DevOps And Tools

Exploring of DevOps and its related tools

@Mohammed Fazuluddin

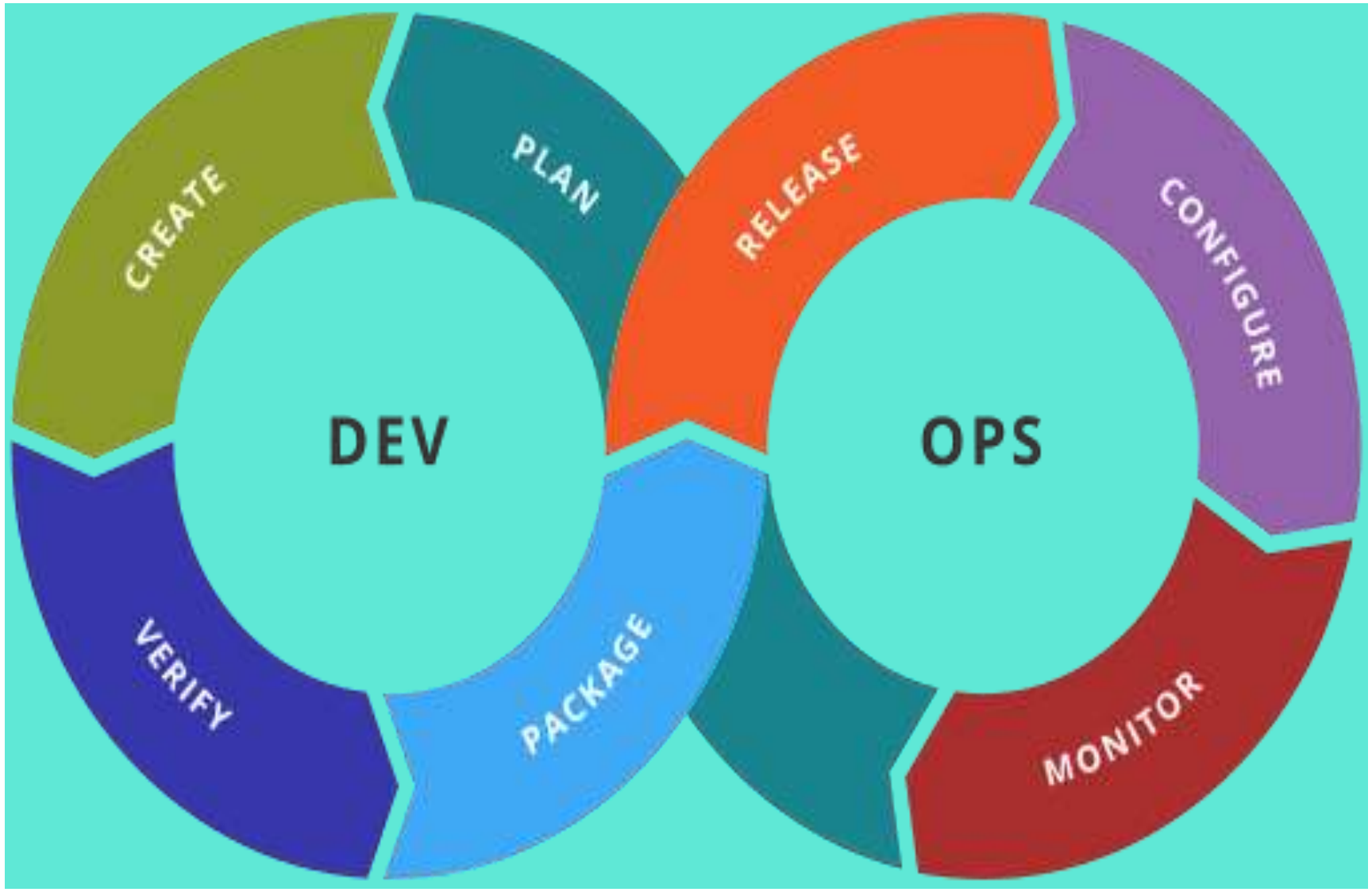
Topics

- DevOps Overview
- DevOps Architecture
- DevOps and Cloud Integration
- DevOps Security
- Types Of DevOps Tools
- Open Source DevOps Tools

DevOps Overview

- In simple words the DevOps means 'It is a combination of software development and operations'.
- It came to emphasize the communication, collaboration, and cohesion between the traditionally separated development and IT operations teams.
- DevOps helps an organization deploy software more frequently, while maintaining service stability and gaining the speed necessary for more innovation.
- 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.
- DevOps is also characterized by operations staff making use many of the same techniques as developers for their systems work.

