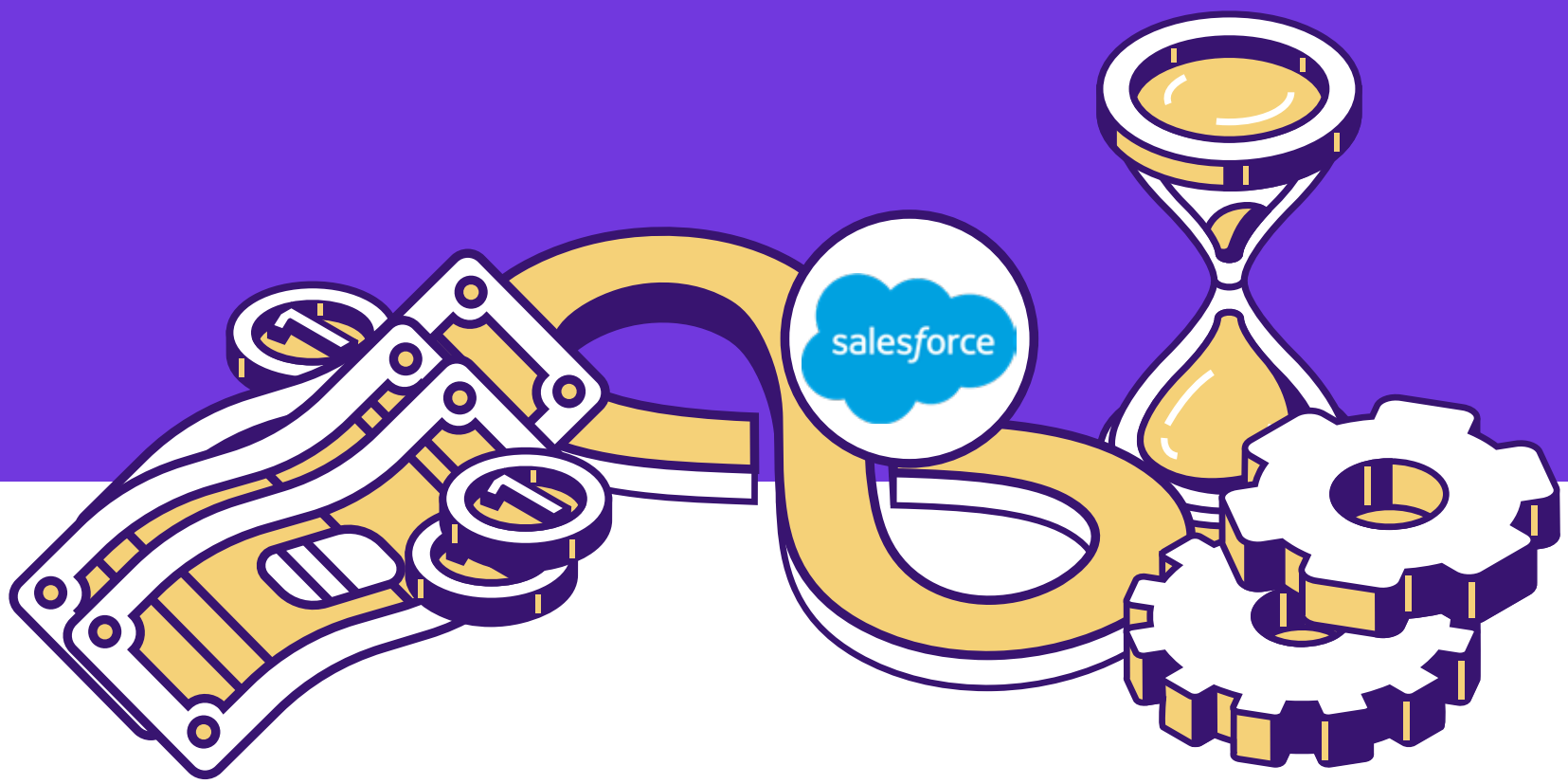
OPSERA

Build vs Buy:

Making the Right Choice for Your Salesforce DevOps Solution



To Build or To Buy...

