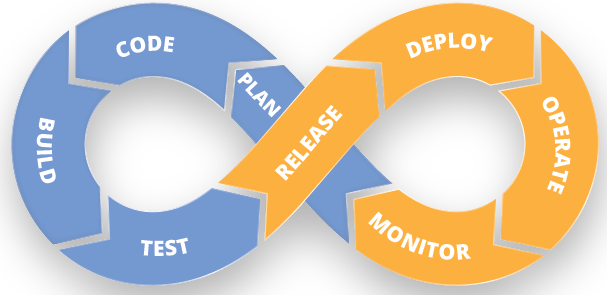
**So sánh** **Puppet vs. Chef vs. Ansible vs. SaltStack**

**Vấn đề về DevOps?**

DevOps là phương pháp, qui trình phát triển phần mềm bao gồm quá trình phát triển, quá trình test, quá trình tích hợp, quá trình triển khai, quá trình giám sát thông qua vòng đời phát triển của sản phẩm. Các bước này diễn ra tuần tự không bỏ qua hoặc nhảy cách. DevOps là phường pháp ưu tiên khi muốn phát triển 1 sản phẩm có chất lượng cao trong 1 thời giăn ngắn mà vẫn khiến khách hàng cảm thấy hài lòng

Mô hình phát triển (DevOps life cycle)



**Quá trình phát triển (Continuous Development )**

Tại bước này, sản phẩm được phát triển, nó bao gồm giai đoạn code và build, thường sử dụng Git, Svn cho việc duy trì các phiên bản, các tool Ant, Maven, Gradle cho phép build tạm thời sản phẩm, cho phép việc chạy test, Qas

**Quá trình Tesing (Continuous Testing)**

Tại bước này, sản phẩm được test, tìm các lỗi. Tại bước này hay sử dụng các tool như Selenium, Junit, … Bước này đảm bảo sản phẩm không có lỗ hổng trước khi đến bước tiếp theo

**Quá trình tích hợp (Continuous Integration)**

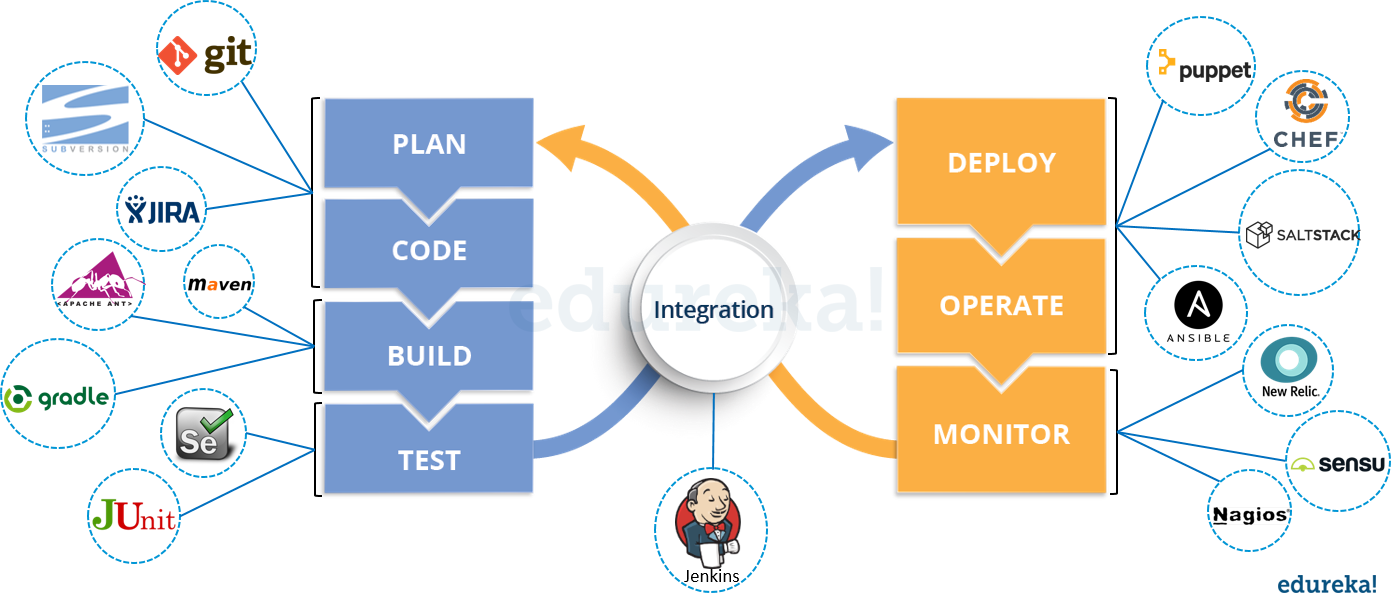
Tại bước này, source code được update thêm các tính năng mới (các tính năng mới phát triển). Quá trình update mã nguồn cần diễn ra chính xác, không sinh ra các lỗi mỗi trường trong quá trình runtime, cho test, dùng thử ổn định. Công cụ thường sử dụng (Jenkins)

