DevOps for CIOs

A Ranger4 ThoughtPaper

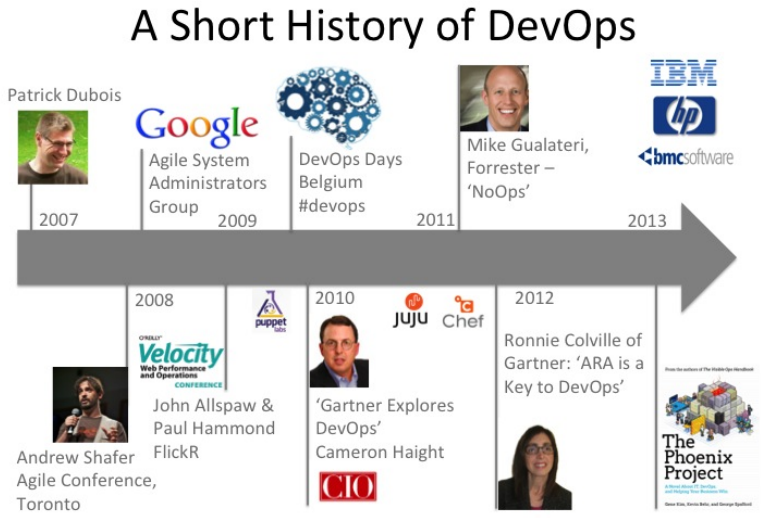
Contents

1.0 DevOps Today

DevOps has undoubtedly arrived - but is it on your CIO agenda yet? With topics such as Cloud, Big Data, Social, Mobile, Security and Risk Management clamouring for your attention when will DevOps get a look in? And how can DevOps help you with all of these other considerations?

There are some who claim DevOps is nothing new, that they have been doing it for years and others who dismiss it as a buzzword. But, at Ranger4, what we hear from our customers is that the increase in data and application complexity and businesses' need to innovate online is creating toxicity between IT development and operations teams - life can be unpleasant when releases are slow and unreliable, production systems are failing and people are missing their targets. And DevOps could well be the answer.

Whilst it's true that some larger enterprises, and technology companies like Google, Etsy and Facebook, have tried to tackle these issues for years or were born in a way that demanded they operate in a DevOps manner from their year dot, there are way more that have evolved in such a way that the noise, stress and burnout in their IT teams are reaching critical proportions.



The term 'DevOps' was first coined back in 2007 but it took a few more years for the analysts to pick up on the movement, during which time several disruptive technologies broke into the market (Puppet, JuJu, Chef amongst others) and then, finally, the big vendors (IBM, BMC, HP) hopped on the bandwagon.

These days it seems everyone's talking about it - PriceWaterhouseCoopers featured DevOps for the CIO heavily in an issue of their Technology Forecast in 2013 with the words:

"Speed versus stability is a paradox for IT. Pushing for both at the same time seems contradictory, but the IT organization must strive to achieve this goal if the enterprise is

ultimately to become more antifragile - that is, less vulnerable to catastrophe."

2013 also saw the release of The Phoenix Project - a novel about DevOps in which Gene Kim, Kevin Behr and George Spafford explore the 3 ways (more on these later) - a stellar success and a must-read for anyone who wants to understand more about the challenges DevOps solves - and how.

There are even some people who think DevOps is already over, already burned out or at least, but from what we see and hear, we've barely started and 2014 promises to be the year of DevOps.

1.1 Benefits of DevOps

- Accelerate innovation time to market
- Compress testing time
- Reduce time spent troubleshooting
- Accelerate release cycles, release more frequently
- Improve application quality and performance
- Reduce downtime and unplanned outages

2.0 Baselining DevOps

We know that it's your job as CIO to have a tight handle on the costs of information technology in your business - and to ensure that investments have the levels of return you expect. So before you embark on a DevOps program it's essential that you measure the baseline of where you are at today.

