|  |  |
| --- | --- |
| |  | | --- | | **TRƯỜNG ĐẠI HỌC ĐÀ LẠT**  **KHOA CÔNG NGHỆ THÔNG TIN**            **QUẢN TRỊ DỰ ÁN CÔNG NGHỆ THÔNG TIN**  **Nhóm: 17**      **Tên đề tài: Tìm hiểu về DevOps**              **Giảng viên hướng dẫn: TS. Nguyễn Thị Lương**  **Sinh viên thực hiện: 1812816 – Hoàng Thị Hồng Nhung 1812800 – Nguyễn Thành Luân**  ***Đà Lạt – 3/2022*** | |

**NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

*Đà Lạt, ngày ...... tháng ...... năm .............*

Giảng viên hướng dẫn

[Ký tên và ghi rõ họ tên]

**MỤC LỤC**

[**I.** **KHÁI NIỆM** 1](#_Toc98605006)

[*1.* *Vòng đời phát triển hệ thống* 1](#_Toc98605007)

[*2.* *Mô hình thác nước* 1](#_Toc98605008)

[*3.* *Mô hình Agile* 2](#_Toc98605009)

[*4.* *Khái niệm DevOps* 3](#_Toc98605010)

[**II.** **VÒNG ĐỜI CỦA DEVOPS** 3](#_Toc98605011)

[**III.** **CÁC** **GIAI ĐOẠN CHÍNH CỦA DEVOPS** 4](#_Toc98605012)

[**IV.** **TÍNH NĂNG CỦA DEVOPS** 5](#_Toc98605013)

[*1.* *Xây dựng máy chủ* 5](#_Toc98605014)

[*2.* *Kho mã nguồn* 5](#_Toc98605015)

[*3.* *Quản lý cấu hình* 5](#_Toc98605016)

[*4.* *Cơ sở hạ tầng ảo* 5](#_Toc98605017)

[*5.* *Bộ chứa* 6](#_Toc98605018)

[**V.** **DEVOPS ROADMAP** 6](#_Toc98605019)

[**VI.** **DEVOPS LÀ LÀM GÌ** 7](#_Toc98605020)

[**VII.** **CÔNG CỤ CỦA DevOps** 8](#_Toc98605021)

[**VIII.** **LỢI ÍCH DevOps** 10](#_Toc98605022)

[**VIII.** **KỸ SƯ DevOps LÀ LÀM GÌ?** 11](#_Toc98605023)

[**IX.** **SO SÁNH DevOps và Agile** 11](#_Toc98605024)

1. **KHÁI NIỆM**
2. ***Vòng đời phát triển hệ thống***

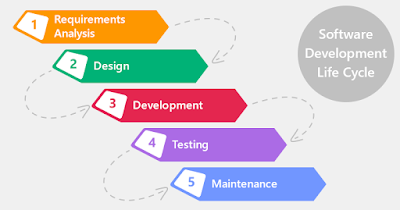
Vòng đời phát triển hệ thống (tiếng Anh: Systems Development Life Cycle, viết tắt: SDLC), hay còn được gọi là Vòng đời phát triển ứng dụng (tiếng Anh: Application Development Life Cycle), là một thuật ngữ được dùng trong Công nghệ hệ thống, Hệ thống thông tin và Công nghệ phần mềm nhằm mô tả quá trình lập kế hoạch, tạo ra, kiểm thử và triển khai một hệ thống thông tin.

Tùy thuộc vào mô hình phát triển hệ thống/phần mềm khác nhau, SDLC sẽ được chia thành các giai đoạn khác nhau. Ví dụ một số mô hình phát triển hệ thống phổ biến: mô hình thác nước (water fall), mô hình chữ V (V model), mô hình xoắn ốc (spiral model), mô hình Agile/Scrum (agile/scrum model).

1. ***Mô hình thác nước***

Thông thường, SDLC theo mô hình thác nước sẽ gồm các pha (stage) sau:

* Phân tích/đặc tả yêu cầu
* Thiết kế
* Lập trình
* Kiểm thử
* Triển khai và bảo trì

[](https://1.bp.blogspot.com/-QUhn_VgWINM/XdX56HbIhCI/AAAAAAAAEtY/PbeXc1wHJ5MkqqHwWwVbXyuBcIe8XUkUwCLcBGAsYHQ/s1600/SDLC.png)

**Ưu điểm của mô hình thác nước:**

* + Đơn giản, dễ sử dụng
  + Dễ quản lý nhờ vào tính cố định của mô hình
  + Nhờ việc kiểm duyệt tại mỗi pha nên khả năng xảy ra lỗi ít
  + Mỗi thời điểm chỉ thực hiện một pha, nên không có hiện tượng chồng chéo trong công việc
  + Thích hợp cho các dự án nhỏ

**Nhược điểm của mô hình thác nước:**

* + Không chấp nhận việc thay đổi các yêu cầu của dự án
  + Khó quay lại bước trước, nếu các bước sau phát hiện có vấn đề cần điều chỉnh
  + Quy trình làm việc nhiều rủi ro vì ít có thông tin phản hồi giữa các pha
  + Không có sản phẩm phần mềm được chuyển giao cho tới thời điểm cuối của dự án
  + Không thích hợp cho các dự án lớn và phức tạp, vì tính rủi ro cao của mô hình

1. ***Mô hình Agile***

Mô hình Agile được đưa ra để khắc phục một số nhược điểm của mô hình thác nước. Trong mô hình Agile/Scrum, mỗi chu kì làm việc (sprint) cũng tiếp cận theo các bước gần giống với mô hình thác nước. Tuy nhiên, Agile tiếp cận theo phương pháp lặp, thường xuyên chuyển giao kết quả và nhận phản hồi của các bên liên quan để điều chỉnh một cách linh hoạt.

