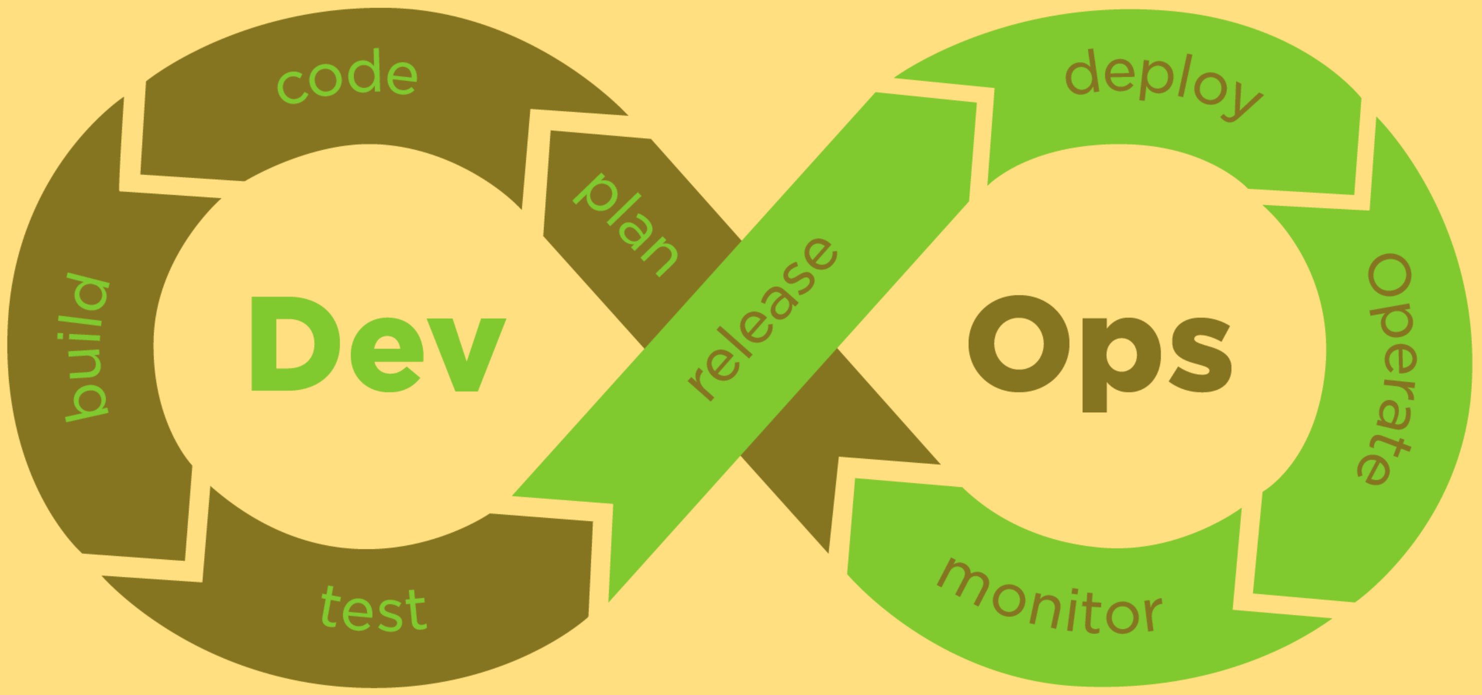


DEFINITION

Practice of operations and development engineers participating together in the entire service lifecycle from design to development and to production support. DevOps talks more about cultural and professional movement focused on how we build and operate high velocity organizations.

DevOps? Who are they?

Development

People involved in making the software

1. Developers
2. QA Testers
3. Product Managers

Operation

People involved in running and maintaining production systems

1. System Engineers
2. Network Engineers
3. DBAs
4. Security Engineers

How do we do it right?