Having a baseline means you can build a business case, apply targets and goals to your projects and measure your success as you progress through your project ultimately reporting back to the board on how you used the money to save or make more money - and improved your teams' satisfaction. Happy people are the most productive people.

2.1 Technical and Financial Metrics

There are hard, quantifiable technical and financial metrics we can take, such as:

- Number and frequency of software releases
- Volume of defects
- Time/cost per release
- MTTR*
- Number and frequency of outages / performance issues
- Revenue/profit impact of outages / performance issues
- Number and cost of resources**

* The MTTR is the Mean Time to Repair, Resolve or Resolution - each of the definitions means the same and can be used interchangeably. This term is more commonly used when talking about Application Performance Management and the speed at which an outage or performance issue can be fixed, but equally can be used when talking about testing and eliminating defects

** It's worth noting that one of the biggest inhibitors to success of DevOps and relating tooling projects is people's perceptions that they are at risk of losing their jobs as their work becomes automated. Often, particularly in areas like release and deployment management, we find that there are specific individuals who hold all the knowledge around a current process (they wrote all the scripts for example) and who are viewed as heroes when they are the only person who can fix an issue and often do it out of hours and at short notice - but are, in fact, bottlenecks. These individuals are often highly talented, but feel secure in the indispensable role they have created for themselves. Though they will often be happier freed up to do more creative and rewarding work, they are often fearful and this needs to be addressed.

2.2 Cultural Metrics

Although cultural metrics are softer, nigh on impossible to apply hard dollar value to, DevOps is about resolving conflict in the workplace, eliminating stress and avoiding burnout - and they are measurable. Happy people are more productive - their health is better, they have more ideas, work more effectively and will put in the extra mile. We constantly review our customers across a number of key cultural indicators around feelings about change, failure, going to work, what a typical day's work entails, in addition to a number of cultural attributes such as:

- Cross-skilling, knowledge sharing and pairing between teams
- Working in a fluid but focussed manner
- Working in multi-disciplinary teams
- Organizing teams around projects rather than skill-sets
- Constantly dancing on the edge of failure (in a good way)
- Position vis a vis business demand
- Extraneous lines of code
- Attitude to continuous improvement
- Obsession with metrics
- Technological experimentation
- Team autonomy

We also look at a number of team features such as:

- Rewards and feelings of success
- Hierarchical and political obstacles and annoyances
- Inspiring and fostering creativity

2.3 Process Metrics

DevOps is not a process or a tool - but there are a number of processes in the software development lifecycle (SDLC) that affect both traditional development and operations staff to greater or lesser degrees that need to be taken into consideration. All of these process components can be optimised, and all of them can then be improved upon further using appropriate software tooling. An ultimate goal of a typical DevOps project is often to attain true continuous delivery (CD) by linking these processes and tools together to allow fully tested, production ready, committed code to proceed to live without impediment - we often refer to the software infrastructure piece of this as the DevOps toolchain. When baselining current state, it's useful to measure these component processes and their relative maturity (taking into account use of existing tools and success of implementation). Typically, we look at:

- Requirements elicitation and management
- Agile development
- Build
- Release and deployment
- Unit testing
- User Acceptance Testing
- Quality Assurance
- Application Performance Monitory
- Cloud

3.0 The Three Ways

Gene Kim and his team assert that the Three Ways describe the values and philosophies that frame the processes, procedures, practices of DevOps, as well as the prescriptive steps. You can read more about the three ways in Kim's seminal work The Phoenix Project, co-authored by Kevin Behr and George Spafford.

3.1 The First Way: Systems Thinking

The First Way emphasizes the performance of the entire system, as opposed to the performance of a specific silo of work or department. This as can be as large as a division (e.g. Development or IT Operations) or as small as an individual contributor (e.g. a developer or system administrator).

The focus is on all business value streams that are enabled by IT. It starts at requirements elicitation and gathering, are built in Development, and then transitioned into IT Operations - there the value is delivered to the customer as a service.