**Quá trình triển khai (Continuous Deployment)**

Tại bước này, source code được triển khai vào các môi trường thực tế, các tính năng mới cần được đồng bộ cập nhật tới tất cả server. Tại đây các công cụ Puppet, Chef, SaltStack và Ansible thường được các sysad sử dụng.

**Quá trình giám sát (Continuous Monitoring)**

Đây là bước tất yếu của quá trình phát triển DevOps. Mục đích bước này là nâng cao hiệu suất thông qua việc giám sát trạng thái, hiệu năng của hệ thống thực. Các công cụ thường được sử dụng Nagios, NewRelic, Sensu.



**Tại sao lại cần đến các tool Puppet vs. Chef vs. Ansible vs. SaltStack trong quá trình deploy?**

Khi cơ sở hạ tầng được phân tán hoặc được xây dựng trên cơ sở cloud, việc phải triển khai thường xuyên các dịch vụ giống nhau diên ra thường xuyên và các vấn đề xảy ra các xung đột trên các server diễn ra thường xuyên, việc cấu hình tay trở nên tốn nhiều thời gian và công sức. Vì vậy, sinh ra các công cụ hỗ trợ việc quản lý các cấu hình và tự động triển khai các dịch vụ 1 cách động bộ để giảm bớt chi phí, công sức, tạo ra nhiều thời gian hơn cho các sysad nghiên cứu phát triển các vấn đề khác.

**So sánh Puppet vs.** **Chef vs.** **Ansible vs. SaltStack**

**Khả năng mở rộng**

|  |  |
| --- | --- |
| Puppet:  - Có khả năng quản lý tới 10.000+ node | Chef:  - Có khả năng quản lý 10.000+ node |
| Ansible | SaltStack |

**Mô hình cài đặt**

|  |  |
| --- | --- |
| Puppet:  - Puppet server chạy trên master machine  - Puppet clients chạy các agent | SaltStack  - SaltStack server chạy trên master machine  - SaltStack client chạy các agent tên minions |
| Ansible  - Chỉ tồn tại duy nhất master chạy trên server machine  - không tồn tại agent, sử dụng ssh cho việc truy cập vào mỗi client | Chef:  - Chef server chạy trên master machine  - Chef clients chạy các agent |

**Tính sẵn sàng, ổn định**

|  |  |
| --- | --- |
| Puppet:  - Sử dụng mô hình multi-master  - Nếu active-master gặp sự cố, 1 master server khác thay thế | SaltStack  - Có thể cấu hình multi master  - Nếu 1 master gặp vấn đề, agents sẽ kết nối master đã được chỉ định trước |
| Ansible  - Ansible server chính chạy dưới dạng primary master  - Nếu primary gặp sự cố, second instance sẽ thay thế | Chef:  - Nếu master server gặp sự cố, backup server sẽ thay thế |

**Ngôn ngữ cấu hình**

|  |  |
| --- | --- |
| Puppet:  - Sử dụng ngôn ngữ riêng tên Puppet DSL (domain specific language) or Ruby  - Client pull source từ server  - Update không trực tiếp đến cách kết nối | SaltStack  - Sử dụng ngôn ngữ Python hay YAML  - Server sẽ đẩy source đến tất cả client  - Update ngay lập tức đến các kết nối |
| Ansible  - Sử dụng YAML (Python)  - Server push đến các client  - Up date ngay lập tức đến các kết nối  - Phù hợp với các app real time | Chef:  - Cần phải hiểu về cấu trúc Ruby DSL  - Client pull source từ server |

