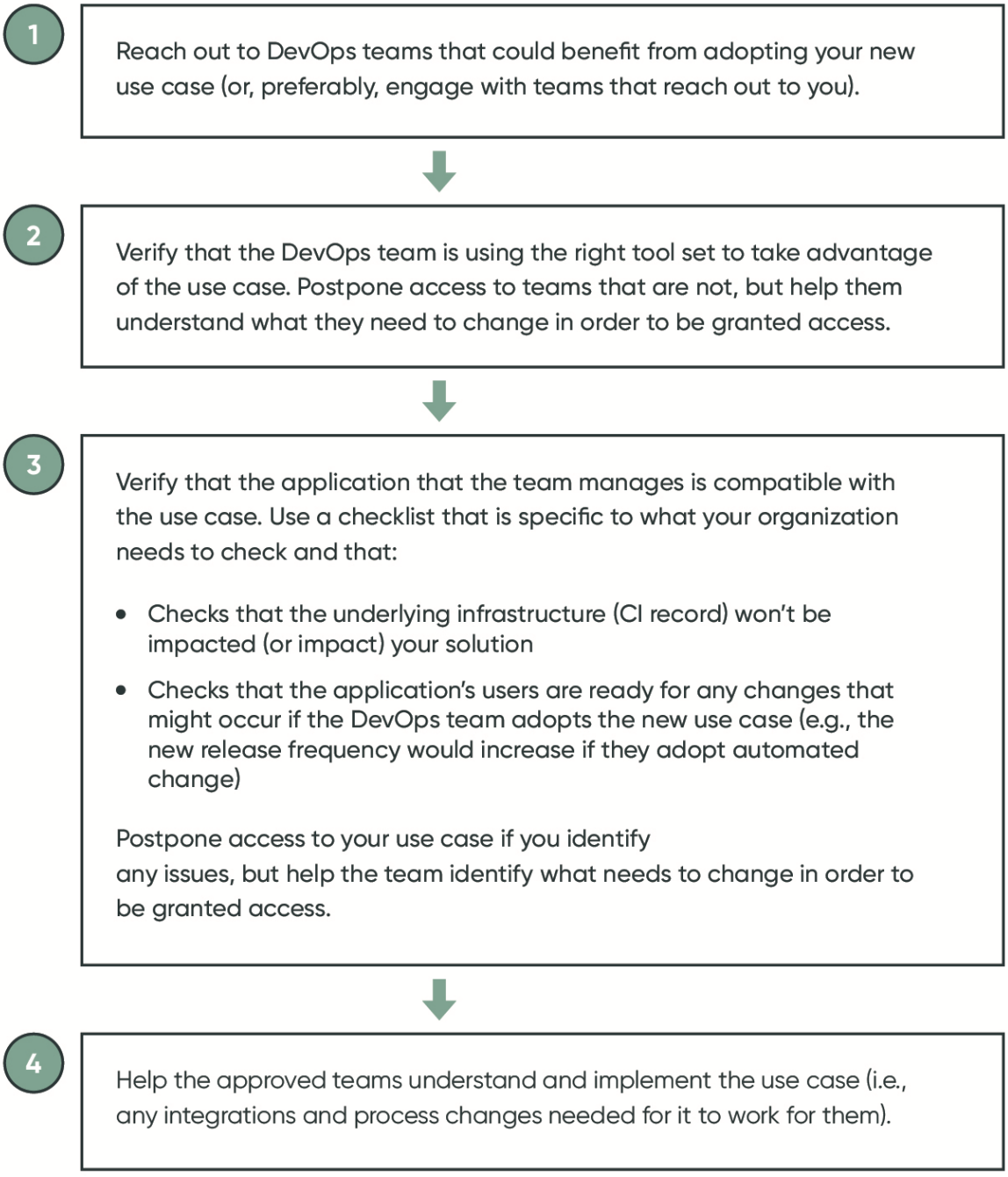


Figure 2: Process used by one leading organization to onboard DevOps teams to the automated change management use case they implemented

"We started by designing an ideal solution with DevOps teams from just one line of business—that solution is working so well that other groups are coming to us asking to use it.… We have a standard onboarding process where we ensure each team is ready so that we don't have to reengineer something every time a new team adopts. Since we have the right solution, it hasn't taken much convincing to get them to comply. Over half of our lines of business had adopted the solution within the year."

– Platform owner at a Fortune 500 finance company

INSIGHT 3:

Onboard quickly, starting with most interested teams

Start by expanding access to teams that have come to you and expressed an interest and willingness to make modifications to how they work so they can take advantage of new functionality. Your adoption rate is key to proving early value, so allow access to as many teams as you can support and that comply with the approval criteria you establish as part of your onboarding process.

What to do next

Once you've gained momentum with your initial use case, investigate and plan to implement other opportunities to deliver value to DevOps teams using ServiceNow.

servicenow

Step 4 – Roadmap your plan to expand support for DevOps with ServiceNow

KEY INSIGHTS

- Continue using the information you collected in your questionnaire (Step 1) to inform how you prioritize future work in your roadmap.
- Take advantage of the experience you gained by using the same process—and teams, when possible—to do future work.