[](https://1.bp.blogspot.com/-rkkPofG-zHQ/XdX6NIBDTII/AAAAAAAAEts/VnuLy5_cC5cKxHn72qh5lm75gJ6i19IugCLcBGAsYHQ/s1600/agile-methodology.png)

**Ưu điểm:**

* Nhanh chóng và liên tục chuyển giao từng phần của phần mềm cho khách hàng
* Tính tương tác trong đội dự án, giữa đội dự án và khách hàng được thực hiện thường xuyên, điều này giúp thông tin được minh bạch, hạn chế tình trạng hiểu sai yêu cầu công việc
* Tính thích nghi cao với các thay đổi
* Chấp nhận thay đổi yêu cầu trong suốt quá trình thực hiện dự án
* Phù hợp với các dự án có tính sáng tạo, tính khả mở cao

**Nhược điểm:**

* Ít tính dự báo, dự đoán
* Dự án gặp rủi ro nếu khách hàng không hiểu rõ về sản phầm mong muốn, hoặc khách hàng không hợp tác tốt trong quá trình thực hiện dự án
* Tài liệu, thiết kế của dự án ít được chú trọng
* Khó xác định được như thế nào gọi là hoàn thành chức năng/dự án
* Tốn nhiều thời gian cho quá trình trao đổi giữa các thành viên trong đội dự án, giữa đội dự án và khách hàng

1. ***Khái niệm DevOps***

Mô hình Agile đã khắc phục được một số hạn chế của mô hình thác nước, giúp gắn kết khách hàng và nhóm phát triển dự án.

Tuy nhiên trong mô hình Agile, vẫn có những khó khăn trong việc kết hợp giữa nhóm phát triển dự án và nhóm triển khai/vận hành. DevOps ra đời sẽ khắc phục được vấn đề này.

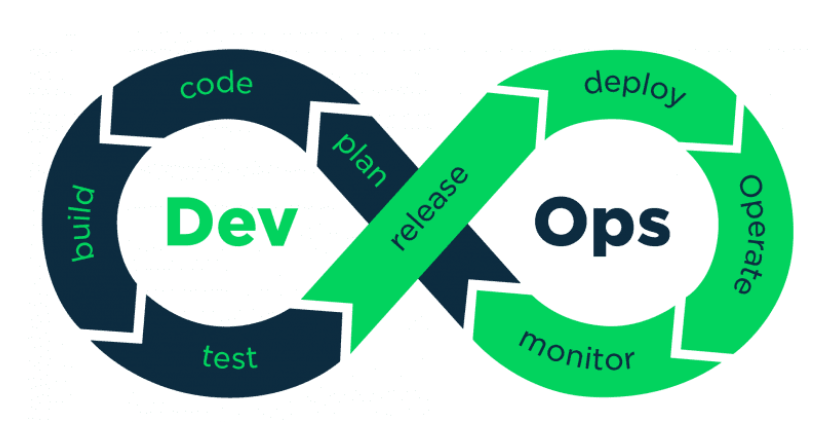
**DevOps**là sự kết hợp của từ **Dev**elopment (phát triển tính năng sản phẩm) + **Op**eration**s** (vận hành). DevOps là một văn hóa làm việc kết hợp giữa kỹ sư phát triển phần mềm (dev) với bộ phận operator (kỹ sư hệ thống, nhân viên bảo mật, kỹ sư mạng, kỹ sư hạ tầng,...) nhằm mục đích rút ngắn vòng đời phát triển sản phẩm (SDLC).

 DevOps thực chất không phải là một tools hỗ trợ làm việc mà chính là sự tổng hợp của các nguyên lý, thực tiễn, quy trình cùng các tool và mang lại sự tự động hóa trong lập trình, chuyển giao phần mềm.

Cụ thể, DevOps cho phép kỹ sư ở cả hai khâu phát triển phần mềm (Dev) với vận hành phần mềm (Ops) được thống nhất thành một đội ngũ, cùng làm một sản phẩm, cùng hiệu chỉnh sản phẩm ngay-lập-tức.

Chính điều này giúp chu trình phát triển sản phẩm được tối ưu hóa và rút ngắn thời gian ra mắt; cũng như đảm bảo khâu vận hành bài bản, nhanh chóng hơn.

1. **VÒNG ĐỜI CỦA DEVOPS**

****

* Plan: Lên kế hoạch phát triển trong sprint
* Code: developer thực hiện code chức năng theo plan
* Build: có thể buid thông qua Docker Image
* Test: cài bản build lên server test (work, staging,..) để QA và DevOps cùng test
* Relese: Tạo bản release và tham gia quá trình release
* Deploy: deploy code lên các môi trường
* Operate: vận hành app và bổ sung
* Monitor: theo dõi hệ thống xem có vấn đề gì không
* Sau đó qua sprint mới lại quay trở lại Plan và vòng lặp diễn ra liên tục

Với sự trợ giúp của mô hình hoạt động mới – DevOps, có thể nói đây là then chốt giúp giải phóng sự “cô lập” nghiệp vụ cho các đội ngũ vận hành bên trong. Các kỹ sư giữa các đội ngũ riêng biệt này giờ đây đã thành một.