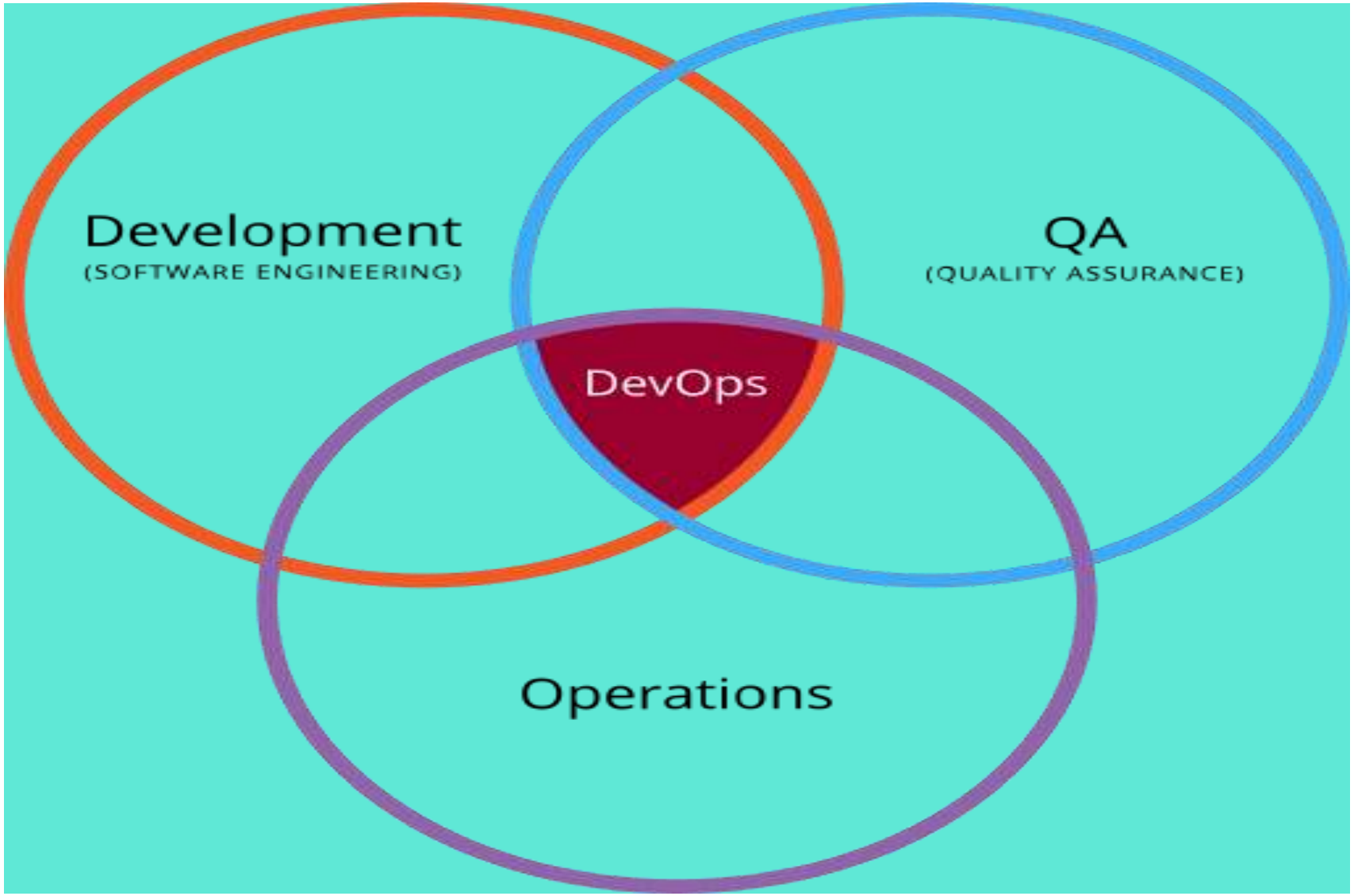
DevOps Overview



DevOps Overview

- DevOps means a lot of different things to different people because the discussion around it covers a lot of ground.
- Agile and DevOps are similar, but, while agile software development represents a change in thinking and practice (that should lead to organizational change), DevOps places more emphasis on implementing organizational change to achieve its goals.
- The need for DevOps was born from the increasing popularity of agile software development, as that tends to lead to an increased number of releases.
- One goal of DevOps is to establish an environment where releasing more reliable applications, faster and more frequently, can occur.
- Release managers are beginning to utilize tools (such as application release automation and continuous integration tools) to help advance this goal—doing so through the continuous delivery approach.

