

the agile admin

What Is DevOps?

DevOps is a term for a group of concepts that, while not all new, have catalyzed into a movement and are rapidly spreading throughout the technical community. Like any new and popular term, people have somewhat confused and sometimes contradictory impressions of what it is. Here's my take on how DevOps can be usefully defined; I propose this definition as a standard framework to more clearly discuss the various issues DevOps covers. Like "Quality" or "Agile," DevOps is a large enough concept that it requires some nuance to fully understand.

Definition of DevOps

DevOps is a new term emerging from the collision of two major related trends. The first was also called "agile system administration" or "agile operations"; it sprang from applying newer Agile and Lean approaches to operations work. The second is a much expanded understanding of the value of collaboration between development and operations staff throughout all stages of the development lifecycle when creating and operating a service, and how important operations has become in our increasingly service-oriented world (cf. [Operations: The New Secret Sauce](#)).

One definition Jez Humble explained to me is that DevOps is "a cross-disciplinary community of practice dedicated to the study of building, evolving and operating rapidly-changing resilient systems at scale."

That's good and meaty, but it may be a little too esoteric and specific to Internet startup types. I believe that you can define DevOps more practically as

DevOps is the practice of operations and development engineers participating together in the entire service lifecycle, from design through the development process to production support.

A primary corollary to this is that part of the major change in practice from previous methods is

DevOps is also characterized by operations staff making use many of the same techniques as developers for their systems work.

Those techniques can range from using source control to testing to participating in an Agile development process.

For this purpose, "DevOps" doesn't differentiate between different sysadmin sub-disciplines – "Ops" is a blanket term for systems engineers, system administrators, operations staff, release engineers, DBAs, network engineers, security professionals, and various other subdisciplines and job titles. "Dev" is used as shorthand for developers in particular, but really in practice it is even wider and means "all the people involved in developing the product," which can include Product, QA, and other kinds of disciplines.

 Follow

DevOps has strong affinities with Agile and Lean approaches. The old view of operations leaned towards the "Dev" side

being the “makers” and the “Ops” side being the “p the harm that has been done in the industry of those DevOps. In this way, DevOps can be implemented through close collaboration of customers, product management iterate towards a better product – DevOps says “yes fundamental part of the value proposition to the client as a top level item.” From this perspective, DevOps “code” to the entire delivered service.

Follow “the agile admin”

Get every new post delivered to your Inbox.

Join 2,412 other followers

Build a website with WordPress.com

” – the realization of : core driver behind development prescribes in the gaps and rapidly systems interact are a) include those concerns l the boundaries of “the

Definition In Depth

DevOps means a lot of different things to different people talk about DevOps being “developer and operations “using automation,” or “using kanban,” or “a toolcha items. The best way to define it in depth is to use a parallel method to the definition of a similarly complex term, agile development. Agile development, according to [Wikipedia](#) and the [agile manifesto](#), consists of four different “levels” of thing. I’ve added a fifth, the tooling level – talk about agile and devops can get way too obsessed with tools, but pretending they don’t exist is also unhelpful.

ers a lot of ground. People “infrastructure,” or it’s emingly loosely related

- **Agile Values** – Top level philosophy, usually agreed to be embodied in the [Agile Manifesto](#). These are the core values that inform agile.
- **Agile Principles** – Generally agreed upon strategic approaches that support these values. The Agile Manifesto cites [a dozen of these more specific principles](#). You don’t have to buy into all of them to be Agile, but if you don’t subscribe to many of them, you’re probably doing something else.
- **Agile Methods** – More specific process implementations of the principles. XP, Scrum, your own homebrew process – this is where the philosophy gives way to operational playbooks of “how we intend to do this in real life.” None of them are mandatory, just possible implementations.
- **Agile Practices** – highly specific tactical techniques that tend to be used in conjunction with agile implementations. None are required to be agile but many agile implementations have seen value from adopting them. Standups, planning poker, backlogs, CI, all the specific artifacts a developer uses to perform their work.
- **Agile Tools** – Specific technical implementations of these practices used by teams to facilitate doing their work according to these methods. JIRA Agile (aka Greenhopper), planningpoker.com, et al.

Ideally the higher levels inform the lower levels – people or organizations that pick up specific tools and practices without understanding the fundamentals may or may not see benefits but this “cargo cult” approach is generally considered to have suboptimal results. I believe the different parts of DevOps that people are talking about map directly to these same levels.

- **DevOps Values** – I believe the fundamental DevOps values are effectively captured in the Agile Manifesto – with perhaps one slight emendation to focus on the overall service instead of simply “working software.” Some previous definitions of DevOps, like Alex Honor’s “[People over Process over Tools](#),” echo basic Agile Manifesto statements and urge dev+ops collaboration.
- **DevOps Principles** – There is not a single agreed upon list, but there are several widely accepted attempts – [here’s John Willis coining “CAMS”](#) and [here’s James Turnbull giving his own definition](#) at this level. “Infrastructure as code” is a commonly cited DevOps principle. I’ve made [a cut at “DevOps’ing” the existing Agile](#)

[manifesto and principles here](#). I personally believe that DevOps at the conceptual level is mainly just the widening of Agile's principles to include systems and operations instead of stopping its concerns at code checkin.