Họ cùng đồng hành là việc xuyên suốt toàn bộ vòng đời sản phẩm, từ khâu thiết kế, phát triển, thử nghiệm, đến khâu phát hành và sản xuất.

Sau xây dựng mã hiệu (code) – thử nghiệm – triển khai, các kỹ sư DevOps sẽ ưu tiên sử dụng các công cụ, phần mềm để tạo ra hệ thống tự động hóa (Automation).

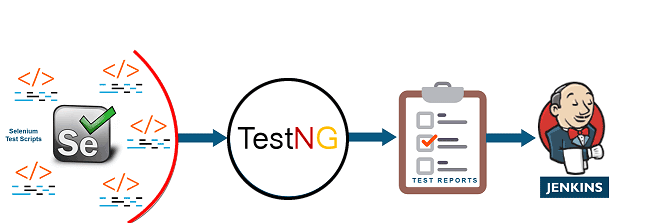
Nếu phát hiện ra lỗi phần mềm, toàn bộ đội ngũ DevOps cũng sẽ nhanh chóng kiểm tra lại sản phẩm, kịp thời tìm ra nguyên nhân chính xác, khắc phục rồi tiếp tục triển khai vận hành.

1. **CÁC** **GIAI ĐOẠN CHÍNH CỦA DEVOPS**

Các bước trong vòng đời của DevOps có thể chia thành các giai đoạn sau:

* Continuous development: Giai đoạn này liên quan đến việc lập kế hoạch và mã hóa phần mềm. Tầm nhìn của dự án được quyết định trong giai đoạn lập kế hoạch. Và các developer bắt đầu phát triển mã cho ứng dụng. Không có công cụ DevOps nào được yêu cầu để lập kế hoạch, nhưng có một số công cụ để duy trì code.

Continuous testing: Giai đoạn này, phần mềm được phát triển liên tục kiểm tra lỗi. Để kiểm tra liên tục, các công cụ kiểm tra tự động hóa như TestNG, JUnit, Selenium, v.v. được sử dụng. Các công cụ này cho phép các QAs kiểm tra song song nhiều cơ sở code để đảm bảo rằng không có sai sót trong chức năng. Trong giai đoạn này, Docker Containers có thể được sử dụng để mô phỏng môi trường thử nghiệm.

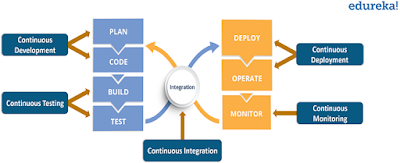


Selenium thực hiện kiểm tra tự động hóa và TestNG tạo báo cáo. Toàn bộ giai đoạn thử nghiệm này có thể tự động hóa với sự trợ giúp của công cụ Tích hợp liên tục có tên Jenkins.

Kiểm thử tự động tiết kiệm rất nhiều thời gian và công sức để thực hiện các kiểm thử thay vì thực hiện việc này theo cách thủ công. Ngoài ra, tạo báo cáo là một điểm cộng lớn. Nhiệm vụ đánh giá các trường hợp thử nghiệm không thành công trong bộ thử nghiệm trở nên đơn giản hơn. Ngoài ra, chúng tôi có thể lên lịch thực hiện các trường hợp thử nghiệm vào các thời điểm được xác định trước. Sau khi thử nghiệm, mã liên tục được tích hợp với code hiện có.

* Continuous integration: Một hình thức trong việc phát triển phần mềm, các developer thường gộp hoặc chuyển các thay đổi trong code về một repo master, và các build và test sau đó sẽ tự động chạy. Mục tiêu của CI đó là tìm bug nhanh hơn từ sớm, cải thiện chất lượng phần mềm, và giảm thiểu thời gian để xác thực và ra mắt các update mới.
* Continuous deployment: Mọi thay đổi được duyệt qua test sẽ tự động được triển khai trên production. Nghĩa là bạn pull bản build mới nhất hoặc tìm repo mới nhất đã được test và tiến hành deploy trên production. CD là một cách tuyệt vời để tăng tốc quá trình deploy sản phẩm liên tục để đáp ứng các yêu cầu liên tục của khách hàng.
* Continuous monitoring: Giám sát là một giai đoạn liên quan đến tất cả các yếu tố hoạt động của toàn bộ quy trình DevOps, nơi thông tin quan trọng về việc sử dụng phần mềm được ghi lại và xử lý cẩn thận để tìm ra xu hướng và xác định các khu vực có vấn đề. Thông thường, việc giám sát được tích hợp trong khả năng hoạt động của ứng dụng phần mềm.

Nó có thể xảy ra dưới dạng file tài liệu hoặc có thể tạo ra dữ liệu quy mô lớn về các tham số ứng dụng khi nó ở vị trí sử dụng liên tục. Các lỗi hệ thống như máy chủ không truy cập được, bộ nhớ thấp, v.v. được giải quyết trong giai đoạn này. Nó duy trì tính bảo mật và tính khả dụng của dịch vụ.

