DevOps, Continuous Delivery, and Release Automation

How to Calculate Return on Investment and Benefits

Business and IT leaders are demanding high-quality solutions faster, and are looking for more effective processes and tools to help realize the required improvements. Enter DevOps, Continuous Delivery (CD), and Application Release Automation (ARA), which apply agile processes and automation to address the issues with traditional software delivery. While companies understand the importance, many are immature with respect to their approach to standardizing the orchestration and automation of the build, deploy, and release lifecycle. DevOps, CI/CD, and Release Automation are driving significant focus on improving collaboration and automation across IT, enabling businesses to address their need for speed, quality, and better cost dynamics.

Flexagon's DevOps platform, FlexDeploy, standardizes, orchestrates, and automates the lifecycle for provisioning, building, deploying, and releasing applications, middleware, database, and other related artifacts across environments. While FlexDeploy helps achieve the benefits described, this paper is written to be vendor/product neutral, and generalizes the ROI and value associated with DevOps, CD, ARA.

How does automation tooling bring value and ultimately a return on investment? What are the key benefits that can help justify the focus?

This paper helps you answer such questions, and in many cases, quantify the value in hard dollars. Flexagon views Application Release Automation spanning Provisioning, and the entire lifecycle of Build, Deploy, and Release of the infrastructure, database, middleware, and applications across non-production and production environments.

Not everyone prioritizes benefits the same, so you will need to decide what matters most for your organization. For example, a deployment related outage in production can have significant cost implications for some companies, and therefore would rank quality improvements and a reduction in deployment errors as a top priority. Others need to improve the speed and agility of software delivery and subsequently focus on increasing the frequency at which deployments can occur.

Benefits can be broadly categorized across 6 areas:

Reduce manual labor and scripting

Automation tooling eliminates manual work and eliminates or greatly reduces scripts via pre-built plugins. This is the easiest category of cost savings to convert into hard dollars. Some companies have people dedicated to handling the deployment and release process, while others spread this work across many individuals within development and operations/administration roles. In either case, you can calculate the hours spent performing manual work and the time spent creating and maintaining scripts and other tools which support the deployment processes.

Whether using manual and/or scripted processes today, make sure you consider the time required to update the processes and scripts over time. In many cases, the products/technologies you are using change their deployment techniques over time. Your current manual processes and other scripts will need to be updated accordingly. When using DevOps tools which support Continuous Delivery and Release Automation, the burden of creating and maintaining the plugins/integrations is taken on by the vendor and not your internal staff.

The number and complexity of environments, technologies, projects, and applications can significantly increase the overall cost. When calculating the cost associated with your process and tools, make sure to capture all deployment related activity. For example, companies using Oracle Fusion Middleware might have WebLogic, SOA, E-Business Suite, and BI products used across 2-6 non-production and production environments. Calculate the cost for each product, across all environments, and for all projects executed in a given year.

Reduce time spent debugging and fixing deployment related errors.
Manual and scripted processes are often error-prone and result in development, operations, and other IT support personnel spending significant time troubleshooting failures. Capture the details of failures across all environments and the time spent on error-resolution activities.

While capturing the hours spent debugging and troubleshooting deployment-related issues, it's important to include times when the failures cause war-room like conditions. This includes the management team being involved until the problem is either rectified or the changes are rolled back to a prior version. The waste associated with those situations can cut across more than the technical staff directly involved with issue resolution, so make sure to capture those details.

Reduce deployment related outages across environments
Outages can impact productivity of development and test teams and have a direct business impact for both internal and external customers. The prior category captured the time spent getting the deployment related issues resolved by the technical teams and other leaders overseeing the troubleshooting. This 'environment outage' category captures the cost related to the users impacted because an environment is not available or there is some type of service interruption. For example, when a deployment fails in a Test or QA environment, it might take several hours to debug and resolve the problem. Developers take a productivity hit because they can't test their changes or develop new features. In many cases, testers (IT and business) are lined up to perform their test activities and are left in an unproductive state when the environment isn't available.

When the outage occurs in production, the impact can be significant. For example, when a critical revenue generating ecommerce system isn't able to accept customer orders, or when customers cannot get their issues resolved due to a support system being down. It is not always easy to quantify; however, the negative impacts and costs are real and can be very difficult to recover.

Increase speed and frequency of software delivery

As many companies put more focus on providing innovative products and services to their customers, the demand for agile processes for software delivery goes up dramatically. How can improvements in speed and frequency of delivery be quantified? You can compare the current deployment processes to automated processes, analyzing factors such as:

- Faster and higher quality deployments reduce development/test cycles. The hours/days saved as part of a project can be significant and result in software delivered to the market earlier. The results can have direct impacts to revenue generation, business operation efficiency, and customer satisfaction. Depending on your business and type of software being delivered, you can calculate the financial impact of delivering software to the market faster.
- Due to the automation, the time it takes to make a change will decrease system downtime in the form of outage windows for deployments/releases. This can be valuable, especially for revenue generating systems.
- Amount of change in a Release can increase because the deployments can be parallelized.
- If non-deployment related issues surface after a deployment/release, the rollback can occur faster, mitigating the impact to the end users.
- Faster feedback cycles ensure the solution hits the mark, reducing time and money spent by developers, testers, and everyone involved in the solution delivery.

Decrease time spent proving compliance and the impact of failing an audit

Standardizing and automating the deployment process results in visibility to the "who, what, when, and where" across all environments. The IT team is better positioned to demonstrate compliance to a controlled process, and improves the likelihood of passing an audit.

Improve employee satisfaction and retention

Deployment activities into production environments are often performed on nights and weekends. Lengthy and error-prone processes can result in challenging situations for key IT resources expected to support the deployments. The late-night activities can be mentally draining and increase risk, as tired and stressed developers and operations team members don't perform consistently well under those conditions.

Enabling teams with processes and tools which improve the deployment lifecycle can have multiple benefits in terms of employee satisfaction and retention. First, it will increase the speed, quality, and efficiency of off-hours deployments, reducing employee stress. Faster and higher-quality deployments can result in the ability to move changes from nights and weekends into normal business hours.

Second, automating the deployment process allows staff members to spend their time performing higher value and more enjoyable activities, therefore improving job satisfaction and retention. Increased retention has an additional impact on the company, as the cost of recruiting and training new employees is high.