- **DevOps Methods** – Some of the methods here are the same; you can use Scrum with operations, Kanban with operations, etc. (although usually with more focus on integrating ops with dev, QA, and product in the product teams). There are some more distinct ones, like [Visible Ops](#)-style change control and [using the Incident Command System for incident reponse](#). The set of these methodologies are growing; a more thoughtful approach to monitoring is a hot topic right now.
- **DevOps Practices** – Specific techniques used as part of implementing the above concepts and processes. Continuous integration and continuous deployment, “Give your developers a pager and put them on call,” using configuration management, metrics and monitoring schemes, a toolchain approach to tooling... Even using virtualization and cloud computing is a common practice used to accelerate change in the modern infrastructure world.
- **DevOps Tools** – Tools you'd use in the commission of these principles. In the DevOps world there's been an explosion of tools in release (jenkins, travis, teamcity), configuration management (puppet, chef, ansible, cfengine), orchestration (zookeeper, noah, mesos), monitoring, virtualization and containerization (AWS, OpenStack, vagrant, docker) and many more. While, as with Agile, it's incorrect to say a tool is “a DevOps tool” in the sense that it will magically bring you DevOps, there are certainly specific tools being developed with the express goal of facilitating the above principles, methods, and practices, and a holistic understanding of DevOps should incorporate this layer.

In the end, DevOps is a little tricky to define, just like its older brother Agile. But it's worth doing. When left at the pure philosophy level, both can seem like empty mom-and-apple-pie statements, subject to the criticism “You're just telling me ‘do my job better,’ duh...” But conversely, just the practices without the higher level guidance turn into a cargo cult. “I do what this Scrum book says so I'm doing Agile” is like “I'm using Chef so I'm DevOps right?” To be a successful Agile or DevOps practitioner is to understand all the layers that explain what it is, what it might be, and what a given implementation might contain or not contain. In the end, what DevOps hopes to bring to Agile is the understanding and practice that software isn't done until it's successfully delivered to a user and meets their expectations around availability, performance, and pace of change.

History of DevOps

The genesis of DevOps comes from an increasing need for innovation on the systems side of technology work. The DevOps movement inherits from the Agile System Administration movement and the Enterprise Systems Management (ESM) movement.

ESM, which arose in the mid-2000's, provided the original impetus of “Hey, our methodology of running systems seems to still be in a pretty primitive state despite years of effort. Let's start talking about doing it better.” John Willis, whurley, and Mark Hinkle from Zenoss were involved in that, and sponsored a [BarCamp around the concept](#). I think during this phase, initial enchantment with ITIL as a governance framework was largely overthrown for the “ITIL Lite” Visible Ops approach, as well as a shift from being “large vendor” focused – used to be, the enterprise frameworks like HP, IBM, and CA were the only meaningful solutions to end to end systems management, but more open source and smaller vendor stuff was coming out, including Spiceworks, Hyperic, Zenoss, and others.

Also in 2008, the first [Velocity conference](#) was held by O'Reilly, focusing on Web performance and operations, which provided a venue for information sharing around operations best practices. In 2009 there were some important

presentations about the developer/operations collaboration at large shops (most notably [Flickr](#)) and how that promoted safe, rapid change in Web environments. Provisioning tools like Puppet and Chef had strong showings there. More people began to think about these newer concepts and wonder how they might implement them.

Somewhat in parallel, as agile development's growth in the development space was reaching its most fevered pitch and moving from niche to common practice, this turned into thinking about "Agile Systems Administration" especially in Europe. Gordon Banner of the UK talked about it early on with [this presentation](#). A lot of the focus of this movement was on process and the analogies from kanban and lean manufacturing processes to IT systems administration. Then sometime in 2009, Patrick Debois from Belgium and Andrew "Clay" Shafer from the US met and [started talking up \(and coined the term\) DevOps](#), and then Patrick held the first DevOpsDays event in Ghent that lit the fuse. The concept, now that it had a name, started to be talked up more in other venues (I found out about it at [OpsCamp Austin](#)) including Velocity and DevOpsDays here in the US and spread quickly.

[In Patrick Debois' view](#), DevOps arose as a reaction against the silos and inflexibility that were resulting from existing practices, which probably sounds familiar. [Here's a good piece](#) by John Willis on the history of the DevOps movement that deconstructs the threads that came together to create it.

DevOps emerged from a "perfect storm" of these things coming together. The growing automation and toolchain approach fed by more good monitoring and provisioning tools, the need for agile processes and dev/ops collaboration along with the failure of big/heavy implementations of ITSM/ITIL – they collided and unconsciously brought together all three layers of what you need for the agile movement (principles, process, and practices) and caught fire. Since then it has developed, most notably by the inclusion of Lean principles by many of the thought leaders.

What is DevOps Not?

It's Not NoOps

It is not "they're taking our jobs!" Some folks think that DevOps means that developers are taking over operations and doing it themselves. Part of that is true and part of it isn't.