Well the first practitioners who started the movement coined the acronym CAMS that stands for

- Culture
- Automation
- Measurement and
- Sharing

C for Culture

Communicate – On the process and progress

Collaborate – With team for feedback, brainstorming and ideas

Respect – Each other for the purpose, time and value

Empathize – Understand (I call it hug ups) one another to increase energies on positive outcomes usually known as going to work thinking

Customer Focus – Be responsive. See yourself as a customer availing a service to understand what and how a customer would expect anything that is brought/offered to work

A for Automation

Definition – Whenever you write a piece of code to perform a task that otherwise you do by hand that counts as automation

Dev - Development side of things, there are also a bunch that can be automated includes, unit testing, acceptance testing, and continuous integration

Ops – Operations on the other side has to have automation to create, scale and manage infrastructure of things

DevOps - Ties these into the deployment process eliminating the boundary between developer and operation side automation

M for Measurement

Measurement is another core part of DevOps. If you want to improve something you have to be able to measure it. When we want to know how the service is doing or whether we've improved it we look to take measurements of its performance

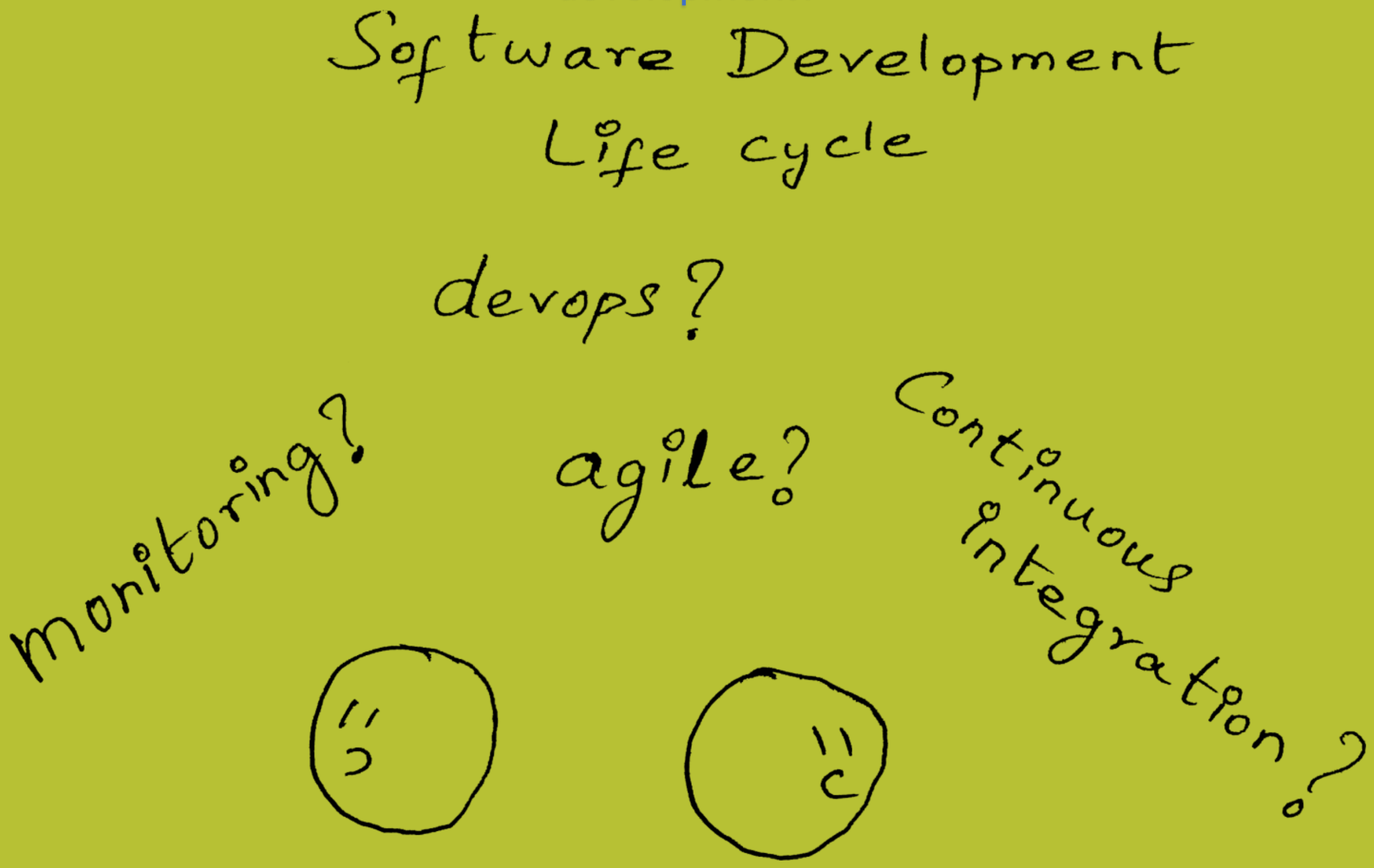
S for Sharing