The outcomes of putting the First Way into practice include:
- Never passing a known defect downstream
- Never allowing local optimization to create global degradation
- Always seeking to increase flow
- Always seeking to achieve profound understanding of the system

3.2 The Second Way: Amplify Feedback Loops

The Second Way is about creating the right to left feedback loops. The goal of almost any process improvement initiative is to shorten and amplify feedback loops so necessary corrections can be continually made.

The outcomes of the Second Way include:
- Understanding and responding to all customers (internal and external)
- Shortening and amplifying all feedback loops
- Embedding knowledge where we need it

3.3 The Third Way: Culture of Continual Experimentation and Learning

The Third Way is about creating a culture that fosters two things:
1) Continual experimentation, taking risks and learning from failure

2) Understanding that repetition and practice are the prerequisites to mastery

Both of these are needed equally. Experimentation and risk-taking ensures we continually strive to improve, even if it means going deeper into the danger zone ever before. And we need mastery of the skills that can help us retreat out of the danger zone when we've gone too far (this is often referred to as 'smart-failure').

The outcomes of the Third Way include:
- Allocating time for the improvement of daily work
- Creating rituals that reward the team for taking risks
- Introducing faults into the system to increase resilience

We use the three ways when looking at process improvement in DevOps driven projects.

4.0 DevOps, ITIL and Agile

Debate continues to rage about where DevOps, ITIL and Agile meet, crossover and conflict, with even some outlandish rumours that DevOps means ITIL is obsolete. It's not. And neither is Agile. Agile is a development methodology designed to resolve the time and budget issues caused by rigid, lengthy waterfall development processes. ITIL is a set of practices (processes, procedures, tasks and checklists) that are not organization-specific, used by an organization for establishing integration with the organization's strategy, delivering value and maintaining a minimum level of competency. It allows the organization to establish a baseline from which it can plan, implement and measure. It is used to demonstrate compliance and to measure improvement - and it has succeeded in taking a lot of chaos out of organization's IT efforts but sometimes it is felt it takes formality and rigidity too far. Application of Agile and DevOps principles over the foundation of ITIL can help overcome this.

DevOps, Agile and ITIL all work together harmoniously.

4.1 ITIL

You're probably already running ITIL processes so it's a good place to start - DevOps further enhances and improves these processes, for example:

4.1.1 Change Management

DevOps is all about change. It's about making change happen better. ITIL does bring a degree of control over the change management process, but all too often it creates bottlenecks along the way with stakeholders blocking changes and changes causing errors as a result of insufficient testing.

If you've done everything ITIL tells you to do (you've got the approval process, prioritizations, stakeholders in place) but change still isn't happening fast enough, you might find that you have lost some of the human touch - people don't understand the changes they are being asked to approve - they don't trust them and aren't bought into them.

DevOps is about collaborating and sharing information - and shifting left. That means doing everything earlier (testing for example) and involving those later down the line in the process earlier so they can see what's coming, make recommendations and plan better for each and every release.

4.1.2 Release and Deployment Management

One of the key complaints of development teams is the time it takes for environments to be delivered (both pre-production and live) for their new and updated applications. One of the key complaints of operations is why the software produced in development doesn't work properly in the staging environments and through the route to live. Operations teams are also often extremely busy with critical outages and a source of conflict is development wanting a speedy release and the operations guys struggling to fit it into their workplan.

As with change management, involving the operations teams earlier helps them schedule the preparation of environments and release windows, keeping development happening. Additionally, this is a key area where automation delivers huge time and cost benefits (most enterprises still perform releases manually or using complex, error-prone scripts today).

Please see the Ranger4 Guide to Application Release Automation for more information.

4.1.3 Testing

Testing today is often far more time consuming and expensive than it needs to be and still not as effective as most organisations hope. By shifting testing left, performing it earlier in the application release cycle, the defect incidence drops rapidly. Additionally, use of service virtualization tools makes it possible to spin up replicas of the applications quickly and easily and test the business flows at high speed.

Please see the Ranger4 Guide to Integration Testing & Service Virtualization for more information.