Over the years, Salesforce has evolved from a CRM platform to the centralized hub for all your applications that touch various sides of your business: from sales, to marketing, to customer success, and finance. Salesforce is the de facto choice for building any custom SaaS application. Yet, many IT teams are still far from having a centralized solution to their Salesforce releases.

Large organizations might leverage their existing software team to build an in-house DevOps solution. Building allows them to customize the solution specific to their business needs. Building also offers more control over what features and integration to prioritize. Some large teams want the DevOps solution to eventually be part of their core product, maintain it as proprietary software and keep all their custom codes. Building is a viable route if the organizations have the resources to document and train the users on new features update and version changes.

Building a Salesforce DevOps solution in-house can take anywhere from 6 months to 2 years, including 2 to 5 engineers to accommodate the weekly updates from Salesforce. Meanwhile, buying a solution can save you valuable designing, developing, and testing resources. To help your team decide whether to build your CICD pipelines for Salesforce by bringing in Jenkins, Bamboo, or other toolchains, or picking a solution off-the-shelf, here are a few perspectives to consider:

1. Scalability

Is there a Jenkins expert on your team who people always consult for specific problems?

They may be the sole engineer with years of experience operating in Jenkins. However, as your business scales, knowledge transfer is vital to avoid project disruption, even in the case of employee rotation. Your only Jenkins experts can quickly become the bottleneck, slowing down new projects, builds, and releases.

As a self-service platform, Jenkins has clear limitations: it doesn't keep state and isn't aware of the assets it created/modified or deleted, therefore it's impossible to keep track of all the inputs by developers. Your DevOps team will end up putting fires instead of focusing on shipping products.

2. Visibility

As an IT leader, you need centralized information for auditing or rollbacks to manage your team's work.

The only way to do it in Jenkins is to download plugins. If an issue occurs, you rely on one home-grown solution owner to remedy it since your tool doesn't allow parallel processing.

3. Security

Vulnerabilities are easily introduced through Jenkins plugins. Anyone can install Jenkins plugins, making it impossible to assure the quality, clarify the plugin owner, the frequency of use and its dependencies.

The system special permission in Jenkins, by default, allows any action executed during the build to do whatever it wants. To prevent security incidents, you have to configure per-project build authorization strategy, limit roles using the Role-based authorization strategy or remove unused plugins.

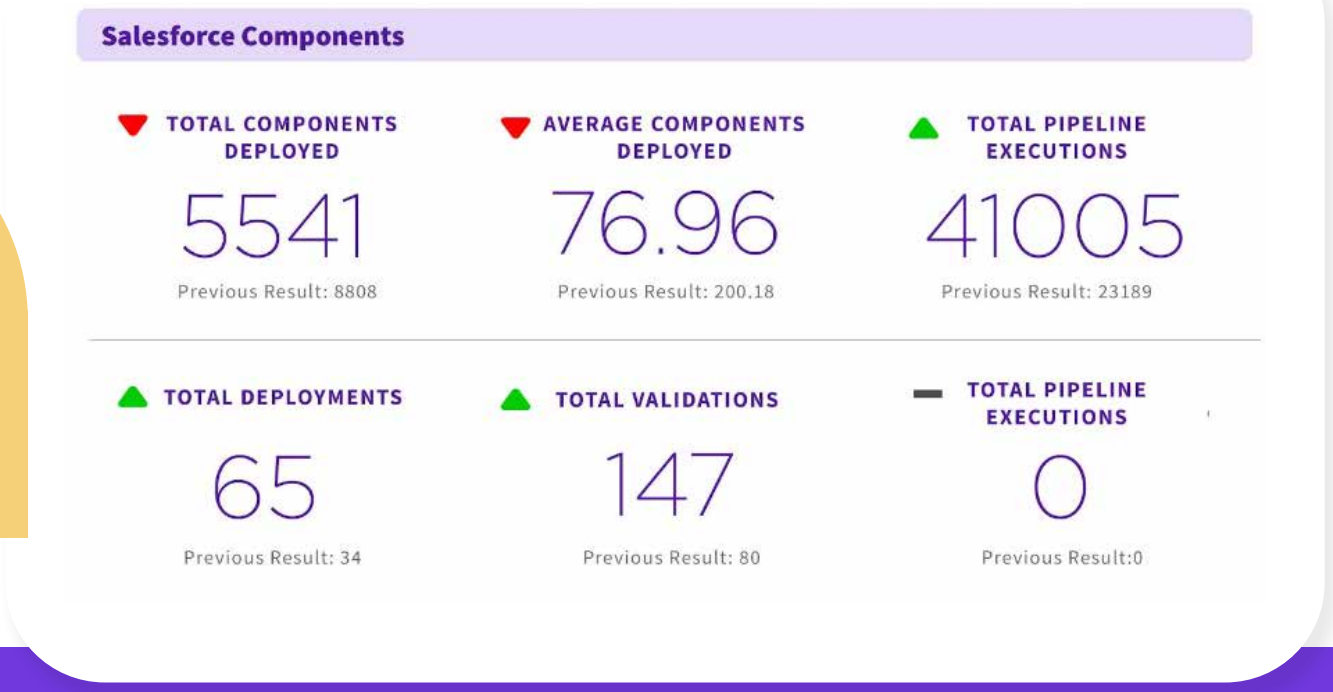
Taking proper controls of Jenkins is time-consuming. According to CVEdetails, most of the vulnerabilities published are due to Jenkins plugins.