While sharing can also be a culture, the end goals are generally formulated by shared input, responsibilities and ownership. People will be willing to work if their thoughts and opinions are heard on the other hand

What DevOps is not?

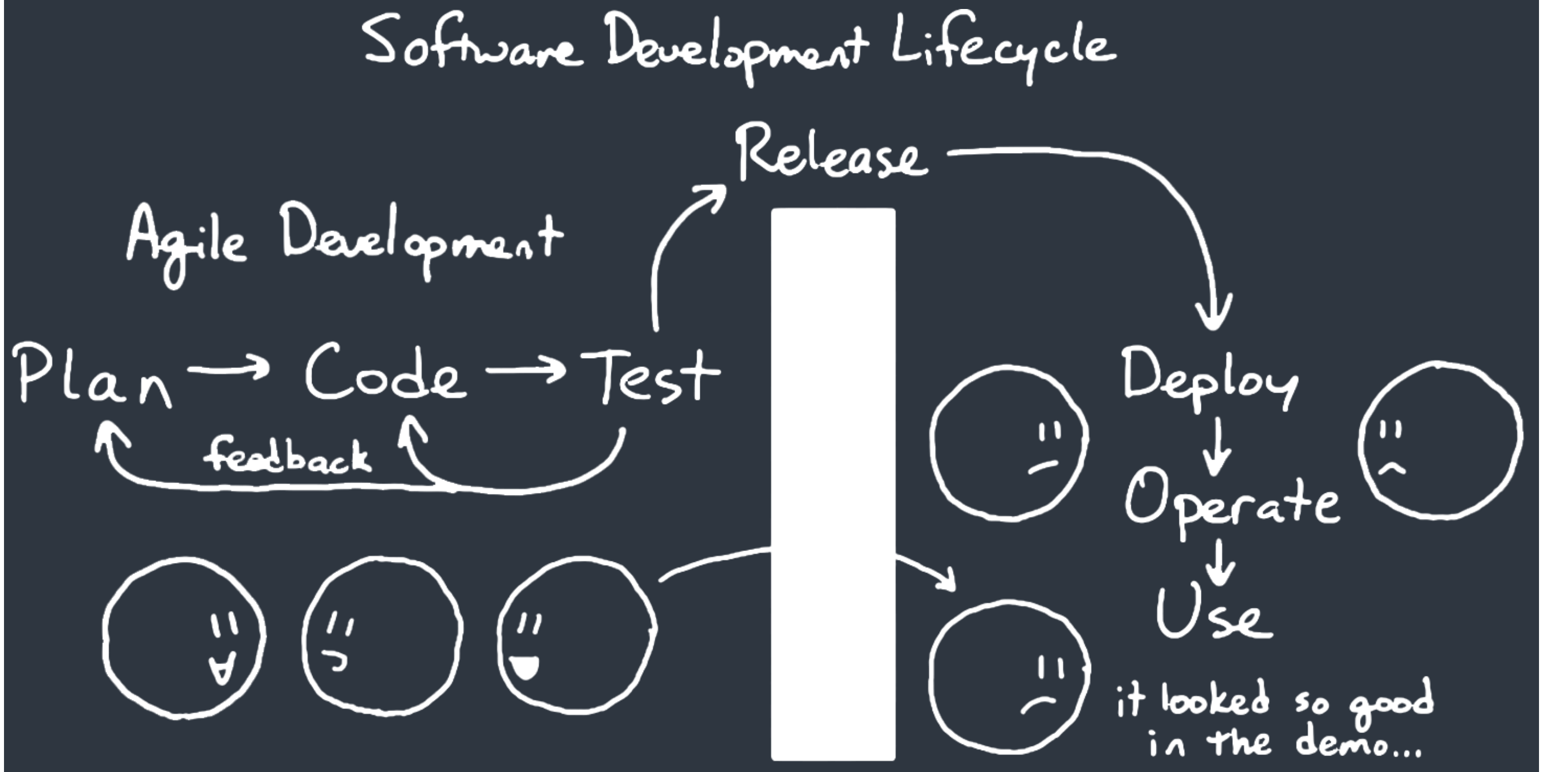
- Doing DevOps does not mean that Ops is going away or that Devs' are taking over Ops' responsibilities. Dev and Ops both have experiences and knowledge to contribute. Neither roles are being eliminated
- "DevOps" has three components: people, process, and tools. People and process are the basis for any organization. Adopting DevOps, therefore, requires making fundamental changes to the core of most organizations - not just learning new tools or not just a bunch of tool to accomplish a goal

