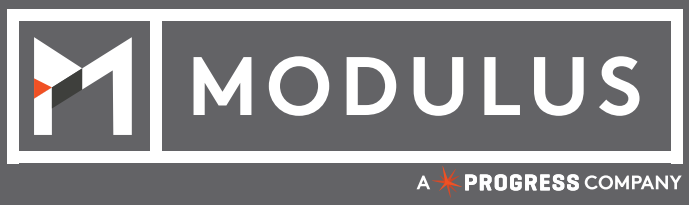


DEVOPS & NODE.JS

Optimizing Business Outcomes

JANUARY 2015

DEVOPS & NODE.JS
OPTIMIZING BUSINESS OUTCOMES

Abstract: DevOps, the combination of software development and IT operations, two previously separate organizational groups, can accelerate the creation and deployment of new software. By breaking down barriers, creating transparency, and minimizing "handoffs" between the two teams, DevOps makes rapid innovation cycles possible. In parallel, the IT industry has seen the emergence of Node.js, a development platform that enables more streamlined software development processes through the use of JavaScript on the server side. Together, DevOps and Node.js promise an even faster-moving software development potential than has been traditionally possible with existing development platforms. This paper explores how DevOps and Node.js reinforce their respective benefits while also highlighting solutions to some of the challenges inherent in making DevOps and Node.js work together effectively.

INTRODUCTION

The corporate world has been blitzed in recent years by changes in customers' technological expectations. The maturing of the Web, coupled with the widespread proliferation of smartphones and tablets, has created a business environment where consumers simply expect to have an online relationship with their favorite brands. They expect to have access to data and commercial processes directly through personal devices. Corporate software users similarly assume that data and workflows will be accessible through rich, mobile interfaces on the device of their choice.

This dramatic shift in user expectations has led to changes in the way that software development is performed and managed. Developers feel pressure to release code quickly and update it often with new features learned from frequent customer feedback. The traditional "waterfall" approach to developing business applications is in decline, especially for customer-facing, Web-based, and mobile apps. Instead, the new mode of development features agile methodologies and continuous integration of code. These are just two of the many novel approaches to corporate software development geared toward staying current with fickle users.

As rapidly evolving user requirements and fierce competition drive an arms race of ever-quickening code release cycles, many IT departments are also exploring new approaches to the traditional develop-test-release process. The most notable of these new methods is DevOps, which combines software development and IT operations—two previously separate organizational groups—into a single, coherent team. By breaking down barriers, creating transparency, and minimizing "handoffs" between the two teams, DevOps makes it possible to get new code up and running with customers more quickly.

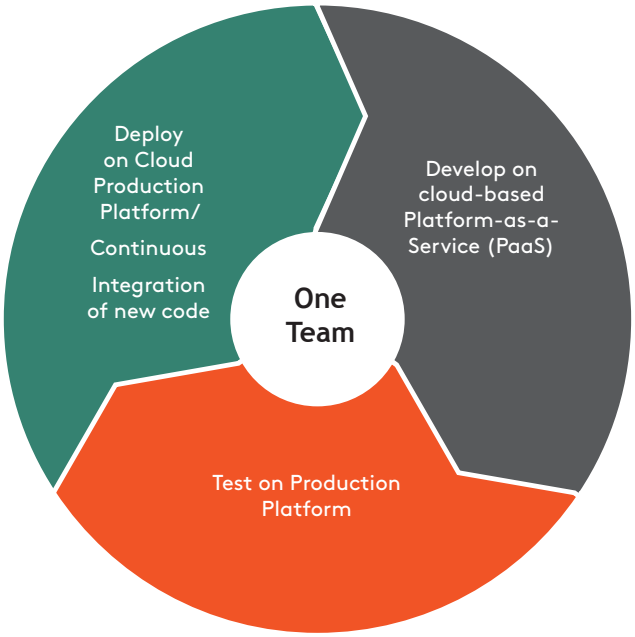
In parallel, the IT field has seen the emergence of Node.js, a development platform that enables more streamlined software development processes through server side use of JavaScript. Together, DevOps and Node.js promise even faster software development than existing development platforms. This paper explores how DevOps and Node.js reinforce their respective benefits, while also highlighting solutions to some of the challenges inherent in making DevOps and Node.js work together effectively.

WHAT IS DEVOPS?

To understand DevOps, you need to know what came before. The term "DevOps" is a contraction of three traditionally separate IT groups: software development teams, testing teams, and IT operations. In traditional software development, each group had its own distinct role in the process of creating and deploying working software to end users. Developers worked with business analysts to gather requirements. After putting the requirements into a development plan, the developers would start to write code. When the code was complete, the developers handed it off to testing, which ran it through various tests to ensure that it worked properly. Then, the software would get handed off to IT operations, which would carefully configure it and make it available on proven production server resources.