**Nền tảng tương thích**

|  |  |
| --- | --- |
| Puppet:  - Puppet server chạy trên Unix, Linux  - Puppet client chỉ chạy trên window, linux, unix | SaltStack  - SaltStack master chỉ chạy trên Unix, Linux  - Salt minion có thế chạy trên unix, linux, windows |
| Ansible  - Ansible server chỉ chạy trên Unix, Linux  - Ansible client chạy trên unix, linux, windows | Chef:  - Chef server chỉ chạy trên Unix, Linux  - Chef client chạy trên unix, linux, windows |

**Độ lớn Project**

|  |  |
| --- | --- |
| Puppet:  - Contributors: 355  - Commits: 19595  - Braches: 9  - Releases: 291 | SaltStack  - Contributors: 1041  - Commits: 49193  - Braches: 11  - Releases: 83 |
| Ansible  - Contributors: 1003  - Commits: 13527  - Braches: 33  - Releases: 57 | Chef:  - Contributors: 369  - Commits: 12089  - Braches: 177  - Releases: 231 |

**Enterprise Cost**

|  |  |
| --- | --- |
| Puppet:  - 12000$ | SaltStack  - 15000$ |
| Ansible  - 10000$ | Chef:  - 7200$ |

**Đánh giá**

**Ansible**

Điểm mạnh

* Triển khai nhanh (không phải cài agent trên các client) sử dụng ssh-base
* Cú pháp đơn giản, rõ ràng dễ học (YAML)
* Sắp xếp project hợp lý

Điểm yếu:

* Hiệu năng kém hơn so với các tool xây dựng dự trên ngôn ngữ khác
* Các logic khó nhớ, thường xuyên phải sử dụng documentation
* Không có tính nhất quán giữa input, output, config file

**Cheft**

Điểm mạnh:

* Có sự tương quan với GIT
* Nhiều module, cấu hình linh hoạt
* Phương pháp tiếp cận linh hoạt

Điểm yếu:

* Phải có hiểu biết về Ruby
* Hạ tầng cơ sở càng lớn thì source code sẽ trở nên phức tập
* Không hỗ trợ tính năng push

**Puppet**

Điểm mạnh

* Được xây dựng ổn định, phù hợp với các môi trường phức tạp
* Đã có cộng đồng hỗ trợ riêng
* Đơn giản trong việc triển khai, khởi tạo
* WebUI gần như hoàn thiện
* Khả năng sinh báo cáo tốt

Điểm yếu:

* Đối với các nhiệm vụ phức tạp, cần sử dụng giao diện cli(cmd), cú pháp dự trên ruby
* Khó đối với người mới, đối với các project phức tạp, việc tiếp cận khó

**Saltstack**

* Xây dựng dựa trên CLI, Python
* Đáp ứng tốt cho việc mở rộng, ổn định

Điểm mạnh:

* Sử dụng đơn giản sau khi triển khai
* DSL linh hoạt, không yều cấu các logic phức tạp
* Đầu vào, đầu ra, config nhất quán – sử dụng YAML
* Có cộng đồng hỗ trợ
* Có khả năng mở rộng cao, ổn định

Điểm yếu:

* Khó khăn cho việc setup, lựa chọn 1 user mới
* Doc khó hiểu trong việc giới thiệu
* Giao diện Web mới, chưa hoàn thiện
* Hỗ trợ chưa tốt các hệ điều hành khác Linux

Nguồn:

<https://www.youtube.com/watch?v=OmRxKQHtDbY>

<https://www.edureka.co/blog/what-is-devops/?utm_source=blog&utm_medium=left-menu&utm_campaign=devops-tutorial>

<http://blog.takipi.com/deployment-management-tools-chef-vs-puppet-vs-ansible-vs-saltstack-vs-fabric/>

<http://kcntt.duytan.edu.vn/Home/ArticleDetail/vn/128/2376/cong-cu-quan-ly-cau-hinh--puppet-phan-1>

<http://www.intigua.com/blog/puppet-vs.-chef-vs.-ansible-vs.-saltstack>

<http://kcntt.duytan.edu.vn/Home/ArticleDetail/vn/128/2376/cong-cu-quan-ly-cau-hinh--puppet-phan-1>