How does DevOps fit into the world of SDLC and relate to things like agile development?



- Huge inspiration for DevOps was agile and it was formed well longer than DevOps
- Agile software development is based on iterative development where requirements and solutions evolve via a collaboration between self-organizing cross-functional teams the term was coined in 2001 when the agile manifesto was formulated
- Agile development generally covers the steps from planning to coding to building and testing and then iterating on the design again it also emphasizes collaboration between cross-functional teams
- Testing here includes unit testing of code as well as integration testing and user acceptance testing but agile doesn't necessarily speak to release process

- Deployment and operation can look very similar in an agile organization to in a traditional software organization and that's not necessarily the best solution when we start to improve this by automating build in unit test processes this is covered by continuous integration. The automation process helps to the iterative development and make sure that changes or new features do not break other functionality


Software Development Lifecycle
Agile Development
Plan
Code
Test
feedback
Release
Deploy
Operate
Use
it looked so good in the demo...

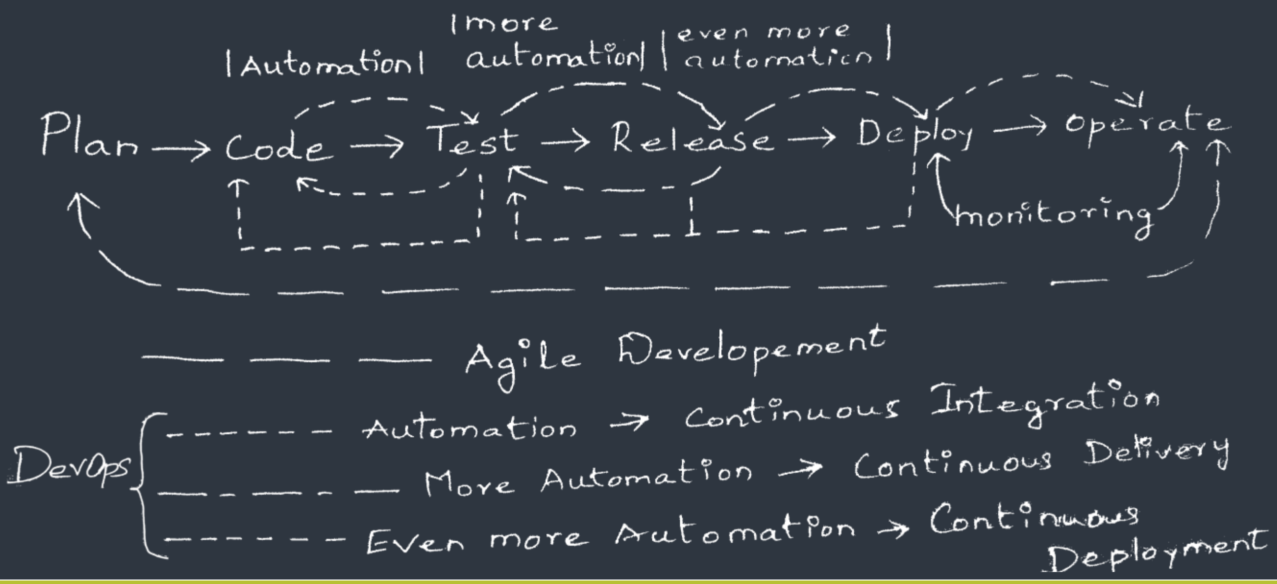
- Automating it all the way to the release is covered by continuous delivery. Continuous delivery pipeline also makes it easy to control user testing and release the software when all the tests pass
- Now to make sure it's possible to deploy all these releases into production to thousands or maybe millions of users there has to be an automated scalable way to run operations and to make sure that everything runs smoothly and to improve the whole process over time it's necessary to measure it all. But for that measurement to be actionable there has to be good communication and a culture of sharing and collaborating between all the teams involved
- Adding automation to the system is good but it's only when you combine it with the culture of communication and collaboration between teams that you really have the DevOps approach to software lifecycle

CI / CD

- Continuous Integration / Continuous Delivery is what everyone say. But these are the systems for managing your SDLC, taking your applications code all the way from the repository through testing, into release to production and automating most of the steps in between.

- Continuous Integration is an automated process of getting changes into existing code bases, building and running tests. Whenever you check-in new code it gets integrated and built but, not by hand. Popular tools like Jenkins does the job for you. It watches a code repo or branch to new commits and spawns a build process to make a complied binaries / artifacts. It then runs the test and reports the results back to the developer.

- Automating the release pipeline down to software release is called Continuous Delivery. But, since some of the steps are still manual process, you cannot release the software completely automatically. For example, QA, UAT and business cases. If you have everything automated like, fully automated testing, that’s when you call the process as Continuous Deployment.


|Automation|
|more automation|
|even more automation|
Plan → Code → Test → Release → Deploy → Operate
monitoring
Agile Developement
DevOps
Automation → Continuous Integration
More Automation → Continuous Delivery
Even more Automation → Continuous Deployment