Figure 1 – The traditional software dev-test-deploy process (The "Waterfall").

This was known as the "waterfall" method, as shown in figure 1. Each step followed a well-thought out progression designed to ensure software quality and reliability. The waterfall method is still in use and quite relevant to certain types of software projects. For example, large-scale back-end corporate development projects may be well-suited to the waterfall approach. However, the waterfall presents a number of difficulties for IT departments that are responding to a business mandate to move very quickly. Today, many development teams face pressure to introduce successive iterations of software in less time than it would take to complete just one step of the waterfall process. DevOps was devised to solve this problem.



DevOps involves forming a cross-functional team out of developers, testers, and operations people. DevOps breaks down the organizational barriers that impede software development in the waterfall. With DevOps, developers and IT operations people work together, which minimizes time-consuming handoffs. The DevOps approach heightens transparency; it's easier for everyone to see what is going on. With a lot less "throwing it over the wall," DevOps can maximize the flow of new code from concept to deployment.

Figure 2 shows the DevOps cycle.

New development tools and deployment platforms have played a major role in making DevOps possible. While not essential, cloud-based Platform-as-a-Service (PaaS) platforms are at the heart of DevOps. When developers, testers, and IT operations people can all work on the same code on the same platform, it becomes much easier for people to see what is happening and correct issues in real time. The single platform also removes some of the obstacles teams traditionally faced when developing in one hardware environment, testing on another, and deploying on a third. The introduction of "Continuous Integration" (CI) tools make it possible for developers to insert new code into production applications more or less at will without affecting end user experience.

HOW DEVOPS GENERATES POSITIVE BUSINESS OUTCOMES

DevOps can contribute to positive business outcomes in a number of ways. By speeding up the software development process, DevOps has the potential to deliver customer-facing value quickly. Applications that make a business competitive flow to the end user in a faster timeframe, keeping the business in the lead. Similarly, DevOps improves responsiveness to customer and end-user needs. For example, if financial services consumers want an app that links their bank accounts with life insurance policy balances, the company that provides it first and best will have an advantage. Then, if consumers subsequently say they want a built-in calculator that compares term life to whole life insurance, a DevOps team can iterate and use continuous integration to put that feature into production far more quickly than would be possible with the waterfall method.

To understanding how DevOps helps businesses be more flexible and agile in adapting to customer needs, consider two of many practical challenges that DevOps solves:

- Monitoring and optimization – An application in production needs to be managed and monitored to ensure that it is providing agreed-upon service levels to users. In the traditional development model, IT operations was responsible for determining the best way to monitor and optimize the application as it went into production. In DevOps, on the other hand, the Dev and Ops people can collaborate early in development to figure out monitoring and management issues before the app is deployed in production.

- Deployment automation – For Operations to work efficiently with an application, they need to automate the deployment process. The server operating systems, databases, network settings, and so forth, should be automated. DevOps compresses the time required to set up the automation. It enables Operations to define the deployment automation so that Development can implement it in early stages and turn it over to Operations later.

One classic hurdle facing businesses seeking technological differentiation was the disconnect between line of business (LOB) people and their IT counterparts. Typically, the LOB perceived IT to be slow in delivering on expected requirements, while IT struggled with mandates to provide secure, reliable systems. Within the traditional software lifecycle, it is difficult to be fast, economical, and provide high quality at the same time. DevOps helps ease some of these constraints. DevOps enables multiple stakeholder groups to work together more efficiently to deliver business value.

NODE.JS AND DEVOPS

Node.js is an open source software platform that uses the JavaScript programming language for an application's front-end interface and back-end, server-side "runtime." Creator Ryan Dahl was inspired to build Node.js in 2009 after becoming frustrated with the slow upload progress he saw for a file on the Flickr service. Dahl's browser did not have a way to measure how much of the file had been uploaded, so it had to query Flickr's Web server repeatedly. This was inefficient, a poor use of the network, and a waste of backend server resources. He realized that there might be a better way for a Web application to handle input/output (I/O) over the network. Dahl solved these problems by having Node.js utilize an event-driven, "Non-blocking" I/O. With this event-driven architecture, Node.js makes far better use of the network and server than other Web platforms. For this and many other reasons, Node.js stands out as the one platform that promises to deliver on the potential of DevOps for desirable business outcomes:

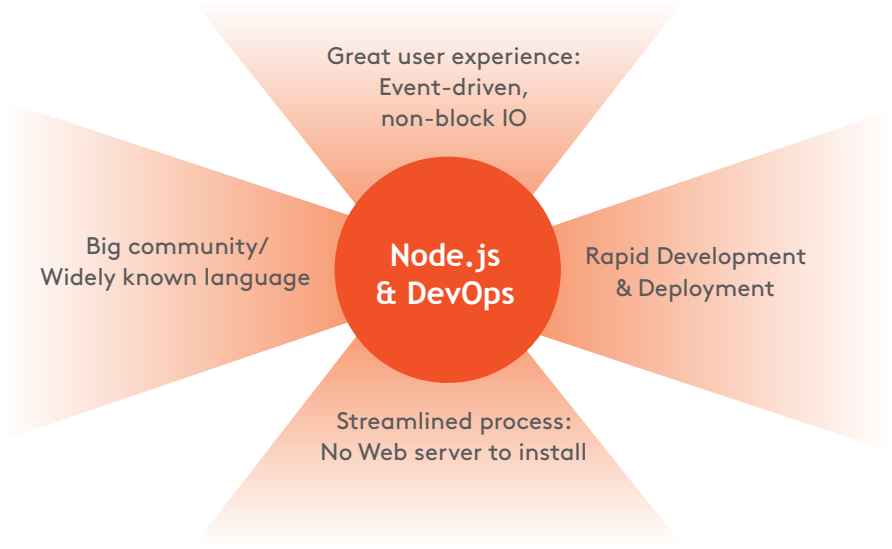


Figure 3 - The combination of Node.js and DevOps promises rapid delivery of apps with great user experience: a fulfillment of the promise of positive outcomes with DevOps.

- **Node.js makes it relatively simple to build fast, scalable network applications.** The platform has a library that enables applications to act as a Web server without the need for software like Apache HTTP Server or Microsoft IIS. Not having to set up a Web server shortens the DevOps team's "to do" list and streamlines the flow of code going from development to deployment. In addition, the widely known JavaScript language is used for both the client and server. Developers get the efficiency advantage of code reuse. Node.js apps can run on the Node.js runtime on OS X, Microsoft Windows, Linux, and FreeBSD.

- **Node.js is designed for fast operation and great user experiences.** With its event-driven, non-blocking I/O model, Node.js enables applications that are lightweight and highly efficient. The platform is considered suitable for data-intensive, real-time applications running on distributed devices. The end user experience also tends to be great because Node.js optimizes an application's throughput and scalability.

- **Node.js has a big community and broad adoption.** Many large, thought-leading organizations have embraced Node.js, including Groupon, SAP, LinkedIn, WalMart, and PayPal. Javascript itself is a ubiquitous Web language, known by almost all developers regardless of other platform affiliations. As a result, the Node.js community is huge and very active, with many people engaged in multiple

projects aimed at enhancing the platform. The explosive growth of npm alone (doubling from over 50,000 to over 110,000 in the past year) validates the number of Node.js modules that are being developed and used throughout the Node community.

CHALLENGES WITH NODE.JS AND DEVOPS

Node.js and DevOps are well-suited to each other for the fulfillment of business outcomes. However, like any coupling of an IT methodology and a platform, DevOps and Node.js will only be as good the people who implement them, their managers, the tooling, and the processes they've agreed to use. The pairing is not a guaranteed, push-button scenario.

Teams that take on DevOps and Node.js face a risk that they will inadvertently block the realization of preferred business outcomes. Without the right tooling, DevOp's merging of roles and duties can create chaos or unintended outcomes. These include issues related to reliability, load management, and scaling. One consequence of rapidly rolling out new iterations of an app in today's device-rich, connected world is the potential of viral adoption. As app use spreads, the operations team will have to scale Node.js instances to meet demand. If the users are in different geographies, this can present a territorial load balancing challenge. An app's achievement of desired business outcomes usually depend on consistently good user experiences. Any obstacle to achieving that goal is a threat.

Versioning and integration also present potential challenges. Rapid-cycling iteration on new code invariably means managing multiple versions of an application. The operations team needs to ensure that the deployed version is the right one, on every server. Integration is a related issue. Many apps today connect with other software, often located in other domains. Developers have to stay on top of app-to-app dependencies, which present both logical challenges and potential load management issues. For instance, if a third party app that sends procedure calls to your app starts to heat up with high adoption, it can create a surprise need for scalability. If you can't react and scale quickly, the third-party app partner may be dissatisfied with the results of the integration.

There's also the problem of getting distracted by the Node.js platform itself. Though Node.js has been proven to help with speeding well-performing apps from concept to end-user, setting up a workable Node.js environment can be challenging. Even if done well, maintaining the platform in production and ensuring a good experience for developers, operations people, and end-users can take a fair amount of focus and resources.

MODULUS: ACHIEVING THE BUSINESS POTENTIAL OF NODE.JS AND DEVOPS

Modulus, a Progress Company, offers IT departments a way to realize the business potential of Node.js with the DevOps methodology. Modulus is a Node.js hosting platform with public and private hosting options. By providing a secure, full-featured, and cost-effective PaaS hosting environment for Node.js, Modulus streamlines many of the platform aspects of DevOps and enables team members to focus on code and processes, not infrastructure.