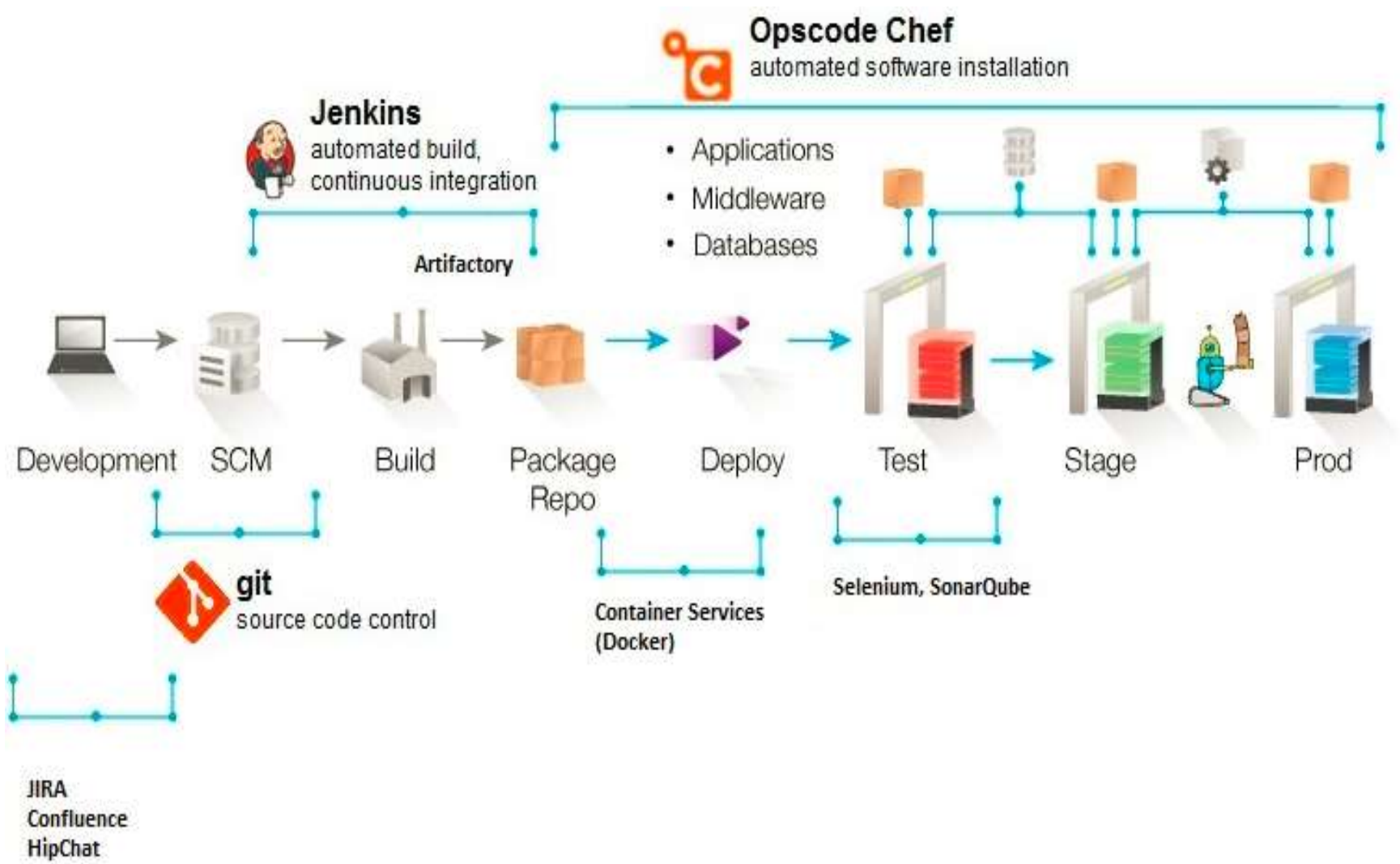
DevOps Overview



DevOps Overview

- A single team composed of cross-functional members and all working in collaboration, DevOps organizations can deliver with maximum speed, functionality, and innovation.
- **Benefits of DevOps:**
  - **Technical Benefits:**
    - Continuous software delivery
    - Less complexity to manage
    - Faster resolution of problems
  - **Cultural Benefits:**
    - Happier, more productive teams
    - Higher employee engagement
    - Greater professional development opportunities
  - **Business Benefits:**
    - Faster delivery of features
    - More stable operating environments
    - Improved communication and collaboration
    - More time to innovate (rather than fix/maintain)

DevOps Architecture



DevOps Architecture

- While designing DevOps Architecture the below points to be considered…
- **Designers must thoroughly understand customer use cases:**
  - Usability, reliability, scaling, availability, testability and supportability are more important than individual features! Quality over quantity is recommended. Designs that anticipate actual customer usage are the most successful.
- **The culture needs to support designers:**
  - Leaders must support the designers with motivation, mentoring and training. No designer can be expected to know everything. It's OK for designers to make some mistakes if lessons are learned and improvement quickly follows. Good DevOps practices including continuous monitoring and quick remediation are examples of practices which help minimize the impact of any mistakes.
- **Supporting tools are needed to realize Design for DevOps practices:**
  - Elastic infrastructures that can be easily orchestrated, created and released as needed to support designers tasks on-demand with minimal delay.
  - Design, code management, monitoring and test tools readily available and scalable with minimal delay.
  - Monitoring tools that track application process performance and report the results to designers in easily consumable formats without delay.