It's a misconception that DevOps is coming from the development side of the house to wipe out operations – DevOps, and its antecedents in agile operations, are being initiated out of operations teams more often than not. This is because operations folks (and I speak for myself here as well) have realized that our existing principles, processes, and practices have not kept pace with what's needed for success. As businesses and development teams need more agility as the business climate becomes more fast paced, we've often been providing less, and we need a fundamental reorientation to be able to provide systems infrastructure in an effective manner.

Now, as we realize some parts of operations need to be automated, that means that either we ops people do some automation development, or developers are writing "operations" code, or both. That is scary to some but is part of the value of the overall collaborative approach. All the successful teams I've run using this approach have both people with deep dev skill sets *and* deep ops skill sets working together to create a better overall product. And I have yet to see anyone automate themselves out of a job – as lower level concerns become more automated, technically skilled staff start solving the higher value problems up one level.

It's Not (Just) Tools

It's also not "about the tools." One reason why I want to have a more common definition of DevOps is that the risk of

having various confusing and poorly structured definitions increases the risk that people will pass by the “theory” and implement the processes or tools of DevOps without the principles in mind, which is definitely an antipattern.

Agile practitioners would tell you that just starting to work in iterations without initiating meaningful collaboration is likely to not work out real well. There are some teams at companies I’ve worked for that adopted some of the methods and/or tools of agile but not its principles, and the results were suboptimal. Sure, a tool can be useful in Agile (or DevOps), but if you don’t know how to use it then it’s like giving an assault weapon to an untrained person.

But in the end, fretting about “tools shouldn’t be called DevOps” is misplaced. Is poker planning “agile” in the sense that doing it magically gets you Agile? No. But it is a common tool used in various agile methodologies, so calling it an “agile tool” is appropriate. Similarly, just because DevOps is not just a sum of the tools doesn’t mean that tools specifically designed to run systems in accordance with a DevOps mindset aren’t valuable. (There are certainly a bunch of tools that seem specifically designed to prevent it!)

It’s Not (Just) Culture

Many people insist that DevOps “is just culture” and you can’t apply the word to a given principle or practice, but I feel like this is overblown and incorrect. Agile has not helped thousands of dev shops because the work on it stopped at “culture,” with admonitions to hug coworkers and the lead practitioners that identified the best practices simply declaring it was all self-evident and refusing to be any more prescriptive. (Though there is some of that). DevOps consists of items at all the levels I list above, and is largely useless without the tangible body of practice that has emerged around it.

It’s Not (Just) Devs and Ops

And in the end, it’s not exclusionary. Some people have complained “What about security people! And network admins! Why leave us out!?!” The point is that all the participants in creating a product or system should collaborate from the beginning – business folks of various stripes, developers of various stripes, and operations folks of various stripes, and all this includes security, network, and whoever else. There’s a lot of different kinds of business and developer stakeholders as well; just because everyone doesn’t get a specific call-out (“Don’t forget the icon designers!”) doesn’t mean that they aren’t included. The original agile development guys were mostly thinking about “biz + dev” collaboration, and DevOps is pointing out “dev + ops” collaboration, but the mature result of all this is “everyone collaborating”. In that sense, DevOps is just a major step for one discipline to join in on the overall culture of agile collaboration that should involve all disciplines in an organization.

It’s Not (Just) A Job Title

Simply taking an existing ops team and calling them “The DevOps Team” doesn’t actually help anything by itself. Nor does changing a job title to “DevOps Engineer.” If you don’t adopt the values and principles above, which require change at an overall system level not simply within a given team, you won’t get all the benefits.

However, I’m not in the camp that rails that you ‘can’t have DevOps in a job title.’ It is often used in a job title as a way to distinguish “new style DevOps-thinking, automation-first, dev-collaborating, CI-running, etc. sysadmin” from “grouchy back room person who aggressively doesn’t care what your company does for a living.” Some people find value in that, others don’t, and that’s fine.

It’s Not Everything

Sometimes, DevOps people get carried away and make grandiose claims that DevOps is about “everything everywhere!” Since DevOps plugs into the overall structure of a lot of lean and agile thinking, and there are opportunities for that kind of collaboration throughout an organization, it’s nice to see all the parallels, but going and reengineering your business processes isn’t really DevOps per se. It is part of an overall, hopefully collaborative and agile corporate culture, but DevOps is specifically about how operations plugs into that. Some folks overreach and end up turning DevOps into a super watered down version of Lean, Agile, or just love for everyone. Which is great at the vision level, but as you march down the hierarchy of granularity, you end up mostly dealing with operational integration – other efforts are worrying about the other parts (you can personally too of course).

Share this:

100 responses to “*What Is DevOps?*”

Pingback: [What’s a “DevOp?” « the agile admin](#)

Pingback: [Intro to DevOps + what we learned at DevOpsDays Boston 2011 | ZeroTurnaround.com](#)

Pingback: [Intro to DevOps + what we learned at DevOpsDays Boston 2011 | Agile Development](#)

Pingback: [Ubuntu 11.10 released, DevOps now distilled \(and defined\) | Journal of Technology and Economic Development | Future Technology | Green Technology | Military Technology | Business | Trading | Finance | Computer | Robots | Entertainment | Games | GPS | Soft](#)

Carlos Gomez

December 7, 2011 at 1:23 pm



I suggest the term ” DevOps Engineer” to clarify that it is not a Developer or IT operation person, it is a complete new role.