[](https://1.bp.blogspot.com/-f75XHvd41-Y/XdeVVnb5MPI/AAAAAAAAEt4/yct4rK2VxIMAKZHoiNPJLg8gqS-Y6XpVACLcBGAsYHQ/s1600/stage.png)

1. **TÍNH NĂNG CỦA DEVOPS**
2. ***Xây dựng máy chủ***

Máy chủ là một công cụ tự động hóa và cho phép mã trong kho mã nguồn được chuyển dịch thành cơ sở mã thực thi.

1. ***Kho mã nguồn***

Kho lưu trữ mã nguồn là yếu tố chính của tích hợp liên tục, đóng vai trò là nơi mà các nhà phát triển có thể quản lý các phiên bản mã khác nhau và thực hiện các thay đổi mã không làm ảnh hưởng đến hoạt động của nhau.

1. ***Quản lý cấu hình***

Quản lý cầu hình được thiết lập và duy trì chất lượng, tính nhất quán của các yêu cầu, tính năng và thuộc tính của cơ sở hạ tầng.

1. ***Cơ sở hạ tầng ảo***

Cơ sở hạ tầng ảo là các dịch vụ dựa trên đám mây cung cấp cơ sở hạ tầng hoặc nền tảng dưới dạng dịch vụ, chẳng hạn như [AWS](https://wiki.tino.org/aws-la-gi/), Microsoft Azure. Khi được sử dụng với các công cụ tự động hóa, cơ sở hạ tầng ảo hỗ trợ DevOps bằng cách cho phép quản trị viên tự động kiểm tra mã mà không cần bất kỳ sự tác động nào.

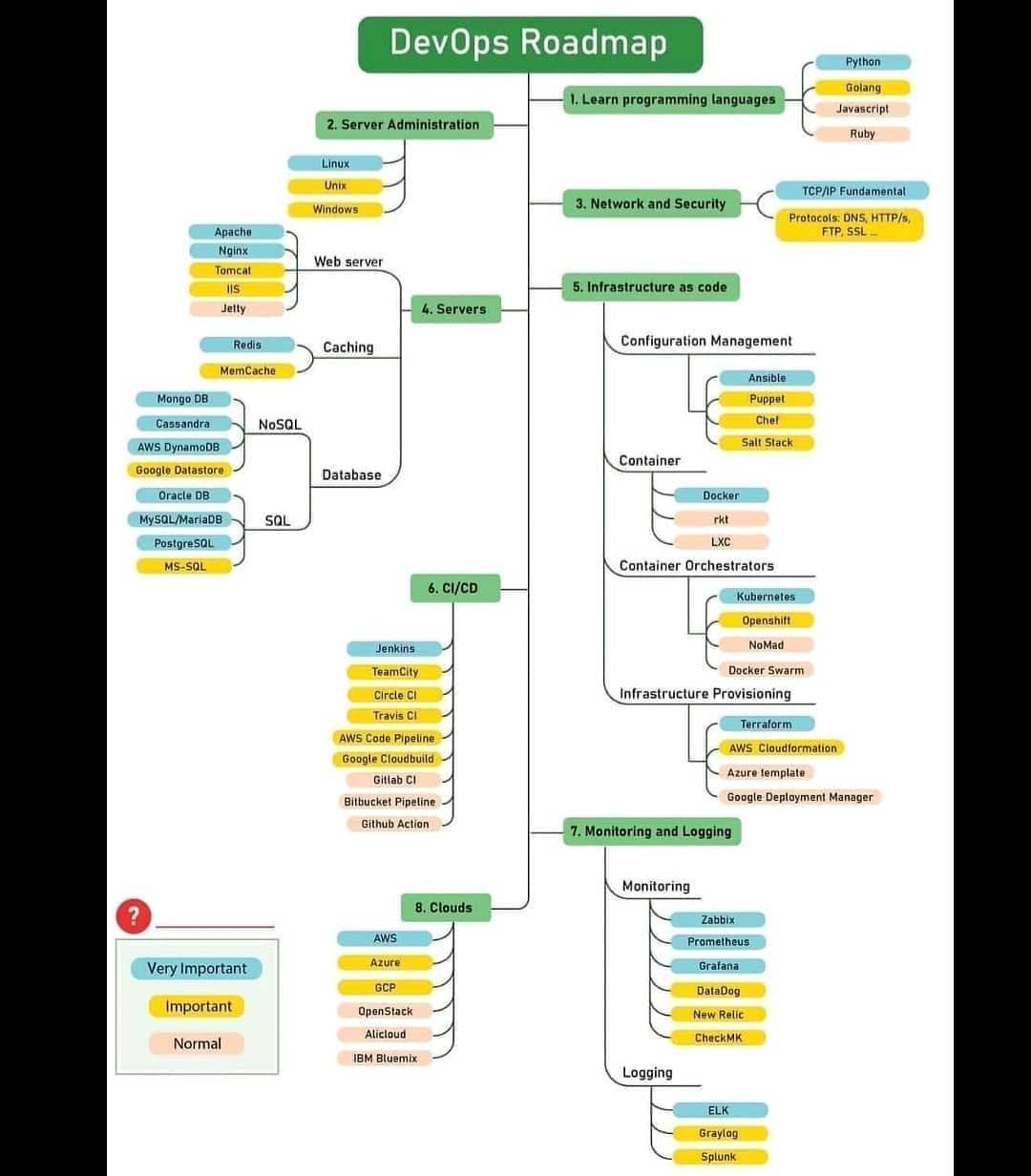
1. ***Bộ chứa***

Bộ chứa Linux là các thành phần ảo hóa, cô lập khối công việc hoặc ứng dụng nhất định ra khỏi hệ thống máy chủ trong quá trình phát triển.

