DevOps Journeys........................................................................................................................................... 3

A DevOps of One........................................................................................................................................ 3

The DevOps Scrum Team................................................................................................................................ 3

The DevOps of many...................................................................................................................................... 4

Delivering DevOps of Business......................................................................................................................... 5

Delivering DevOps ROI.................................................................................................................................... 6

Examples DevOps adoption delivering at scale............................................................................................... 6

Review....................................................................................................................................................... 6

Steps to build or transform to DevOps ........................................................................................................ 7

Assess the current state of your organization and scale requirement............................................................. 7

Gap Analysis between current and desired...................................................................................................... 7

Design the transformation in prioritized stages............................................................................................. 7

Project review and modification..................................................................................................................... 7

Commit and begin transformation.................................................................................................................. 7

Celebrate Success......................................................................................................................................... 8

What might the technical architecture or environments look like?................................................................. 9

What might the SDLC workflow look like?....................................................................................................... 11

What issues still exist?.................................................................................................................................... 11

Must it be this way?.......................................................................................................................................... 11

The social sides of the equation.................................................................................................................... 12

What are some common practices in Development which exist also in Operations?................................. 13

What are some common practices in Organizations?.................................................................................... 13

About Zefflin Systems.................................................................................................................................... 13

Further resources........................................................................................................................................... 14
INTRODUCTION

DevOps is about people, process, and tools, working together. DevOps Transformation is about change from an existing working environment to a new environment encompassing a DevOps model. If you don’t already have development and operations teams you may not transform anything, but form a DevOps team from scratch. This paper will highlight a best practices approach, key risk areas, and what you can expect from DevOps, to help you make informed decisions.

In simplest terms the word “DevOps” is a Portmanteau which, by definition, is a term formed by munging two words together into a compressed or tighter single word. In practical terms, it involves taking two huge and complicated organizations and their associated practices, priorities and processes, “Development” and “Operations” and making them a single, powerful coherent entity.

Traditionally, application development and IT operations have been organized in silos, with clear delineation of responsibilities. App dev has been responsible for design, code, build, test, bug fixes and QA, where operations was responsible for production cutover, migration, support and reporting back of bugs. Because of the clear lines of responsibility, hand-off processes had to be derived and implemented. The result? Both sides optimized their respective organizations as they saw fit. This seemed like progress, but in fact resulted in inefficiencies, finger pointing and less then optimal use of resources. DevOps as a discipline seeks to correct this situation and establishes a single set of processes and underlying technologies that enable the next generation of optimization, efficiency and cost reduction in IT.
DevOps Journeys

A DevOps of One.

One day you have found yourself running a new company with a viable but tight budget. You can afford one technical resource. With luck you find that one world class guru you can bank your whole business on and they have to do everything. One body responsible for ideation, architecture, engineering, software development, infrastructure, laptops, servers, software and today’s clouds, testing and quality assurance, vendor management, mixed platform integration and when you go into production with the app you’re hanging your shingle on that one body is going to be your 24/7 operations as well. Of course you have a great person who has access to everything, no limitations beyond budget and focus on what is most important to the product you’re making. And of course add in some really great processes and the best tools you can find them even if they write their own to fit technical and budgetary needs to keep your business running.

Now keep reading because you’re right: there is more to it.

For some time and for good reason, job specialization in roles such as Java Developer, .NET developer, Oracle DBA, Test Lead or Network Operations Engineer and the management of those roles with responsibilities to conform to standards and endure audits has produced a specific operating model. This model has role-based authorization and in many cases formal handoff of work product from stage to stage. It has a real side effect of creating roadblocks along the way in the form of checks and balances that are there to ensure quality, security, repeatability, durability and standard.

In the DevOps model, we strive to integrate those roles, thereby removing the use of silos. One means of achieving that integration is the agile team, which typically has seven to nine members, including coders, testers, operations and someone responsible for product. This team is given ownership and access to its resources and the responsibility to ensure their products work. In many cases this team follows their product into production and carries the pager answering calls 24/7. This means that they wake up at 3am and resolve issues they themselves created. This maintains the checks and balances while eliminating roadblocks and increasing aggregate productivity.

So we’ve covered that DevOps is in part the idea that we go back to the basic premise of an integrated DevOps of one. Therefore, we combine what is done in development and operations again as if our teams were small startup companies where both the work and the reward are shared.