[Reply](#)

ernestm

December 7, 2011 at 5:31 pm



I don’t believe DevOps is a new role or job title. It is a way of working. It is people with both developer and operations skill sets (one, the other, or a mix) working together on product teams to create products.

In a larger org, you may have some operations folks that embed into product teams and others that don’t directly. You may have developers working on system provisioning tools, release automation, or monitoring and testing

frameworks. Those may have specific role names, but which one is “the” DevOp? None of them, for that’s not a job title.

When agile came, you didn’t start calling developers AgileDevs. If a dev does test driven development, they’re not a DevTester. When a QA person works closely with developers to automate their testing, they’re not a DevQA. At best, agile is an adjective or something put in a job description to indicate “we want people used to collaborating in this way” – but you don’t hire an “Agile.” Same with DevOps. There are DevOps devs and DevOps ops and DevOps who have both skill sets and do some of both.

Reply

Bill Higgins

June 8, 2014 at 7:55 am



I agree with your point of view on the idea of “DevOps engineer” as a new, different role. I think it’s harmful for the reasons you cite above and the reasons Jez sited in his “No such thing as a DevOps team” article. However, I keep seeing people new to DevOps immediately adopt that anti-pattern, so I’ve been trying to think of a good, simple analogy to explain why it’s a bad idea.

The best I can come up with so far is two people dancing. Originally they danced awkwardly together, partially because they didn’t really like each other, but partially because they didn’t understand the other’s rhythm. Now, they’ve changed and they’re starting to figure out how to dance together better, and they’re starting to understand and even practice the other’s rhythm. The last thing you want to do is introduce a third party who acts as the broker between the two dance partners! It’s actually even more awkward!

Can you think of a better analogy or can you improve upon this one? I think it would help make this subtle concept more obvious.

kevin c

February 17, 2012 at 10:33 am



This is perfect; I’m the Scrum Master for an organization that is upping it’s agile implementation substantially. As the dev teams have ramped up the gap with OPs has become very clear, and we’re looking at how, with the people we have, to bring the OPs folks more closely into the teams as we already have with UX and QA.

Your articles, and specifically your comment above very clearly outline not only the “why” though also offer very clear ideas on the “how”.

I’ll definitely be forwarding your site on to both the product owner and VP of engineering.

Thank you again for putting the time into this valuable topic!

Reply

Pingback: [G's view of the world - On dev-ops, marketing, the c-word, and pneumonia](#)

Ray Scott

February 28, 2012 at 11:35 am



I do not often subscribe to BLOGS as my thoughts on Agile often conflict with many of the traditional mindset Agile-ists. I found your posting on DevOps enlightening and look forward to both reading more of your postings and adding my own comments also.

I come from a rare area for Agile advocacy, that of Quality, please note I did not say QA or testing. Not that there is anything negative about QA or testing, however, after 15+ years living in this area and 6+ years offering my ideas supporting the Agile Principles to my senior leadership, it has become blatantly obvious that divides still exist. I have managed many QA teams on large and small projects and influence upwards with the goal to increase Quality. Until now I had never heard of the role DevOp and now I feel much more content knowing that I am a DevOp proponent.

I also recently returned to the UK after 21 years in the USA working as a Developer, Test Manager, Scrum Master and Agile coach and I have easily seen much more eagerness to move towards Agile principles than I ever saw in the USA...why? I do not know yet, but will drill down to find out.

Cheers

Ray Scott

[Reply](#)

ernestm

March 8, 2012 at 9:51 pm



Glad it's resonating with y'all! I first heard the term at Opscamp 2009, and I realized "So THAT'S what I've been trying to get at..."

[Reply](#)

Pingback: [Windows Azure and Cloud Computing Posts for 4/2/2012+ - Windows Azure Blog](#)

Pingback: [DevOps | andrewleaning.com](#)

John Graber (@JohnGraber)

August 14, 2012 at 11:00 am



OUTSTANDING. I'm a sysadmin and dba (with a strong codemonkey streak) that is just now becoming aware of DevOps. Your post was extremely helpful in making the concept clearer to me.

[Reply](#)

Pingback: [Agile and DevOps: A Perfect Match](#)

Pingback: [Devops Reading List | IT Revolution](#)

Joe

March 18, 2013 at 4:22 pm

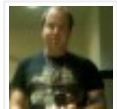


DevOps is slowly catching on but I've noticed a struggle over control that seems to hinder it's success. It's not cut and dry, there are [shades of DevOps](#) while Dev and IT cultures slowly merge onto the same page. There's definitely a lot of baggage (processes & people) to overcome...

[Reply](#)

Phill Hocking (@phillhocking)

April 1, 2013 at 5:22 pm



Fantastic. Infrastructure and Dev have been eschewed from each other entirely too long. I am glad that the lesson has been learned we all are more useful to each other as collaborators than adversaries.

[Reply](#)

Pingback: [Intro to DevOps | zereturnaround.com](#)

Pingback: [DevOps Scares Me - Part 1 - AppDynamics: The APM Blog](#)

Patricio

October 23, 2013 at 6:36 pm



Hi,

Would you allow me to translate this post to spanish and post it in our blog?