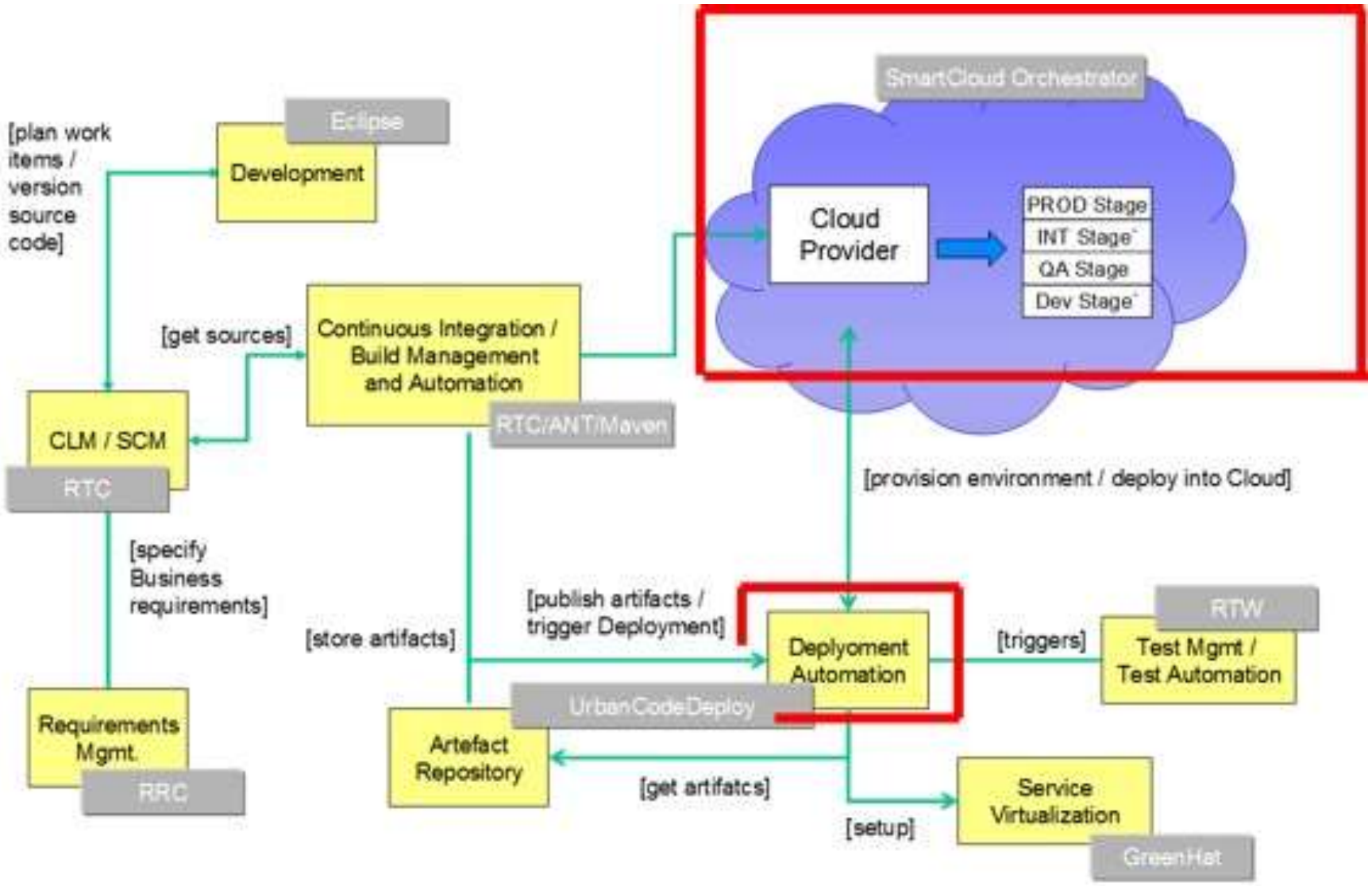
DevOps Architecture

- **Design practices should support QA:**
  - Understand the QA process.
  - Software code changes are pre-checked with unit tests prior to commit to the integration/trunk branch.
  - Software source code changes are pre-checked with Static Analysis tools prior to commit to the integration branch.
  - Software code changes are pre-checked with dynamic analysis and regression tests prior to commit to the integration/trunk branch to ensure the software performance has not degraded.
- **Design practices should support Operations:**
  - Understand delivery and deployment pipeline processes.
  - Software features are tagged with software switches (i.e. feature tags or toggles) during check-in to enable selective feature level testing, promotion and reverts.
  - Automated test cases are checked-in to the integration branch at the same time code changes are checked-in. Evidence that the tests passed are included with the check-in.
  - Tests are conducted in a pre-flight test environment that is a close facsimile of the production environment.

DevOps Architecture

- **Design coding practices are critical:**
  - Products are architected to support modular independent packaging, testing and releases.
  - In other words, the product itself is partitioned into modules with minimal dependencies between modules. In this way the modules can be built, tested and released without requiring the entire product to be built, tested and released all at once.
  - Where possible, applications are architected as modular, immutable microservices ready for deployment in cloud infrastructures, in accordance with the tenets of 12-factor immutable non-monolithic apps, rather than monolithic mutable architectures.
  - Software code changes are pre-checked using peer code reviews prior to commit to the integration/trunk branch.
  - Software changes are integrated in a private environment together with the most recent integration branch version and tested using functional testing prior to committing the software changes to the integration/trunk branch.
  - Developers commit their code changes regularly, at least once per day.