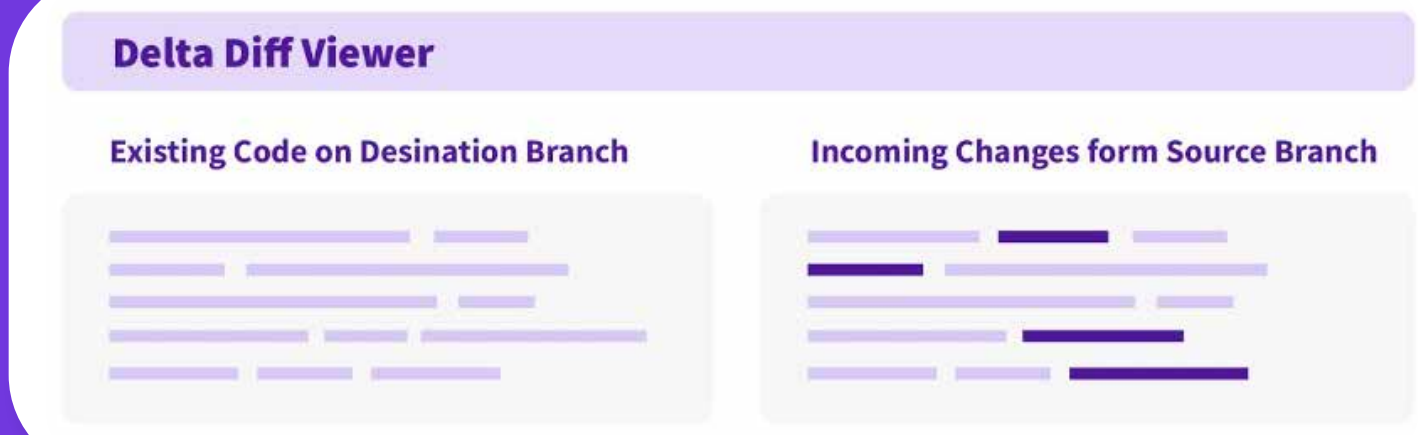
Here's the list of common capabilities you should review for your Salesforce DevOps