The DevOps Scrum Team.
Most organizations will begin at and perhaps sustain a small DevOps team. In this case the Agile Model is very relevant to the discussion so we will start there. The agile Scrum Team is in best practice a small team of people with the skill sets needed to work within a process, enabled by a tool set to provide fast delivery of quality products. Normally this team will be seven to nine people. The exact size of the teams can vary as will the number of teams composed from a group of one hundred for example. The functions of architecting at this scale, developing, testing, ensuring quality, delivery and support within an approximate 2 week cycle exist in the team itself, meeting daily to address and remove roadblocks. The team works from a two tier backlog model long term and short term where priority dictates activity and capacity is constantly reviewed. The team is DevOps oriented, requiring a tool set nominally composed of a CI/CD framework including the development, test and delivery environments into one. This enables the developer to commit to trunk or branch to integrate code and deploy to test, thereby providing immediate results with any failures, and the opportunity to rework the components or when viable upgrade status of the components to the next phase, based on quality and dependability. Then when it moves along the cycle to releasing, staging and then production, everything has been tested including the automatic deployment process, ensuring that when it rolls into production it is a dependable, supportable product. This team is also mutually responsible for the support of everyone authorized to access and correct any problems in production, with a common commitment to answer at the moments of failure within a SLA. Operations tasks such as answering to a pager call at three in the morning in particularly with the smaller teams is also the responsibility of the DevOps team members. This eliminates a number of issues with traditional over the fence models where the team is very highly motivated to deliver dependable, supportable product. The customer feedback chain is much shorter, where the developer knows promptly when a business process or UI faults. That ensures a high level of responsiveness to customer needs increasing customer satisfaction, feature delivery and hence time to market.

As size grows we find multiple teams and some drift in models. Multiple scrum teams and integrations with teams operating in the more traditional waterfall model can be working together, scaling to sometimes hundreds of teams as required by the depth of an organizations’ portfolio. It is common to find an average of twenty-one to thirty-six people in a software organization. At some point the bulk of people will cause stress in managing the portfolio within a larger section of people. When you hit the one hundred to two hundred staff mark further focus on process needs to be applied. At this level we should consider using the same kinds of people, processes and tools but organized as independent sections able to maintain communication and work within closer physical proximity. Here the organization might also consider advancing from the standard Agile Model to the Scaled Agile for Enterprise model (SAFE.)

The DevOps of many.
The polar opposite of the one body IT startup is a multi-organization global project like the space program. Here the complexity is so extreme that compression becomes unrealistic. In this case ideation feeds from many people of various viewpoints, countries, governments, their science organizations, militaries and a daunting number of other named and un-named groups. All of the ideas, requirements, and government mandates etc. funnel through an overall Space Agency who, via funding of multiple bodies, manages and shares management of the grand operations required to architect, build, and run such a program. At this scale we face an aggregate IT organization with extended numbers in the hundreds of thousands. Architects exist in all areas, as do developers, testers and QA staff. We will see a prime contractor required to have one sub-contractor develop the software that a second sub-contractor tests, a third sub-contractor installs and operates, and a fourth sub-contractor provides Q&A and acceptance before its place in a flight ready platform is established and approved by the central space agency. This is daunting to say the least.

Here we can employ the same model as a standard with people, process and tools to best achieve the greater goals with an ability to integrate with flexibility.

Delivering DevOps of Business.

The Fortune 500 operate today at widely varying magnitudes of scale with technology and organizational footprints from small to very large. Firms must at times work together while maintaining internal independence and often need to operate within standards and with high levels of integration between their operations and those of other firms, including their competitors. For years IT organizations have deployed people, processes and tools to optimize how this work is done at these scales, and have done that effectively, delivering to standards which have evolved as measures of successful organizations (ITIL, PCI DSS, SOX, CMMI, ISO 9001, CISSP). DevOps delivers to those same standards.

Even so, the idea that cutting through the red tape can yield better and faster results repeatedly proves to be true in business as anywhere else. Specialization along division of work grows a healthy competition and yet a separation into silos that ensure stability, quality and durability at the
expense of good communication, alacrity, agility, and velocity of change which undermine innovation and restrict time to market. Independent silos also result in redundancy with multiple people on staff doing exactly the same jobs with multiple copies of the same kinds of infrastructure and tools, which increases cost.

**Delivering DevOps ROI.**

DevOps as it turns out, is the compression of all the complexity to reduce the red tape, to improve alacrity, agility and delivery quality of products and support services. This improves customer satisfaction at high velocity with better time to market and timely release of products to maintain and gain market share. Properly implemented, cost reduction and fast ROI can be achieved.

**Examples DevOps adoption delivering at scale.**

One example of where we see DevOps employed by a global community to deliver relevant, disruptive core products at very large scale is in open source and how it is being employed as an enabler to cloud product delivery. Keep in mind that open source software is by definition open which allows reuse and contribution to the code base by multiple voluntary entities. Although it is not always free, open source products are often sold as part of a common business model and can come with durability, dependability, comprehensive support, service level agreements (SLAs), and standards. Some very large scale IT industry leaders are open source contributors.