[Reply](#)

ernestm

October 24, 2013 at 9:05 am



Looks like you already have; that's fine but please credit/link to us here...

[Reply](#)

Patricio

October 24, 2013 at 9:58 am



Sorry about that, it wasn't support to be publish just yet.

But thanks, I'm going to update it to add the credit and links.

Thanks again.

[Reply](#)

Pingback: [HomeOps: A call for the application of Devops principles at home, too | oxf8.org](#)

Pingback: [The Phoenix Project Review | BrokenOps](#)

Pingback: [Is Your Product Team Using Ideal Devops Processes? Take a Look at These Devops Best Practices | Virtustructure](#)

grubernd

June 7, 2014 at 10:28 am



“DevOps Engineer” sounds so much better than “one man show”. gotta remember that.

[Reply](#)

ernestm

June 9, 2014 at 7:44 am



You seem to have come away with the common misconception (not from this article, clearly, but I’ve seen the popular articles on the Internet that make that claim) that DevOps means one person does everything. That is clearly not the case. “Collaboration” doesn’t mean “one man show,” it means “working together with others.” Now, working together with people does require some basic understanding of and caring about what they do – if you’re the guy who deploys apps to production but you know nothing about what the devs do, you’ll be terrible at your job. And certainly, the nature of the ops jobs are changing to require more automation – sorry, but like making buggy whips, the days of lovingly installing the OS on 20 servers by hand are passing, it’s just not professional any more. But claiming DevOps (or Agile) requires everyone to “do everything” is not credible; get out there and see some real world implementations. The trick is to help those different specialties work together without the harmful siloing and turf warring of the past.

[Reply](#)

Rajeev Kumar Gupta

June 20, 2014 at 12:36 pm



Today, I was participating in Agile India week 2014 confernce in Bangalore and came to know DevOps ‘term’ first time. After reading this blog, I felt, for so many years, why agile community only concentrated till CI and never extended to oprations. It amazing.

[Reply](#)

ernestm



June 23, 2014 at 12:06 pm



Yes, it's definitely one of those "how did we overlook this" moments! Welcome!

[Reply](#)

Pingback: [Austin, Devops and Great Folks | Cloud Insanity](#)

Tony

July 17, 2014 at 3:38 am



In Devops, is the deployment done to the real working customer environment? If so, does it make the customer (Operations) working difficult?

[Reply](#)

Pingback: [DevOps | Ciclo SW](#)

Pingback: [DevOps? A good definition | Automize.org](#)

Pingback: [Selecting a Delivery Pipeline: 10 Considerations | Jayme C Edwards](#)

Pingback: [Four Pillars of DevOps – Presentation at Agile Cambridge 2014 | johnfcshaw](#)

Pingback: [Why Media Companies Should Pay Attention to Software Development/Operations Integration - Olivier Travers](#)

Pingback: [The next wave of IT fadeouts | Nagg](#)

Chika (@mistie710)

November 27, 2014 at 9:53 am



To be honest, this sounds a little like a combination of what I used to do many years ago when designing "applications" and what I did more recently with scripting and such. It was all about first finding out what needed to be done and designing the work flow long before you got your hands dirty and actually put something together that would provide what the users needed to happen. I'm a great believer in the great cycle of technology in that nothing is ever left behind, whether it is a hardware technology, software concept or a way of thinking and this just strikes me as returning to what I did in the 1980s and 1990s before everyone got obsessed with buying the latest brand names and putting up with the inadequacies.

But I could always be wrong. It wouldn't be the first time.

[Reply](#)

Pingback: [DevOps and agile - building a new best practice](#)

Pingback: [Is There Value in DevOps? | Jason Armstrong](#)

Thomas

January 25, 2015 at 11:09 pm



Hi,

This is a very information article. Thank you very much, Could you please share your thoughts on how QA fits in the DevOps model?

[Reply](#)

ernestm

January 27, 2015 at 4:15 pm

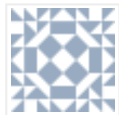


Sure, I'll write a post on it.

[Reply](#)

Tom Clifford

February 2, 2015 at 4:35 pm



Work that depends on several groups to work well has always depended on the good relations between the groups, mutual respect, and an appreciation of what those groups do, their workload, and how they have been historically treated.

Operations seems to have always been treated with less than respect, and with the ever-increasing demands on them, the increasing level of expertise needed, and the dependence of other groups on them, it pays to give them the respect they deserve.

This also goes for any group in an organization. Inter-dependence demands that we give respect and the benefit of the doubt to people that the rest of us have not walked in the shoes of.

You are all in this together: make it work with good relations.

[Reply](#)

Rom

July 1, 2015 at 11:04 am



Great article!

But in reality the mindset that the executives of less than 2000 employee company is that with devops, operations simply totally goes away because the developers takes over the operations. It is easier for developers to take over operations than operations to get some coding practice done (developers will not show their code & code

design to operations) even if a ops person can code. In the whole devops execution, ops guys (including good ones) get nothing & get fired. The whole devops thing is all about developers gaining more roles & responsibilities & job satisfaction, and is about total elimination of ops team in entirety. That is the truth.