Once you've built a foundation and proven value with an initial use case (Steps 2 and 3), you can expand the functionality you offer—either by enabling one of the other two initial use cases we recommend (such as enabling audit or pipeline performance dashboarding after delivering automated change management) or by branching out into implementing other, more complex functionalities.

Either way, we recommend following a consistent process to explore your options and make a plan for what to do next. To do this, complete these action steps. Keep in mind our customers' insights as they performed these actions.

Action step 1: Explore other use cases you can enable for DevOps teams

Start by reviewing the information you collected in your questionnaire (Step 1) and the top 10 use cases we shared in Table 3 (Step 2). List all other use cases in Table 3 that would deliver something that DevOps teams demanded strongly.

Also consider if there are other use cases that you think might add value to DevOps teams at your organization, even if they are not included in Table 3. Include those ideas in your list as well.

Record some information and context for each use case on your list, including justification for why you think the use case would be valuable—you'll use this information to prioritize what you work on next and what you add to your backlog in the next step.

Action step 2: Prioritize and backlog options in a roadmap

Take the list you assembled in the last action step. Answer the following questions for each of these use cases you're considering enabling:

- What percentage of DevOps teams at your organization (based on your questionnaire findings) do you think would benefit from this use case?

- How hard would it be to design and implement this use case at your organization?
- How hard would it be for DevOps teams to adopt this use case at your organization?

Based on your answers, draft a 12–18 month roadmap that prioritizes use cases that would benefit a large group of DevOps teams, that would be relatively less difficult to enable, and that you think DevOps teams would be able to readily adopt.

INSIGHT:

Invest in expanding your first solution before implementing new use cases

When reasonable, always prioritize projects to expand or improve on your first solution on your roadmap before implementing new use cases. Proving strong value from your first batch of integrations will make it much easier to justify expansion into new use cases.

Action step 3: Follow a consistent process to enable future use cases

Take advantage of what you learned when implementing your first use case when you start work to enable the next use case on your roadmap. The process you used in Step 3—along with any lessons you learned during that effort—should inform the process you use to implement new use cases. For example, the need to start by defining data sharing and process improvement needs still applies.

INSIGHT:

Work with your initial pilot team whenever possible

The process of implementing an initial use case (Step 3) provided an opportunity for your team to learn how to design, implement, and manage integrations to connect ServiceNow with DevOps tooling. Include the people from this initial team when possible for future work to take advantage of this experience.

The takeaway

It's critical that ServiceNow teams explore how they can use the platform to support DevOps at their organization now while DevOps is still predominantly an emerging practice. The right use case can deliver tremendous value to DevOps teams to ensure that they can move fast without undercutting the operational value and rigor provided with processes implemented on ServiceNow. Invest in discovering how DevOps teams are working at your organization, in understanding how ServiceNow fits into their work, and in delivering use cases that deliver value to them.

What does "good, better, and best" look like for this activity?

Good – The ServiceNow team has an understanding of how DevOps works at their organization and has modified how their processes work to reduce their burden of DevOps teams.

Better – Use cases have been enabled on ServiceNow to support and automate how DevOps teams work.

Best – ServiceNow is an integral part of how DevOps work gets done at your organization. DevOps teams benefit from the use cases enabled on your ServiceNow implementation, preferably without even knowing about them, because ServiceNow integrations work behind the scenes as DevOps teams use their own tools.

What should I convey to my team?

DevOps is an emerging practice that is quickly solidifying at many organizations. This emerging state makes it hard to know how to fit a mature ServiceNow implementation into that model, but we need to figure this out now before DevOps practices mature without applying the value that our ServiceNow can offer to them.

If you have any questions on this topic or you would like to be a contributor to future ServiceNow best practice content, please contact us at best.practices@servicenow.com.

Related resources

- Success Playbook – Automate incident and change management
- ServiceNow DevOps

servicenow

Appendix A

Initial ServiceNow DevOps use cases	Description	Desired outcomes	Solution	Availability in ServiceNow DevOps product
Automated DevOps change management	Automate the change management process by connecting to existing the CI/CD tool set to collect the data required to create change tickets, calculate risk, and automate approvals. This cuts the time needed for DevOps teams to obtain approval for changes.	• Increase developer productivity by eliminating the overhead involved in participating in change management tasks. • Release features into production faster. • Mitigate risk by bringing all code releases under change management control in one place. • Mitigate risk by making changes smaller and more frequent.	• Integrate with planning, development, build, and test tools. (We highly recommend using Jenkins, Azure DevOps, or GitLab for your building and deployment tool to reduce the level of effort required to implement.) • Create change tickets automatically via API. • Automatically calculate risk based on customizable business logic. • Automate change authorization based on the calculated risk level. • Deploy low-risk changes instantly and escalate high-risk changes for further review.	Available in the current product release