OpenStack is an example of this well-established model. It has a great number of contributing developers worldwide, and many enterprise software companies are jumping on board. VMWare integrated OpenStack into its product line, for example. HP also delivers its own distribution of OpenStack. RedHat includes OpenStack integration and management in its platform. There is a primary code trunk and multiple branch distributions where product specialization is done and code is contributed back to the community as open source. These specialized versions are used in some mission critical areas within the enterprise as is the truly open trunk. The processes to do this are common with standards that evolve from the community. Practices combining development and test, QA and operations via automated tooling for deployment and solid product management are jointly built, employed and refreshed. This model is so successful, many organizations large and small, embrace it completely.

**Review**

So we have covered the DevOps of One, of small teams, and DevOps of many. Therefore, we can draw some conclusions about DevOps:

- It delivers within industry standards (PCI, HIPPA, etc.)
- It enables faster time to market
- It can produce relevant ROI
- It is employed by both startups and industry leaders
- It has a broad community support
- It integrates well
Steps to build or transform to DevOps

Assess the current state of your organization and scale requirement.

It is important to know where you are to see where you need to go. Assessing where your organization is currently will normally involve a mini workshop to set expectations and initial interviews, depending on the size of your organization and how that effects scale.

Gap Analysis between current and desired

In a longer workshop involving interviews with management and subject matter experts within your firm, an analysis is used to assess where you are and what obstacles/milestones exist between this initial states and where you will be going.

Design the transformation in prioritized stages.

Given the information gathered in the workshop one result should be a project design to deliver your transformation in phases within a roadmap.

Project review and modification.

Once your plan is in place it is best practice to go back to the beginning and confirm your assumptions still apply. People, processes and tools change as will scope. With a review before you move on your project is more likely to succeed.

Commit and begin transformation.

Commitment from senior staff and executive level in your organization is critical at this phase. Once you have begun you will need to stay the course.
Celebrate Success.

Keep in mind that your entire organization gains from the benefit of a successful formation or transformation to a DevOps team model. Be proud of it and take credit with your people. It matters and it is fun.
What might the technical architecture or environments look like?

DevOps as we’ve covered is about people, process and tools. We have the people side of this down now. Here are a couple of images to highlight the process and tools.

First off we highlight the standard environments used in development and operations in an organization. Some firms don’t develop code but they do have a form of “crash and burn” similar to the traditional “dev” environment where things are “tried out.” A test environment where an organization has a fairly stable copy they “test” and qualify applications before moving them into the critical position normally called “production.”

The key difference in DevOps here is that the access, responsibility and tooling to run and use these environments is shared. Here again at the lower point in the diagram is the foundation of Cloud Management and Orchestration.

The architectures to achieve the orchestration of these virtual or logical environments will vary both by the technology chosen and as technology evolves. Below is a diagram showing a modern association of tooling functions to manage a larger scale enterprise cloud service which will support from dozens to thousands of servers and both internal and external use cases.
Generalized Cloud Architecture for DevOps

Service Portfolio Offering
- UI Handler
- API Handler
- Business UI
- DevOps UI
- other

API Aggregation Layer of branded and open APIs
- UI aggregation and routing
- Fulfillment
- Monitoring and Metrics
- Admin
- Security
- Business integration
- Audit
- Whole Scale Orchestration
- Resource Manager
- Cloud Service Broking
- Catalog driven aggregation
- Service Usage
- Forecasting
- Economic Controls
- Resiliency
- Sla based Run state
- Auto Recovery
- Customer, infrastructure, access accounting and operations
- Cache
- Cloud Data Records
- CMB
- Customer QB aggregation
- Endpoint management
- Hot and Cold Storage capacity

Logical Service Provider Resource API Bus – many aggregated APIs
- AWS
- Azure
- Rackspace
- AT&T
- Google
- Other Tools
- Other Vendors
- Back office as a Service
- DBaaS
- PaaS
- Other services
What might the SDLC workflow look like?

DevOps as we’ve covered is about people, process and tools. Part of the process is the usage of tools to manage work flowing from ideation to operation. People will ask if we really mean that one tool set will be used for the activities involved in all of the related stages for that to happen. The answer is conceptually yes. This is what that looks like if you can achieve it. One management platform which might include multiple components and capabilities as noted above orchestrates the resources from the raw development that happens at ideation (raw dev) to the first shelf in a repo, where automation also kicks off testing, QA, and testing against reference platforms, integration and into a staged or pre-production shelf before a certified version is promoted to production. Generally multiple versions will be flowing through this process at the same time. We commonly see things which are proposed and released in a limited manner, those things which are next to become the primary branded service and the current mainstream app in a portfolio.

External partners and customers also have similar states and if the ideals of common integration and APIs for endpoints exists issues may be contained to user acceptance from proposed, to next and current. The most stable of all configurations obviously should be current production.