[Reply](#)

Pingback: [DevOps theory for beginners – CSC Blogs](#)

Pingback: [Docker, SaaS, IT's Role, and Cloud Platforms - In 2015](#)

Pingback: [Definition of DevOps and the Definition of “Done” – Quick Links | Agile Advice](#)

Ritesh

February 21, 2015 at 12:08 am



Thanks, well explained.

[Reply](#)

Pingback: [DevOps, The Art of Walking in Someone Else's Shoes - The Colorful and Gray World of Engineering Management - Site Home - MSDN Blogs](#)

Mohammed

March 19, 2015 at 5:25 am



This is a great article! Thanks for posting it.

I used to work as an IT Operations Manager in a bank between the years 2002 and 2007 and since I had worked as a developer in the early years of my IT career I understood both the Ops and the Dev methods and processes. Yet, both worked in silos and there were always arguments between the two teams when a new or upgraded system was to be put into production. I tried so hard to make the Dev team to try and understand the Ops requirements but they always had the upper hand and Ops would loose out by force. What actually happened was the new/upgraded systems would fail in production because certain Ops requirements were not catered for.

What I don't understand is why would we have to wait until someone somewhere has to coin a term, “DevOps’ in this case, before the teams realise that they have to work together to ensure a seamless end-to-end process.

[Reply](#)

ernestm

April 8, 2015 at 6:27 pm



Well, it didn't wait on the term necessarily – but there's a difference between isolated people trying and a

community sharing best practices, so having common language can catalyze change.

[Reply](#)

Anwesh Singh

April 8, 2015 at 8:44 am



Interesting and very relevant thoughts. It makes sense for Ops team to be collaborating more with Dev from start as normally just after the go-live project teams are disbanded and everyone moves to other projects etc. Hence, If the collaboration had been there from the very beginning, it will be easier for Ops team as they would have known what's coming their way and help them minimise any incidents mean time to resolution (MTTR).

Anwesh

[Reply](#)

Pingback: [New CISSP CBK \(after April 15th 2015\) « Deepees's Blog](#)

Pingback: [Best Guide to Getting Started In DevOps](#)

akshatgreen

April 30, 2015 at 6:44 pm



How can we relate Devops to IT Infrastructure? Would it be the collaboration between the Build and Operate Teams(both being further segregation of sysadmin; one getting a new server ready and other taking care of it post application Go Live!) or just Infrastructure as a code via any of the configuration management/automation tool? In the second case we can surely reduce the time to deliver the system to the end user but Devops as a concept seems to be missing.

Please let me know your views.

[Reply](#)

Saurabh Mahapatra

May 13, 2015 at 10:18 pm



Thanks for the great Article.

In my organisation we are looking into the innovative way of doing operation. Major gaps found is in the expertise level between Operations and the Development. Development team are not completely aware of all the business process where as the Operations never know the coding technology.

We are still working on Dev Ops Model for doing efficient operations.

[Reply](#)

Pingback: [The DevOps interview: What to expect - Cloud2Class Blog](#)

Pingback: [5 Things I'd Tell My Enterprise IT Self | Bryan Friedman: Clouding Up](#)

Pingback: ["What is DevOps" - Can you define it?](#)

Balaji

May 20, 2015 at 9:50 am



This is perfect explanation. I liked the way this article is narrated with DevOps. Thanks for writing such wonderful and meaningful article. Also do you have image representation for DevOps? that really helps people to understand quickly

[Reply](#)

Pingback: [TECH::DevOp\(inion\) – Are we asking IT and developers to become Skynet? | Why Is The Internet Broken?](#)

Pingback: [DevOps for non-engineers - DevOps.com](#)

Frank

June 4, 2015 at 8:26 am



Great article!
It helped me understand better what DevOps is.
Thanks!!

[Reply](#)

Pingback: [Why this top cloud analyst moved to Microsoft | php Technologies](#)

Pingback: [Why this top cloud analyst moved to Microsoft - Browser Zone](#)

Pingback: [4.74 Degrees of Separation: Bill Higgins | Social Business Insights Blog](#)

Sheldon Hage

June 18, 2015 at 1:42 pm



Hmmm. This is ~exactly~ (and I mean exactly) the same job description I had 30 years ago. This is not a new job; this is a forgotten job. Once upon a time, when companies used mainframe, and processes were already mature, we had already solved what you describe here as new. Let's say "new to you".

[Reply](#)**ernestm**

October 13, 2015 at 2:48 pm



Yeah yeah, nothing is “new” in the computing world, someone has always done it before. However that’s irrelevant, it’s about what is common practice and what is better than common practice. Good on you for already knowing it all, but experience shows that most IT organizations nowadays don’t and are bleeding as a result. Use that knowledge to help...

[Reply](#)

Pingback: [Cloud Building Blocks | claudit4you](#)

Pingback: [Define DevOps: What is DevOps? | On Skylights](#)

Obaid

July 10, 2015 at 1:35 am



Started career as developer and then move to server operations and gain some where in place where managing development team with operations and Got importance of DEVOPSthers a big need for guy who aware of both else things most of time hangs between developer and admin/operations where thy pour things on each other wasting resource and time .