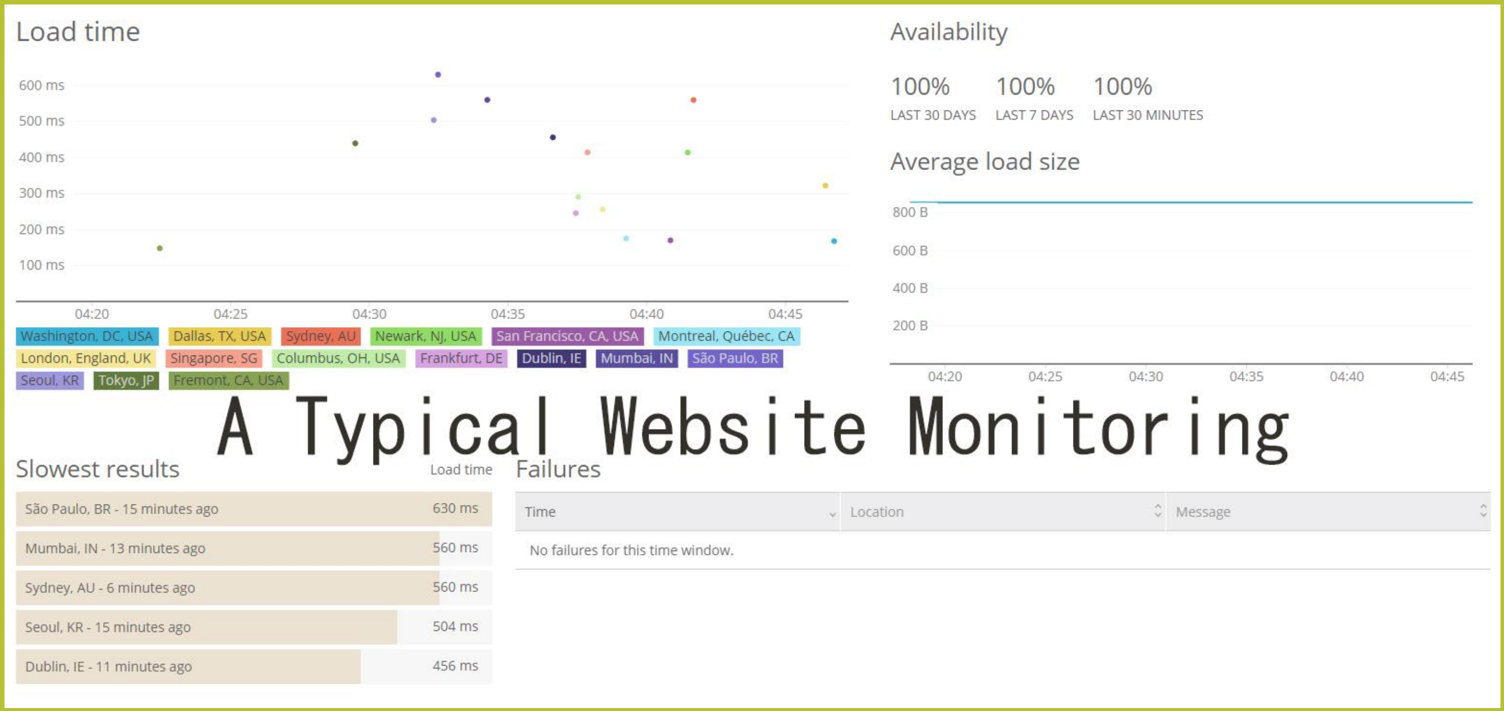
Security Compliance

Security has been a major threat in the world of Information security and compliance that are critical to any businesses around the world. DevOps offers a great prospect for improved security. Because of the automation that extents the entire pipeline. When accomplished all right, DevOps enables you to

- Secure from the start
- Secure Automatically
- Detect Early and Fix Quickly
- Integrate Easily
- Audit Regularly

Monitoring

- Like CI and CD, we can also refer CM as Continuous Monitoring. One of the principles of DevOps is measurement that includes how effective our process is. But that is not just the one.
- Monitoring also includes how well our services are running. You can't obviously scan through all of those errors with your naked eye. You can on a single server but, not on the dozens of them.
- Also one of the delusions about monitoring is that it's only meant for alerting when a service is broken. While that's the primary purpose, monitoring also tells you things that are business relevant such as
  - How well the service traffic is growing
  - Page views
  - How well the service is meeting the purpose
  - Growth Trend
- Apart from the Ops perspective on Monitoring that includes measuring infrastructure and network utilization, it gives Ops the necessary inputs to automate the scaling



A Typical Website Monitoring

Slowest results

	Load time
São Paulo, BR - 15 minutes ago	630 ms
Mumbai, IN - 13 minutes ago	560 ms
Sydney, AU - 6 minutes ago	560 ms
Seoul, KR - 15 minutes ago	504 ms
Dublin, IE - 11 minutes ago	456 ms

Failures

Time	Location	Message
No failures for this time window.		

Conclusion

DevOps practice has been an absolute game changer for challenges facing companies who are looking to hold scalable software deployments as well as the architectures and thought processes they can use to address them. The DevOps approach to the Software Development Life Cycle improves the whole process overtime by its necessary to communicate, collaborate, automate, measure and share.