From end to end a modern DevOps team will be using one fully functional tool set including continuous integration and delivery (deployment) known as CI/CD. Configurable automation of those tools allow for security and safety in all stages.

What issues still exist?

Must it be this way?

Does saying you have adopted DevOps or Agile Methodology mean you’re firm is actually operating the same or adhering to the practices or ideals in the same way internally as other organizations? No, each organization is subject to implementing based on its own understanding, capabilities, constraints and business objectives. Every organization is different.
Should they be the same? Not necessarily. Practical considerations should prevail and business objectives, constraints and organizational realities must be taken into account, rather than blindly jumping on the bandwagon. Evangelism of DevOps is valuable, but the cost of formation in or transformation to this model has to be weighed against other factors.

Standards and interoperability remain critical. As noted in the vision of “DevOps of many”, your organization’s success may depend on its existence in an ecosystem composed in part of other organizations with proprietary practices and code bases. This lack of visibility into means with which they operate can inhibit the integration of products. This maintains the need to operate as closely to a standard practice as possible to maintain interoperability at the integration points the products share. Commonly today DevOps best practices is to architecture applications with APIs at the integration end points. This practice is growing in enterprise products development.

The social sides of the equation.

Are Developers part of IT? Yes and No. This can be an emotional difference that has resulted from specialization and the “over the fence” model of responsibility, and security. Just sending people to classes or telling them they are now all the same is a little naïve and most business leaders won’t expect that to work immediately. Changing how people feel cannot be simply mandated. That takes time and an investment on the part of the business as well as the individuals. The outcome will vary as much as the community you’re working with, so electing to go down this path means you need to evaluate the impact of your staff’s willingness to change.

Is Operations part of IT? Yes and No. Many development groups do run their own operation and often don’t exist within an enterprise IT organization. Cool innovations of new apps like “Angry Birds” don’t tend to be developed in the IT Call Center doing tier 1 phone support or Network operations. Many IT staff have evolved a thick skin to endure the perception of being a necessary evil, a cost, or the police force of the organization. They also tend to have a bit of an ivory tower complex or at least can be perceived to be that way by other parts of an enterprise.

Program Management offices and IT Operations of Datacenters are groups that tend to hold a lot of power and may hold onto their domains to a degree that the “over the fence” model is not changed and DevOps never includes the application development parts of the organization. In this model the Datacenter operations teams may embrace a form of DevOps to build IT Operations tools (develop) and operate them. The benefits can be high but the overall enterprise may still remain fragmented and the “over the fence” trust model may still inhibit velocity, time to market, and innovation.

Improper application of the models may also exist. Development teams may use “agile” as a time and task tracking process only and bug tracking tools to document within waterfall style management. Some improvements may still be seen by converting multiple teams to newer tools, using common tools, and improving communication and those can have clear benefits. None the less highly controlled development environments can still inhibit velocity, time to market and innovation as much as never changing can.
What are some common practices in Development which exist also in Operations?

Both Development and Operations require a fairly similar flow where one may have a higher rate of iteration in some ways than the other.

Plan
Innovate, Create or implement
Test
Release
Review
Repeat

What are some common practices in Organizations?

An enterprise with both Development and Operations requires a means of operating the business itself. The success of DevOps has dependence on that organization.

- Advocate people (they are both your producer and your customer)
- Mentor people to improve their skills and sense of achievement
- Evangelize the processes, tools, and your product to show business commitment
- Market people, processes and tools in and out of the organization. People are examples we follow
- Coach people, teams, and the community. Elevate everyone.
- Train, invest and promote.

About Zefflin Systems

Zefflin is focused exclusively on Data Center Automation and Cloud Management solutions implementation and integration. As a world-class, agile, center of excellence, our aim is to work with best of breed software, combined with the industry's best technical consulting and integration talent. We provide consulting services in data center strategy, DevOps Transformation, DevOps automation, OpenStack consulting and software implementation. We cut through the hype, identifying which tools can be implemented and integrated to effectively automate application development and IT operations. We offer high quality, cost effective solutions addressing the automation of the entire lifecycle of complex computing environments, from request/catalog management, automated provisioning (OS, application, database, storage, network), to policy governance and compliance. Our vision is to bring to market consulting/software solutions that enable the lights-out data center. This will allow our customers to implement fully automated private, public and hybrid cloud systems, delivering low cost, high quality services to their customers while minimizing personnel cost. Our current software resale and implementation portfolio includes Scalr and CloudForms for cloud management and Cloudify for cloud orchestration, as well as support for all major OpenStack distributions. www.zefflin.com
Further resources


https://console.ng.bluemix.net/solutions/open-architecture


http://www.forbes.com/sites/patrickmoorhead/2014/05/12/intels-four-openstack-pillars-for-private-clouds/