[Reply](#)

Pingback: [Best Guide to Getting Started In DevOps - XebiaLabs](#)

Pingback: [DevOps & Cloud: A Match Made in Heaven | Flexiant](#)

Pingback: [Vagrant Series » Introduction](#)

Pingback: [Innovating: So What Is Possible? | Paul4innovating's Innovation Views](#)

Pingback: [What is DevOps? - Daniel Bowlin's SQL Server BI Blog](#)

Pingback: [DevOps Nedir? |](#)

Pingback: [Devops – Brief explanation | rafazzevedo](#)

Pingback: [DevOps is Trending as the New Agile IT Management Methodology : eDiscovery Times](#)

prashanthpanduranga

August 14, 2015 at 1:29 am



nicey written. Let me know what you think about the DevOps definition I have provided in the following link:

<https://prashanthpanduranga.wordpress.com/2015/08/03/devops-nirvana/>

Reply

ernestm

October 13, 2015 at 2:46 pm



I do like how it notes that you can try to get there via multiple paths, “better collaboration between the silos” and “single agile team busting the silos” – just because you can’t get to agile teams for whatever reason, you can certainly improve collaboration and move towards some of the benefits.

Reply

Pingback: [Meros: My new venture | Growth Engineering](#)

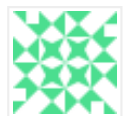
Pingback: [New Definitions Added to the Ultimate ITSM Glossary | Kinetic Vision](#)

Pingback: [Why support organizations should care about DevOps |](#)

Pingback: [Derk-Jan de Grood](#)

Dineshkumar Sakure

September 12, 2015 at 12:13 am



Thanks for wonderful explanation on DevOps...

It’s great to have operations folks involve in requirement analysis & design phase so as to prepare them for production roll out, alert stakeholders on operations requirements early in cycle. It’s also essential for Development folks to have basic understanding of Operational processes/practices.

More collaboration, cultural change with enabling tools for release & deployment would surely help. There is also necessity to have same set of tools to be used in dev & Ops for similar processes such as configuration & executable building etc. This will eliminate issues during production deployments/fixes/code management and Ops & Dev speak in same language.

However, deep multi-skill of Ops & Dev folks in each other’s areas is something not convincing. At the end of day Development folks will have to take deep dive in design & development of solutions and Ops will have to manage

steady state. Here respective specialization is surely required. Any thought...

[Reply](#)

ernestm

October 13, 2015 at 2:43 pm



Well, don't confuse skills with roles. Good production support requires expertise in both systems but also in applications; implementing APM tools and getting dev expertise onto prod support was a huge factor in improving our uptime at NI. Also, you really want dev/test systems to be built in the same way as production. You need proportionately more "traditional dev" skill set in the dev org and proportionately more "traditional ops" skill set when doing system builds but you a) need some of both skill sets throughout the service lifecycle and b) need collaboration between different teams participating in that lifecycle, if the service is so large it requires multiple teams worth of people.

But this is no different than any other subspecialization. In dev we know who the UI specialist, service specialist, really strong DB person, etc. is. But we don't put them in different groups, we have however many of each we need for the service at hand. Similarly, instead of siloing off ops, the service team needs the right amount of ops expertise to participate in all the phases – and better operational planning in service design means a lot less traditional ops monkey-work with the service in production.

[Reply](#)

Pingback: [Overcoming the IT-Business divide in an era of big data, Part 2 - The Data Roundtable](#)

Semantic MI - Microsoft MI Experts

October 26, 2015 at 7:22 am



I find it all a bit weird. I've worked in Niche tech consultancies all of my career. It's the way we have always worked since it gets the job done, quality can be protected and projects get finished and fully operational. If we didn't work this way we would never get paid. I've been to many customers that are struggling to get things done and 1 of the biggest headaches is dev, admins and ops working against each other all hiding behind superfluous paper work and processes.

[Reply](#)

Pingback: [Why Support Organizations Should Care About DevOps | HDI of the Palm Beaches](#)

jmcallister

November 2, 2015 at 12:39 pm



This is an excellent writeup, which aptly defines the concepts and role of DevOps. What hurdles have you seen in your areas of implementations? Would you suppose its primarily people, process, automation, or architecture? How did you handle those?

[Reply](#)**ernestm**

November 2, 2015 at 12:53 pm



Thanks! For that I guess I'd say "see all the other devops articles on this blog," I'm not sure I can answer all that just in a comment. People issues are always the blocking factor, however; if you don't have a people problem then any process or tech problem is just a matter of work to resolve/improve. The lack of shared goals and values among contributors to a system is the key failure point and has to be overcome to garner success. Feel free to ask about points you'd like me to expand on in a full post...

[Reply](#)**Aniket Mhala**

November 7, 2015 at 7:27 am



Great Article!

I am completely agreed with the thoughts mentioned in the article.