1. **DEVOPS ROADMAP**

Một DevOps Engineer cần khá nhiều kỹ năng, vừa phải biết về system cũng như coding và nhiều kỹ năng như:

* Am hiểu và sử dụng thành thạo  linux, window, macOS. Khả năng sử dụng  thành thạo lệnh Terminal trong linux, CMD và powercell trong window và lệnh trong Linux hoặc Macos. Đồng thời cần phải có kiến thức cơ bản về: Process Management, Threads and Concurrency, Sockets,  Memory storage and File systems, I/O Management, Virtualization.
* Nắm chắc  kiến thức cơ bản về: DNS, HTTP, HTTPS, FTP, SSL.
* Có khả năng  cài đặt và sử dụng vài Web Server phổ biến như: Apache và Nginx. Nắm bắt được khái niệm và chức năng thường được sử dụng: Caching Server, Load balancer,  and Firewall, Reverse Proxy.
* Hiểu biết hoặc có sử dụng một số dịch vụ cloud như AWS của Amazon, Google cloud, và Azure của Microsoft.
* Khả năng code một số ngôn ngữ hệ thống như bashscript, java, javascipt , python, Php…
* Có thể sử dụng 1 số tool để triển khai CI&CD như jenkins, jira , git (Gitlab, Bitbucket…). Một DevOps Engineer nên am hiểu về Infrastructure as code):
* Containers: Docker, Kubernetes.
* Các công cụ quản lý cấu hình: Ansible, Pupet, Chef,…
* Thuần thục với  các tool monitoring server như: Nagios, Zabbix, Icing, Datadog…



1. **DEVOPS LÀ LÀM GÌ**

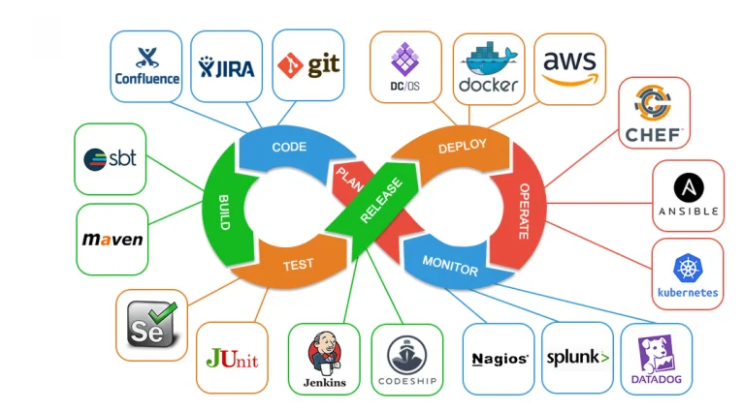
DevOps không phải là một phương pháp của riêng nó; thay vào đó, nó sử dụng các phương pháp và quy trình Agile đã biết như Kanban và Scrum, vốn phổ biến trong nhiều tổ chức CNTT. Về cơ bản, đây là các phương pháp quản lý dự án dựa trên các khái niệm Agile và được sử dụng để phát triển (chủ yếu là Scrum) và hoạt động (chủ yếu là Kanban). Các khái niệm chính là phát triển liên tục, tích hợp liên tục và triển khai liên tục. Điều quan trọng là làm việc với các thay đổi nhỏ thay vì các bản phát hành lớn (giúp giảm thiểu rủi ro), loại bỏ các bước thủ công bằng cách tự động hóa các quy trình và có môi trường phát triển và thử nghiệm càng gần với môi trường sản xuất càng tốt.

Nó giúp hoàn thiện việc chuyển đổi quy trình phát triển và vận hành phần mềm từ mô hình thác nước (waterfall) sang mô hình phát triển/phát hành liên tục (CI/CD).

Tất cả đều phục vụ cho mục đích cuối cùng là cải thiện khả năng triển khai phần mềm một cách nhanh chóng. Từ đó, tăng khả năng cạnh tranh của sản phẩm/doanh nghiệp.

1. **CÔNG CỤ CỦA DevOps**

Lý do chính để triển khai DevOps là cải thiện quy trình phân phối và quy trình tích hợp bằng cách tự động hóa các hoạt động này. Do đó, sản phẩm có thời gian đưa ra thị trường ngắn hơn. Để đạt được quy trình phát hành tự động này, nhóm phải có được các công cụ cụ thể thay vì xây dựng chúng từ đầu.



|  |  |  |  |
| --- | --- | --- | --- |
|  | Domain | Vòng đời | Các công cụ DevOps mã nguồn mở phổ biến |
| 1 | Dev | Plan | Kanboard, Wekan, Trello; GitLab, Tuleap, Redmine, JIRA; Mattermost, Roit.im, IRC, Slack |
| 2 | Code | Git, Gerrit, Bugzilla; Jenkins và các công cụ CI/CD mã nguồn mở |
| 3 | Build | Apache Maven, Gradle, Apache Ant, Packer |
| 4 | Test | JUnit, Cucumber, Selenium, Apache JMeter |
| 5 | Ops | Release | Kubernetes, Nomad, Jenkins, Zuul, Spinnaker, Ansible, Apache ZooKeeper, etcd, Netflix Archaius, Terraform |
| 6 | Deploy |
| 7 | Operate |
| 8 | Monitor | Grafana, Prometheus, Nagios, InfluxDB, Fluentd và các công cụ khác |