Automate audit reporting	Maintain an end-to-end audit trail that correlates the pipeline, releases, and incidents to eliminate manual audit data collection time. Effectively reduce the time needed for "reactive" audit data gathering from weeks to minutes by integrating with the tools that generate needed data with audits, aggregating it in ServiceNow, and enabling automated report generation.	• Dramatically shorten the time required to collect the information required for audits.	• Integrate with multiple CI/CD tools. • Aggregate and correlate data across development and operations data. • Automate report generation.	Available in the current product release
DevOps insights/continuous integration tracking (pipeline visualization)	Integrate across the tools in the CI/CD pipeline to gain visibility and to track the end-to-end CI/CD pipeline to dashboard performance and identify opportunities for improvement. Eliminate manual tracking mechanisms like Excel spreadsheets from the process.	• Help DevOps teams track performance and potentially increase their release speed while gaining visibility into tooling for ops. • The business determines when to release. • You have a transparent release pipeline. • Your time to release is faster. • Your releases into production are a higher quality.	• Integrate with multiple CI/CD tools: – Git/GitHub – Jenkins – Jira – SNOW ITBM – BitBucket – Azure DevOps – GitLab – SonarCube – Selenium • Automate workflow orchestration to replace Excel tracking. • Track developer performance.	Available in the current product release

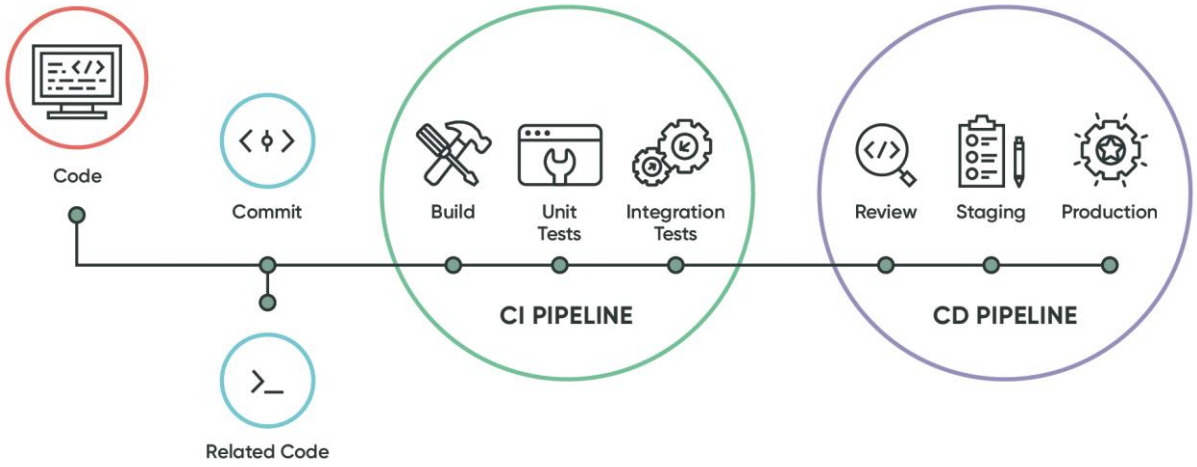
Developer onboarding	Reduce the time to productivity for new developers by enabling developers to onboard quickly and become productive on day one and automate access control to development environments and IDE/PaaS access. This helps attract and retain developer talent.	• Attract and retain more millennial developer talent. • Reduce time to productivity for new developers.	• Consumerize the developer onboarding process with a mobile application that uses ServiceNow Service Portal. • Provide a customized HR portal for day one of onboarding of developers. • Integrate end-user compliance. • Connect, communicate, and set any expectations.	Requires services and custom work

DevOps self-service portal	Empower developers to self-serve by providing a single system of engagement for development teams to collaborate and communicate requests and delivery through workflows for shared service. Also create a curated catalog of developer tools and libraries. This reduces the "swivel chair" effort for development teams to work across different tools. It also helps ServiceNow teams get greater visibility to DevOps processes.	• Provide greater visibility to DevOps processes. • Enable elf-service for developers and testers. • Reduce your dev teams' "swivel chair" efforts.	• Use the Service Portal for building custom dashboards for each actor in the development cycle: – Developers – Testers – Executives – IT operations • Integrate the service catalog with PaaS.	Requires services and custom work

Incident/problem management integration	Integrate problem management with development planning tools to introduce development work that needs to be done to resolve recurring incidents directly into the development lifecycle. Operationally, incident and problem management become easier to manage because managers have more information (pulled from development tools via integration) in tickets.	Integrates the development work needed to resolve recurring problems directly within your development team's workstream.	• Integrate incidents and problems with defects. • Link defects or enhancements to the requirements backlog for continuous improvement.	Requires services and custom work
Predictive operations and code reverts (ITOM performance management)	Tie DevOps changes to related infrastructure (by tying all information in the change record to relevant CIs) so that if there is a problem with infrastructure (such as an incident), you have more information to identify the issue than in the past, help root cause analysis, and revert faster.	Speeds up how operations are able to react to bad releases from CI/CD toolchains.	Using insights from work already managed in the Now Platform, tie production incidents back to releases and the planning pipeline, completing the feedback loop from ideation to operations.	Requires services and custom work

Appendix B

Continuous integration/continuous delivery pipeline



Source: https://dzone.com/articles/learn-how-to-setup-a-cicd-pipeline-from-scratch

Appendix C

DevOps tools landscape