Capabilities	Home-grown Salesforce DevOps	Opsera Salesforce DevOps
Scalability		
Merge Conflict Resolution	Manual / Error Prone	Git merge sync wizard highlights conflicts between the source and destination.
Profile Migration	Manual / Difficult	Move changes with the delta. Require permission.
Rollbacks	Manual, time-consuming process	Automatic metadata backup enables rollback
Ongoing maintenance and upgrades	Time and money to continually build new features and automation. Updates must be performed almost weekly.	Roadmap defined and implemented for new features and functionality. Support for upgrades.
Deployment frequency	Process to be rebuilt for any changes	Low code or no code pipeline with configuration speeds time to move changes to production
Streamlined user	Manual. The users need to learn CLI and coding with ongoing investment.	Simplified UX with Wizard based pipelines, configuration and execution via CLI, bulk uploads, one click notifications and bring your own tools
Environment Sync	Manual, time- consuming process.	Org to org sync standard
Branching Strategy	Keeping branches in sync within an org is challenging. No UI-based editor. All activities need to be done outside of Jenkins and within Git.	Support for any branching strategy, version control with a built in editor for conflict resolution. Bulk migration to keep branches and org in sync.
Cherry picking of components and deployments	Need to build the capability	Out-of-the-box
Salesforce compatibility	Salesforce updates will break processes and cause outages. Not compatible backward and forward	All new features and releases are CLI and API compatible backwards and forwards.
Integration with Devops tools	Every tool integration plus updates needs to be coded.	One click integration with 50+ tools in the market (eg Slack, Jira, Provar, Sonarqube)
Implementation	From six months to two years, plus ongoing maintenance, potentially costing millions.	1 day to 1 week installation, with virtually no maintenance required.
GITOPS	Requires significant coding and development to build	Source driven development standard with no/low code.
Visibility		
Pre/post Deployment	Not supported	Steps tracker for pre and post activities
Insights	No support for insights to track productivity	150+ KPIs to track productivity.
Testing	Must build the capability for a manual process.	One click integration with top testing tools, automated testing.
Security		
Security	High cost requiring the right security technology to be built	Out-of-the-box security policies with code analyzer
Compliance	Compliance needs coding and upkeep.	Included: Audit report, role based access, customizable value stream KPIs and log comparison standard.

Opsera Salesforce DevOps allows you to release 80% faster, leveraging your existing team and tools. Opsera has a multi-SaaS architecture with SaaS Apps Release Automation Strategy.

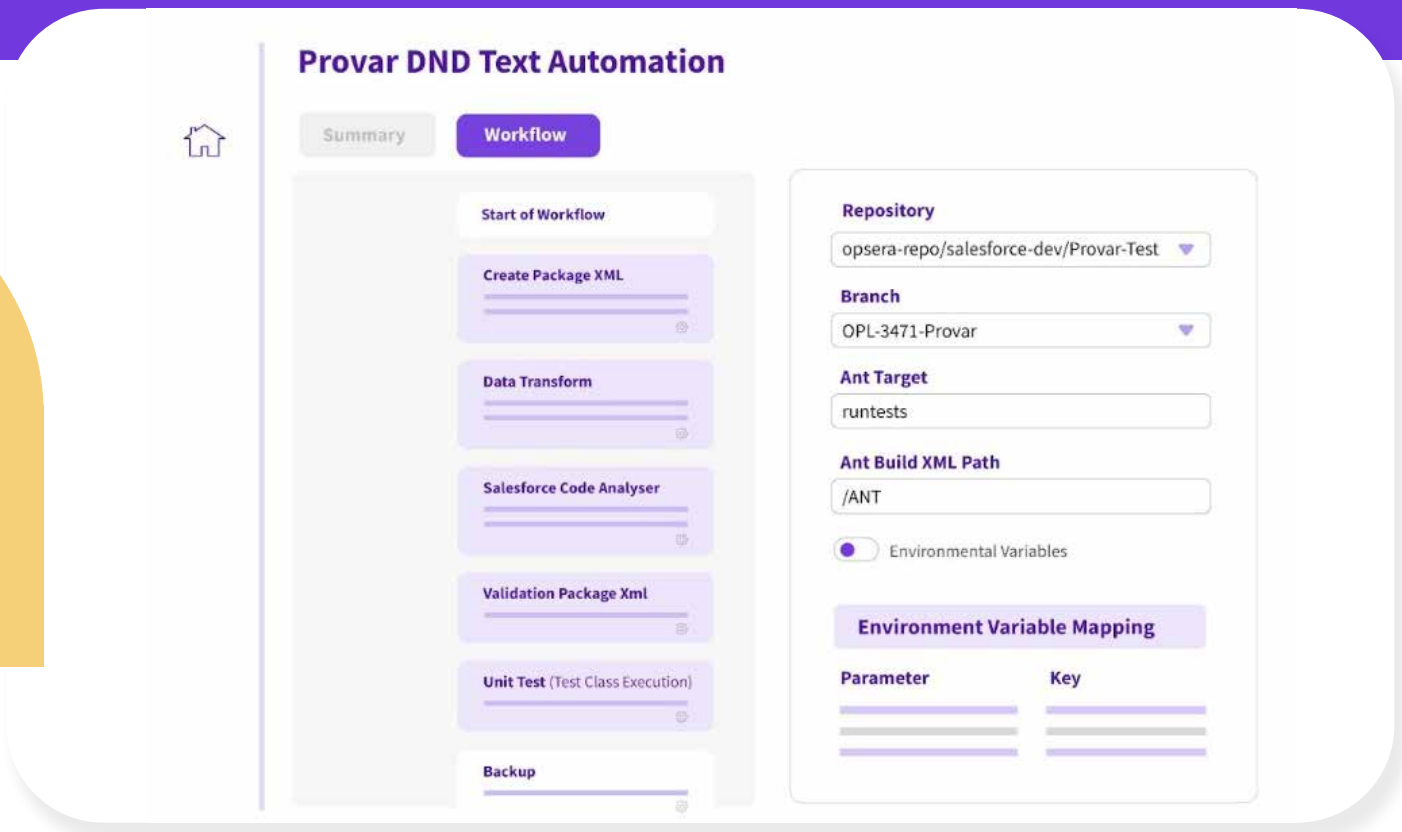
Gain visibility into your Salesforce team's progress