Hiện tại, các công cụ DevOps hiện có bao gồm hầu hết các giai đoạn của quá trình phân phối liên tục, bắt đầu từ môi trường tích hợp liên tục và kết thúc bằng quá trình chứa và triển khai. Mặc dù ngày nay một số quy trình vẫn được tự động hóa với các tập lệnh tùy chỉnh, hầu hết các kỹ sư DevOps sử dụng các sản phẩm mã nguồn mở khác nhau. Chúng ta hãy xem xét những cái phổ biến nhất:

* [Jenkins](https://jenkins.io/) là một công cụ để sử dụng như một máy chủ để tích hợp liên tục hoặc một trung tâm phân phối liên tục đi kèm với nhiều plugin bổ sung để điều chỉnh quy trình phân phối liên tục.
* [Selenium](https://www.seleniumhq.org/) là một trình duyệt tự động cho phép nhóm QA viết kịch bản và kiểm tra các sản phẩm web. Nó tương thích với tám ngôn ngữ lập trình phổ biến. Tìm hiểu thêm về Selenium trong bài viết của chúng tôi về [các công cụ tự động hóa QA](https://www.altexsoft.com/blog/engineering/comparing-automated-testing-tools-selenium-testcomplete-ranorex-and-more/?utm_source=DZone&utm_medium=referral).
* [Git](https://git-scm.com/) là Hệ thống kiểm soát phiên bản với kho lưu trữ để quản lý mã nguồn cho phép làm việc trực tuyến và ngoại tuyến.
* [Chef](https://www.chef.io/chef/) là một công cụ dành cho cơ sở hạ tầng như quản lý mã chạy cả trên máy chủ phần cứng và đám mây. Một công cụ phổ biến khác trong danh mục này là [Ansible](https://www.ansible.com/) tự động hóa quản lý cấu hình, cung cấp đám mây và triển khai ứng dụng.
* [Docker](https://www.docker.com/) là một công cụ giúp mã đóng gói thành các đơn vị độc lập, tức là các thùng chứa.
* [Nagios](https://www.nagios.org/) là một công cụ giám sát cơ sở hạ tầng trình bày các phân tích trong các báo cáo trực quan.

Trong khi kỹ sư DevOps - chúng tôi sẽ thảo luận chi tiết hơn về vai trò này bên dưới - phải vận hành các công cụ này, những người còn lại trong nhóm cũng sử dụng chúng dưới sự hỗ trợ của kỹ sư DevOps.

1. **LỢI ÍCH DevOps**
2. **Tốc độ** :*DevOps giúp các developers và team operations đạt được mục tiêu ở một tốc độ khác giúp cải tiến sản phẩm nhanh chóng phục vụ người dùng, thích nghi với thị trường tốt hơn và điểu chỉnh hiểu quả kinh doanh hiệu quả hơn.*
3. **Chuyển giao nhanh chóng**: *Tăng tốc độ release thường xuyên để chúng ta cải thiện sản phẩm nhanh hơn và cho ra mắt các feature nhanh hơn cũng như fix bug, giúp phản hồi cho khách hàng nhanh chóng và xây dựng nên lợi thế cạnh tranh tốt hơn.*
4. **Độ tin cậy** : *DevOps đảm bảo chất lượng bằng cách áp dụng CI /CD, Monitoring và logging process. Bằng cách update mà team infrastructure cấp quyền cho team development để chuyển giao nhanh hơn mà vẫn duy trì được trải nghiệm người dùng tốt.*
5. **Mở rộng** : *Team vận hành, quản lý infra và các quy trình. Lên kế hoạch về quy mô và nâng cấp môi trường giúp quản trị các hệ thống phức tạp hoặc hay thay đổi hiệu quả cũng như giảm thiểu rủi ro.*
6. **Bảo mật** : *DevOps giúp di chuyển mà không chịu tổn thất về bảo mật bằng các chính sách, kiểm soát và phương pháp quản lý configuration. Thậm chí các team có thể kêu gọi bạn theo các tiêu chuẩn từ sớm bằng cách cung cấp các setup các tool theo dõi.*
7. **KỸ SƯ DevOps LÀ LÀM GÌ?**

Kỹ sư DevOps là người am hiểu về Vòng đời phát triển phần mềm và làm việc sâu sát với cả Developer và IT Operation. Họ phải đảm bảo rằng giữa 2 bộ phận có sự liên kết chặt chẽ; thống nhất về mục tiêu và định hướng.

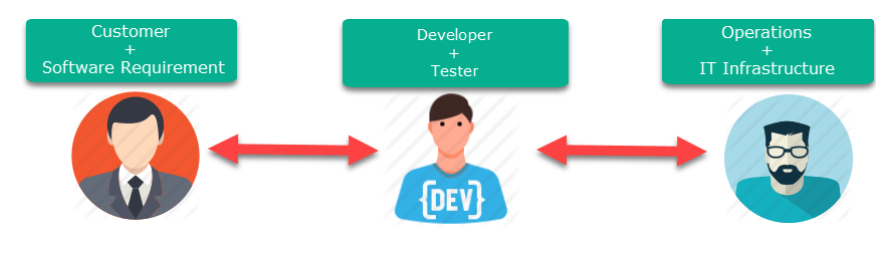
Nhờ đó, kỹ sư DevOps vừa giám sát vừa thúc đẩy việc tạo ra những đoạn code hiệu quả; đồng thời quản lý cơ sở hạ tầng công nghệ một cách tối ưu.