4.1.4 Incident Management

Things do go wrong. Outages happen. Applications' performance is impacted. Traditionally, this has resulted in the setting up of war rooms and a lot of finger pointing between the developers, system administrators, DBAs, security and networking teams. A central tenet of DevOps is that blame has only a negative impact and must be avoided. In order to innovate fast, people must be allowed to fail. Systems must be in place that allow reversion to a last known good release at the click of a button (see section 4.1.2) and diagnostics must be implemented to allow the shortest possible MTTR.

Of course, getting change, release and testing processes right will mean a huge reduction in system failure too.

Please see the Ranger4 Guide to Application Performance Management for more information.

4.2 Agile

Three of the 12 principles in the Agile Manifesto emphasize software delivery practices:

- "Our highest priority is to satisfy the customer through early and continuous delivery of valuable software."
- "Deliver working software frequently, from a couple of weeks to a couple of months, with a preference to the shorter timescale."
- "Working software is the primary measure of progress."

The manifesto was written over a decade ago, and most organizations are still working to become Agile. However, there are some organizations who are exceeding these principles - their software is always ready to release, and there are no walls or silos around the teams: they have one delivery team - consisting of experts in development, sys admin, security, testing, database, operations, etc continually delivering software to their customers. This is true agility.

The key driver behind the DevOps movement is the pain caused by the conflict involved in releasing and maintaining software applications across teams. On typical software projects, two teams are responsible for developing and delivering software: development and operations. The two teams intend to collaborate, but their goals are in conflict. Development are driven to deliver innovation, where operations, are primarily concerned with stability. Delivering innovation demands change and change puts stability at risk - code defects, performance issues, system outages, release failures - all disasters that put the teams at loggerheads.

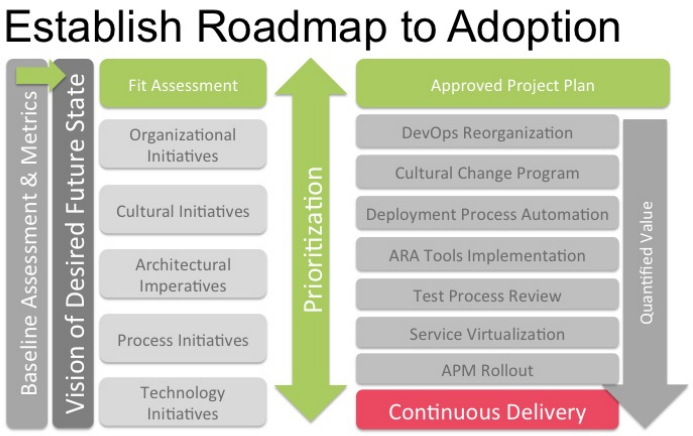
The core goal of the DevOps movement is for developers and operations people to work together effectively and collaboratively. As a result, a new breed of DevOps engineers is emerging who take the best of both disciplines and combine them to deliver value to users. This is also manifesting in the rise of cross-functional teams that are experienced in development, configuration management, database, testing, and infrastructure.

Where Agile was all about developing software faster, DevOps is about integrating Ops into the process - a superset of principles for rapid software and development and delivery, if you like.

5.0 Assessing DevOps Maturity and Readiness

In section 2 we described the importance of baselining your current culture, processes and environment in order to build your investment case and to apply measurable goals to your project and report to the board on quantifiable outcomes.

It's also useful though, to know where your business is on a scale compared to the rest of the market, or in your particular industry - and to have an understanding of just how ready you are for DevOps and how much change you have ahead of you.



Ranger4 have developed the DMI - the DevOps Maturity Index, which allows us to apply a number between one and one hundred (one hundred being 'fully DevOps') describing where you are on the DevOps maturity scale. This helps you map your current position, set a goal for a specific time in the future and measure yourself against it. The DMI takes into account, people, culture, process and tooling attributes.