I believe DevOps is CRISP approach with CALMS as its principles and values

CRISP

C – Collaborative way of working between Dev and Ops (One Team – One Goal)

R – Repeatable process using principles and practices to provide better results to deliver the QUALITY and STABLE product/application

I – Iterative and Incremental approach for continuous improvement

S – Saleable across multiple teams of the product

P – People, Principle, Practices and process oriented disciplinary collaborative approach

CLAMS

C – Culture (promotes collaborative and open culture between Dev and Ops)

A – Automation (automate wherever applicable)

L- Learning (continuous learning & experimentation)

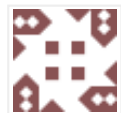
M – Measure (Measure with shared metrics across the Dev and ops for better management)

S – Sharing (Shared delivery process across Dev and Ops to build , deploy, maintain and monitor product with mentality of One Team – One Goal)

After all DevOps is collaboration between Dev and Ops using DevOps values, principles, practices and process together to remove the silos and inflexibility and deliver faster and better product to the end user in real sense.

[Reply](#)**rcd123**

November 8, 2015 at 12:46 pm



I appreciate this author's work in bringing definition to what is rapidly becoming a hackneyed term. Let's face it. The IT community has been flailing for decades, trying to figure out how to apply principles of business operation to what has been for such a long time a community of high-tech artisans. In its meanderings, it has taken comfort in latching onto a term or a phrase more for its iconic than its actual value. The principles underlying the formulation of a phrase are usually lost in the mire created by persons who don't care to think critically about what it is that they do for a living or how their role fits into a more global view of an enterprise.

At its core, both DevOps and Agile attempt to woo IT professionals out of their silos and to compel them to talk to each other. The compulsive element is not inherent in the Agile or DevOps process itself but rather in the way either is applied. A manager, for instance, who has been won over to the Agile side of the fence, may choose to employ stand-up meetings, planning poker, and the like, thinking that by requiring his charges to participate, eventually they will see the benefits therefrom. In my experience, it doesn't happen that way. Application developers, who are herded into a conference room where they are prohibited from sitting while having to listen to their peers repeat mechanically that there are no impediments to getting their work done, usually grouse about having to waste their time doing nothing constructive. They do, after all, have sprint deadlines to meet. Perhaps it would play itself out differently if the minions of IT were treated like valued collaborators instead of easily replaceable cogs in a machine. Nothing breeds contempt faster than being sold a new way of doing things by those who still have a sweatshop management mentality.

For reasons that have little to do with the style of project management employed, the IT world should not be expected to sing "Kum ba yah" anytime soon. At the heart of IT's problems lies the way that IT management makes choices – choices about hardware, choices about software tools, choices about how it measures value, choices about how it values people. I cannot begin to count the number of IT departments whose applications have been written in a hodgepodge of development languages, which were often selected because a programmer thought that one particular language or another was "cool." The evidence of this polyglot presence can be seen by simply reading job listings on any of the popular job boards. An advertisement for an application developer might stipulate that the successful applicant must have expert capability in VB6, C#/VB.NET, Python, J2EE, WebLogic, IIS, SAS, Oracle, SQL Server, MySQL...the list goes on and on. What does such a description say about the management of an IT department other than that it has been managed poorly by allowing itself to be pulled first in one direction and then another by the fashions of technology and the smooth talk of sales people? Yes, technology changes, but it is up to those who manage IT departments to discern when changes merit the investment of corporate resources. Too often, that discernment is lacking.

When IT departments are comprised of persons who ply their trades using different technical languages and concerns, should it surprise anyone that cooperation has to be coerced? It's easy to slip into us-versus-them when there are real reasons to do so. So when a methodology comes along, like Agile or DevOps, that holds the promise that it can overcome the unmanageable, it is no wonder that the very words "Agile" and "DevOps" become imbued with near mythical properties.

Agile and DevOps can bring real value to companies. Of that, there should be no doubt, but the precursor to the adoption of any new project management method should be a focus on fixing the problems of how companies are managed and how people communicate with each other.

A person whom I greatly respect gave me a definition of communication that I have remembered for a very long time. He told me that communication does not occur when Joe delivers a directive to Tim. Communication also does not occur when Joe says something to Tim and then waits to hear Tim echo back the same words. Communication, he

said, only occurs when Joe says something to Tim and then can feel within himself how Tim feels when Tim hears Joe's words and understands their meaning and their intent. If companies relied more on that kind of communication instead of bulletins pinned to a wall, mass email messages sent out to employees, or other artifices that require no personal investment in the success of another, perhaps the principles of Agile and DevOps would seem to be the natural things to implement.

[Reply](#)

Pingback: [Top DevOps Meetups - 49 Meetups For DevOps Pros](#)

Yuliya

November 9, 2015 at 12:44 pm



Correct me if I'm wrong but it seems more appropriate for large enterprise software. Kind of an overkill and too complicated if we only to mid size software like mobile applications.

[Reply](#)

ernestm

November 9, 2015 at 5:43 pm



You're wrong... It's a lot easier to do DevOps in smaller teams (fewer barriers to collaboration); one of my first DevOps gigs was with a single small team. But pretty much any group of techies above about 1 person can suffer from bottlenecks, poor communication, sloppy manual work instead of automation, etc. None of this has anything to do with scale except in that implementation can have special challenges at scale (like Agile).

[Reply](#)

Pingback: [Software Subculture Part 2: DevOps – 201 Digital](#)

Pingback: [What's happening in Service Management in 2016? | Camford MC](#)