Như vậy; chúng ta có thể nhận ra một Kỹ sư DevOps sẽ cùng lúc nắm giữ nhiều vai trò quan trọng:

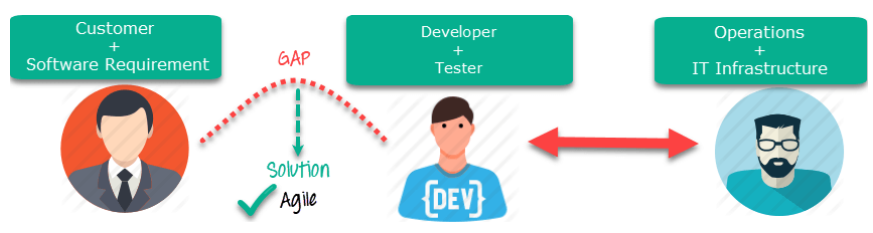
* **DevOps Evangelist:** Người chịu trách nhiệm triển khai các hoạt động DevOps
* **Release Manager**: Người cho ra những tính năng mới; đảm bảo sản phẩm hoạt động ổn định sau mỗi đợt release.
* **Automation Expert**: Chuyên gia về tự động hóa và sắp xếp các công cụ hỗ trợ.
* **Software Developer/ Tester**: Lập trình và kiểm tra (test) phần mềm.
* **Quality Assurance**: Người kiểm tra và đảm bảo chất lượng thực tế của sản phẩm.
* **Security Engineer**: Theo dõi tính bảo mật và sức khỏe của sản phẩm.

1. **SO SÁNH DevOps và Agile**

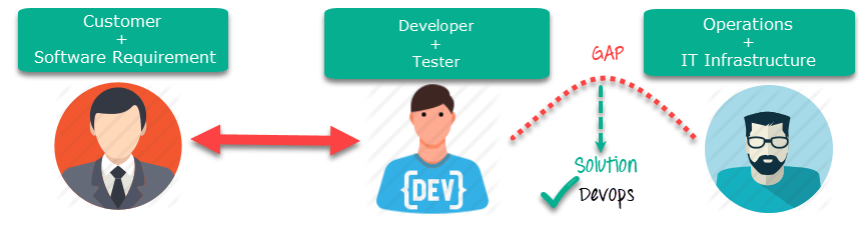
Các bên liên quan và chuỗi truyền thông một quy trình CNTT điển hình:



Agile giải quyết các khoảng trống trong giao tiếp giữa khách hàng và nhà phát triển



DevOps giải quyết các lỗ hổng trong truyền thông Hoạt động của Nhà phát triển và CNTT