It's tempting to go out and buy a DevOps expert or a DevOps tool, but as we'll see in the next section, that's not always the best approach. When looking at a DevOps delivery program, it's best to consider all of the options available and perform a fit assessment. Sometimes there are quick wins, but sometimes it's the tougher, longer projects that will deliver much more benefit to your business.

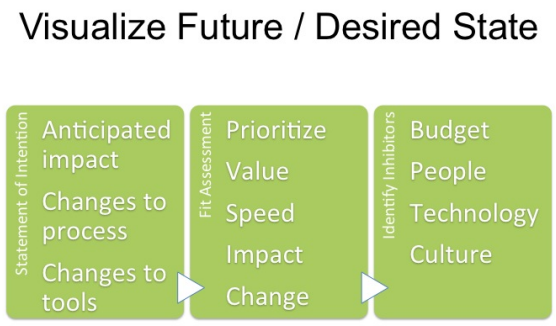
6.0 Delivering DevOps

There is a great deal of material available debating the nature of DevOps, but we'll summarise what DevOps is not:

- DevOps is not a job title
- It's not a person or a team
- It's not a process
- It's not a tool and it isn't continuous delivery - although CD is often a DevOps goal

DevOps IS a movement towards a cultural change where:

- Collaboration is king - proactive involvement of all stakeholders from project conception (shift left)
- Failure is embraced as essential to improvement (and conducted smartly)
- Blame does not exist
- A united focus is placed on taking innovation to market at optimum speed
- Everyone is involved - not just development and operations but all of your IT staff, the architects, security, QA and testing guys and gals too - and (gasp) also the business



So you can't just 'buy' DevOps - it requires all organisations, especially long established enterprises, to take a view and mandate from the top, find enthusiastic key stakeholders, enlist support at all levels and develop and implement a long-term program of change. There will be pain, there will likely be resistance from some quarters, but as described earlier in the paper, there will be very significant and measurable benefits both financially and culturally.

So what does a successful DevOps operation look like? Here are The Seven Habits of Highly Effective DevOps as identified by Forrester Research (See section 6 for a link to the full text):

1. Getting the two sides to talk to each other
2. Taking an outside-in approach to everything
3. Automating the build, test and release processes so they contain less human error
4. Simplifying and standardizing the development and production environments
5. Instilling a culture of systems engineering across both development and operations

6. Implementing feedback and feed-forward loops
7. Putting developers on the front line of support

It may be tempting to think that a tool will offer immediate comfort to an obvious software development pain - for example, if an organisation's software is reported to be released with an unacceptably high proportion of defects, the immediate reaction may be to focus on automated the testing process. However, by taking a step back, and looking at the whole DevOps environment and culture, it might be found that defects could be eliminated earlier on in the process, through closer collaboration between the development and operations teams as the software requirements are specified and the whole chain prepared for the eventual release. Equally, there are some tools that will help baseline the current situation, such as Application Performance Management, and thereby support a business case that will offer some immediate relief, by helping faster discovery of issues, but the business will still only fully benefit from a DevOps cultural shift when all the processes and components of the DevOps toolchain have been optimised.

As CIO, you are ideally placed to lead your DevOps revolution. Successful DevOps requires advocates empowered to effect change. So where do you start?

We have developed the Ranger4 DevOps Maturity Assessment as an ideal place for enterprise CIO's to start. The assessment is a 4 - 10 day consulting engagement that:

- Gives you baseline measurements including a DMI rating for your position today
- Documents your desired future state
- Delivers a roadmap to your desired future state including fit-assessed DevOps program components

To request a DevOps Maturity Assessment please go to www.ranger4.com.

7.0 Additional Resources and Reading

The Seven Habits of Highly Effective DevOps from Forrester Research

The Phoenix Project by Gene Kim, Kevin Behr and George Spafford

DevOps Matters on LinkedIn

The DevOps Guys blog

DevOps Days

DevOps People to Follow (via PuppetLabs)

CIOs Can Accelerate Innovation with DevOps (CXO Today)

IT Revolution

DevOps Angle