Modulus is a DevOps force multiplier, automating much of the operational side of the DevOps process. The heart of the Modulus DevOps capability is its seamless horizontal scaling through mini-servers called "Servos." Each Servo runs an instance of a Node.js application. Modulus automatically handles load balancing between them, no matter how large the application gets. Modulus will always scale to meet demand. The platform also gives users powerful statistics, mobile management, and built-in integration with MongoDB.

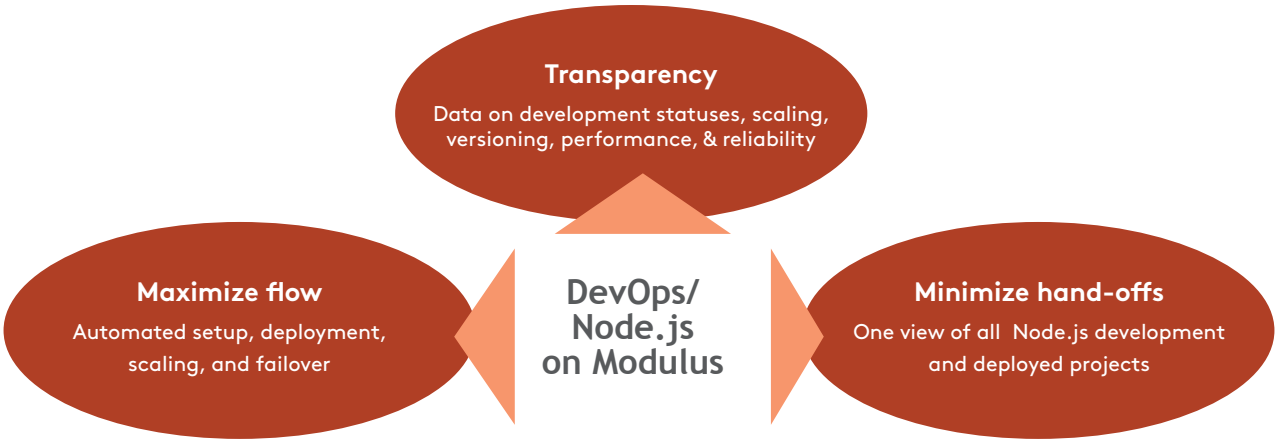


Figure 4 – Modulus enables the key principles of DevOps: Transparency, maximization of flow, and minimization of hand-offs.

Modulus facilitates business outcomes with DevOps and Node.js by removing barriers to the flow of code from development to successful deployment (Figure 4). By providing a single platform with extensive deployment automation as well as data on app performance, Modulus enables the key principles of DevOps:

- **Transparency** – Everyone on the team can see the status of all Node.js apps in development, revision, or deployment. Everyone can see real time data on performance, reliability, and scaling through Modulus' statistics, advanced logging, reporting, and analytics.

- **Minimize hand-offs** – With one central platform, everyone on the DevOps team is working from the same code on the same infrastructure.

- **Maximize flow** – Automated deployment and scaling accelerates the process of getting apps out to end users.

Modulus was built for application developers working in tandem with Operations. The platform enables them to collaborate early in development on the requirements to optimize the performance of an application. A Modulus project is instantly set up with access to custom SSL, WebSockets, and unlimited persistent disk and database storage. Modulus tracks every route to the app individually. IT operations can identify the parts of the application that get the most hits and parts that cause performance issues in real-time. Modulus enables IT ops to add servos or manage load using a Web interface. When a performance issue emerges, IT ops can take action immediately.

Modulus also provides the option of letting DevOps effectively leverage the Modulus platform and team as an IT operations department. One Modulus client, the Differential app development agency, does just this. They view Modulus as their operations team so they can concentrate on coding. Colin Flynn, Differential's General Manager, commented, "It was really an issue of focus. We were doing great work in Node.js, but we lacked a coherent development environment and our attempts to find a hosting platform were not going well. Would we have to take our eye off the development ball and allocate people and money to the platform and hosting issues? We preferred not to. With Modulus, we can flip a switch and turn servos on and off to scale quickly. The platform lets us troubleshoot rapidly. As a PaaS layer, Modulus gives us the ability to react if we get hit with traffic. We can add servos with the click of a button and Modulus handles all of that for us."

CONCLUSION

DevOps, like Node.js, is new and evolving. The story has not been completely told yet. However, what's clear is that the DevOps and Node.js, which radically simplify the app development and deployment process, are rewriting the industry's understanding of what "dev" and "ops" are really about. DevOps with Node.js on an automated hosting platform such as Modulus blurs the line between the developer and operations roles. Developers can do operations. Operations people are deeply involved in development. There are fewer handoffs because individuals can handles tasks that used to be performed in different groups. As LOB managers become more familiar with this potent form of DevOps, they are likely to embrace the methodology as a path to business advantage through technological leadership.