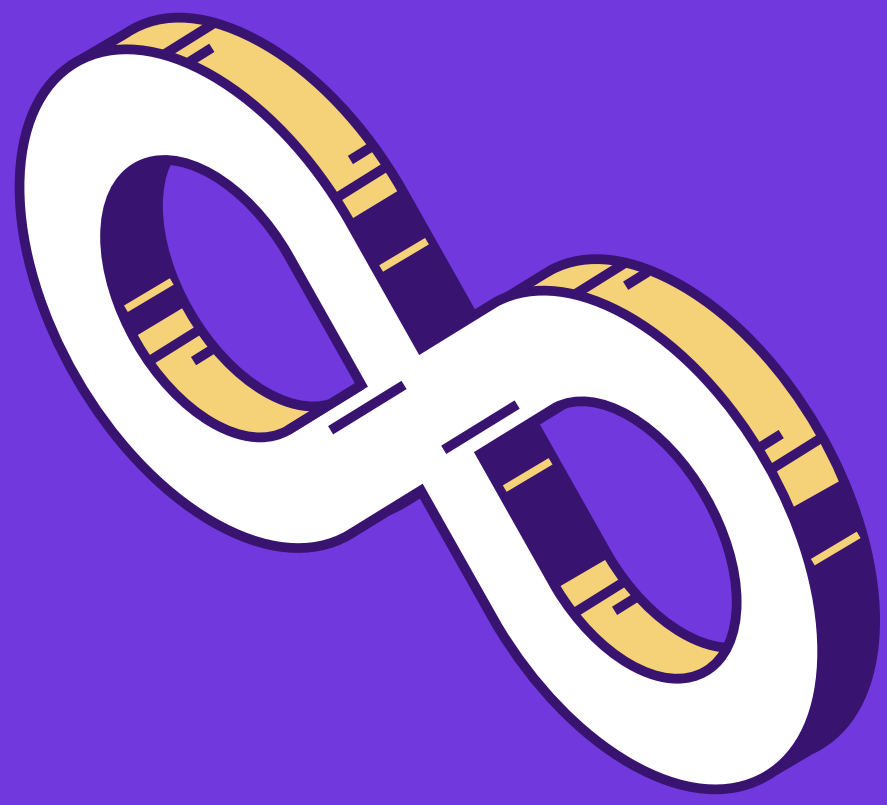
Let your developers compare branches of code before merging

Stay away from future tech debt: allow your team to monitor code quality



Opsera Salesforce DevOps is part of the **Opsera Flow Platform**, trusted by high-performing engineering teams, customers and recognized by the industry.

Built on three core pillars to give leaders visibility from ideation to deployment, quality software delivered at speed, and shift-left security built into the platform so teams don't need to worry about protecting current or future software investments.



See Opsera Salesforce DevOps

Couchbase

NortonLifeLock™

Infoblox
NEXT LEVEL NETWORKING

Honeywell