|  |  |  |
| --- | --- | --- |
| Parameter | Agile | DevOps |
| - | Agile đề cập đến một cách tiếp cận lặp đi lặp lại, tập trung vào sự cộng tác, phản hồi của khách hàng và các bản phát hành nhỏ, nhanh chóng. | DevOps được coi là một thực tế đưa các nhóm phát triển và nhóm hoạt động lại gần nhau. |
| Mục đích | Agile giúp quản lý các dự án phức tạp. | Mô hình chính của DevOps là quản lý các quy trình kỹ thuật từ đầu đến cuối. |
| Công việc | Mô hình agile tập trung vào những thay đổi liên tục. | DevOps tập trung vào kiểm thử và phân phối liên tục. |
| Thực hiện | Agile có thể được thực hiện dựa trên những chiến lược có sẵn như sprint, safe and scrum. | Mục tiêu chính của DevOps là tập trung vào cộng tác, vì vậy nó không có bất kỳ framework được chấp nhận phổ biến nào. |
| Kỹ năng team | Phát triển nhanh, nhấn mạnh đào tạo tất cả các thành viên trong nhóm để có kỹ năng đồng đều. | DevOps chia và dàn trải kỹ năng giữa các nhóm phát triển và vận hành. |
| Team size | Nhóm nhỏ là cốt lõi của Agile.Khi nhóm nhỏ hơn, số người càng ít thì họ càng di chuyển nhanh hơn. | Kích thước nhóm tương đối lớn. |
| Thời lượng | Quá trình agile được quản lý theo đơn vị "sprints".Thời gian này ít hơn một tháng cho mỗi giai đoạn. | DevOps tập trung vào thời giạn release những bản phát hành chính thức. Mục tiêu là bàn giao code sản phẩm hang ngày, hàng giờ. |
| Phản hồi | Phản hồi được đưa ra bởi khách hàng. | Phản hồi đến từ nhóm nội bộ. |
| Mục tiêu | Phát triển phần mềm. | Giải pháp kinh doanh đầu cuối và giao hàng nhanh. |
| Nguyên tắc Shift-Left | Tận dụng shift-left. 'Shift Left' có nghĩa là thay đổi từ phải sang trái bắt nguồn từ 'Giai đoạn Kiểm thử' hoặc rõ hơn là liên quan đến Kiểm thử ở mọi giai đoạn và kiểm thử liên tục trong suốt quá trình phát triển phần mềm . Hỗ trợ việc tạo thành các nhóm Scrum Agile, bao gồm các QA có các vai trò khác nhau. | Tận dụng cả hai, cả trái và phải (shift left và shift right). |
| Nhấn mạnh | Agile nhấn mạnh vào phương pháp phát triển phần mềm để phát triển phần mềm. Khi phần mềm được phát triển và phát hành, team agile sẽ không quan tâm điều gì xảy ra với nó. | DevOps là tất cả về việc lấy phần mềm đã sẵn sàng để phát hành và triển khai nó một cách đáng tin cậy và an toàn. |
| Chức năng xuyên suốt | Bất kỳ thành viên nhóm nào cũng có thể làm những gì cần thiết cho tiến độ của dự án. Ngoài ra, khi mỗi thành viên trong nhóm có thể thực hiện mọi công việc, nó làm tăng sự hiểu biết và liên kết giữa chúng. | Trong DevOps, các nhóm phát triển và các nhóm hoạt động riêng biệt. Vì vậy, giao tiếp khá phức tạp. |
| Giao tiếp | Scrum là phương pháp phổ biến nhất để triển khai phát triển phần mềm Agile. Cuộc họp scrum hàng ngày được thực hiện. | Giao tiếp trong DevOps liên quan đến thông số kỹ thuật và tài liệu thiết kế. Điều quan trọng là nhóm hoạt động phải hiểu đầy đủ về việc phát hành phần mềm và các hệ thống phần cứng / mạng để chạy quá trình triển khai. |
| Tài liệu | Phương pháp Agile ưu tiên cho hệ thống có tài liệu hoàn chỉnh. Tuy nhiên, nó có thể sẽ không tốt khi bạn đang cố gắng chuyển mọi thứ sang một nhóm khác để triển khai. | Trong DevOps, tài liệu quy trình là quan trọng nhất vì nó sẽ được gửi cho nhóm hoạt động để triển khai. Tự động khắc phục, giảm thiểu tác động của việc tài liệu mô tả thiếu.Tuy nhiên, trong sự phát triển của phần mềm phức tạp, thật khó để chuyển tất cả các kiến thức cần thiết. |
| Tự động hóa | Agile không nhấn mạnh vào tự động hóa. Mặc dù nó hỗ trợ. | Tự động hóa là mục tiêu chính của DevOps. Nó hoạt động trên nguyên tắc để tối đa hóa hiệu quả khi triển khai phần mềm. |
| Mục tiêu | Nó giải quyết khoảng cách giữa nhu cầu của khách hàng và các nhóm phát triển và thử nghiệm. | Nó giải quyết khoảng cách giữa phát triển, kiểm thử và Ops. |
| Tiêu điểm | Nó tập trung vào sự sẵn sàng chức năng và phi chức năng. | Nó tập trung nhiều hơn vào sự sẵn sàng hoạt động và kinh doanh. |
| Tầm quan trọng | Agile tập trung vào quá trình phát triển phần mềm. | Phát triển, kiểm thử và thực hiện tất cả đều quan trọng không kém. |
| Tốc độ so với rủi ro | Các nhóm sử dụng Agile hỗ trợ thay đổi nhanh chóng và cấu trúc ứng dụng mạnh mẽ. | Trong phương phápDevOps, các team phải đảm bảo rằng những thay đổi được thực hiện cho kiến trúc không tạo ra rủi ro cho toàn bộ dự án. |
| Chất lượng | Agile sản xuất các bộ ứng dụng tốt hơn với các yêu cầu mong muốn. Nó có thể dễ dàng thích ứng theo những thay đổi được thực hiện về thời gian, trong suốt cuộc đời dự án. | DevOps, cùng với tự động và gỡ lỗi sớm, góp phần tạo ra chất lượng tốt hơn. Các nhà phát triển cần tuân theo các phương pháp tốt nhất về Mã hóa và Kiến trúc để duy trì các tiêu chuẩn chất lượng. |
| Công cụ được sử dụng | JIRA, Bugzilla, Kanboard là một số công cụ Agile phổ biến. | Puppet, Chef, TeamCity OpenStack, AWS là các công cụ DevOps phổ biến. |
| Thách thức | Phương pháp nhanh nhẹn cần các đội làm việc hiệu quả hơn, rất khó để phù hợp với mọi thời gian. | Quá trình DevOps cần phải phát triển, thử nghiệm và môi trường sản xuất để sắp xếp công việc. |
| Lợi thế | Agile cung cấp chu kỳ phát triển ngắn hơn và phát hiện khiếm khuyết được cải thiện. | DevOps hỗ trợ chu kỳ phát hành của Agile. |

**TÀI LIỆU THAM KHẢO**

1. <https://vi.wikipedia.org/wiki/DevOps>
2. <https://aws.amazon.com/vi/devops/what-is-devops/>
3. <https://glints.com/vn/blog/vi-tri-devops-dac-biet-the-nao/#.YjS3g-pBxPZ>
4. <https://helpex.vn/article/devops-nguyen-tac-thuc-tien-va-vai-tro-cua-ky-su-devops-609820ecbb15259b1aaddd07>
5. <https://www.powershow.com/view0/8e9bf7-M2I5N/Introduction_to_5w_s_of_DevOps_powerpoint_ppt_presentation>