DevOps and Cloud Integration



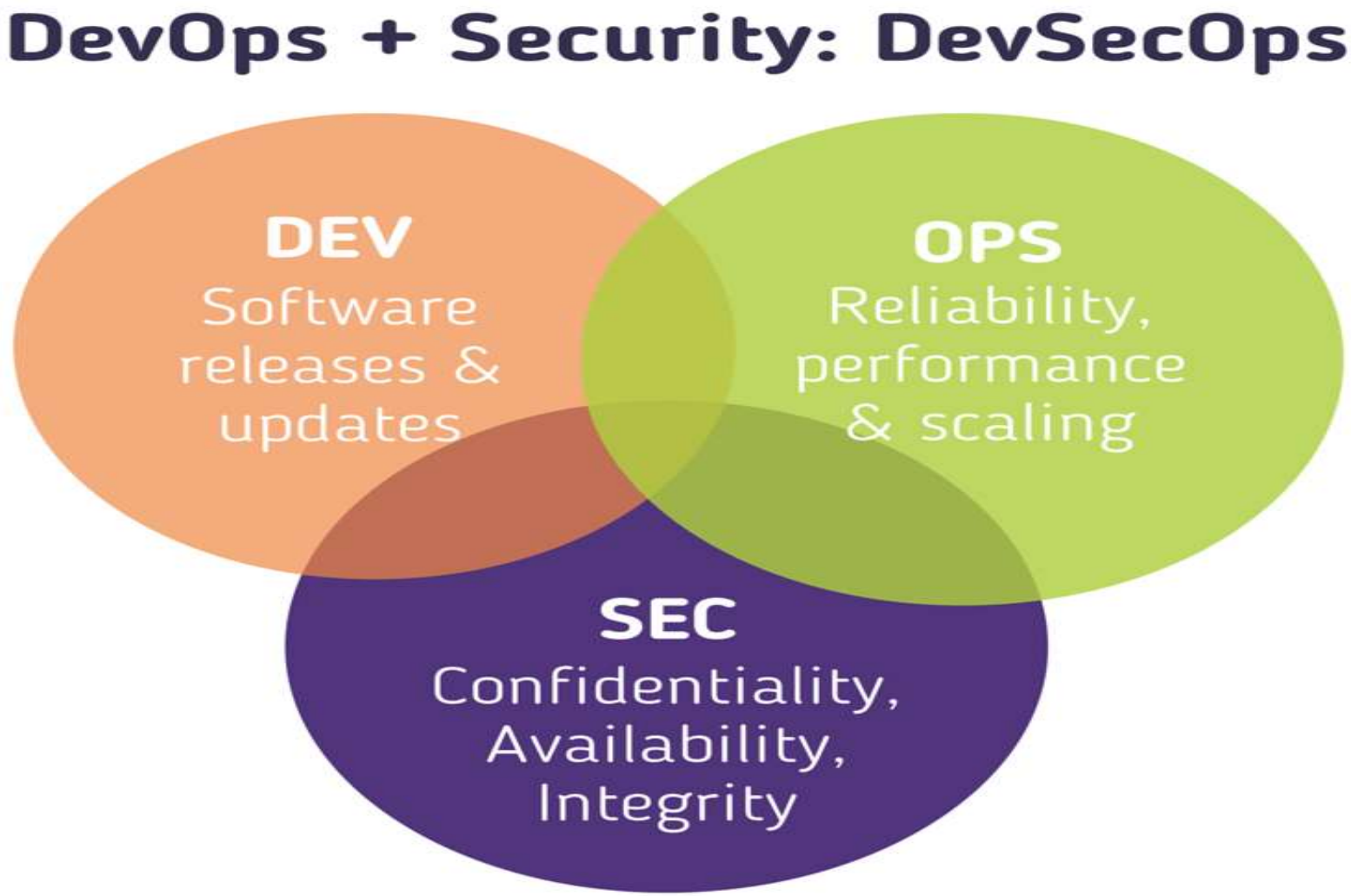
DevOps and Cloud Integration

- Cloud and DevOps are independent but mutually reinforcing strategies for delivering business value through IT.
- From a DevOps perspective, the most important implication of Software-as-Service is the way in which it dissolves the separation between function and operation.
- At the same time as they expect high levels of functional and operational quality, users also expect service providers to deliver continuous change on top of that quality platform.
- Cloud computing, Agile development, and DevOps are interlocking parts of a strategy for transforming IT into a business adaptability enabler.
- Security models change in the cloud, where you'll typically employ identity-based security models and technologies. But you need to extend security to the DevOps tools and organization as well.
- Don't forget about service and resource governance while integrating DevOps with Cloud.
- Choose DevOps tools that work with more than one cloud.

DevOps Security

- To secure the DevOps environment we need to take of following aspects…
- **Manage, secure and control access to privileged accounts:**
  - All privileged accounts in the DevOps environment must be managed and secured.
  - As part of this, credentials should be centrally stored and regularly rotated.
  - To maintain segregation of duties, access to credentials should be controlled so only authorized users have access to privileged accounts.
- **Secure credentials used by applications and scripts:**
  - Credentials used by DevOps solutions to authenticate to and access sensitive resources should be managed and secured.
  - They should be removed from codes and configuration files, securely stored in a central repository and regularly rotated.
  - Additionally, each application instance should have its own unique account and credential.
  - These credentials should be retired as soon as they are no longer needed.

DevOps Security



DevOps Security

- **Authenticate and authorize apps and container access to sensitive DevOps resources:**
  - To keep malicious apps out of the DevOps ecosystem, access requests to different DevOps resources, as well as web requests between application and services, should be granted only after the authentication of the requestor entity (apps and containers) is validated.
- **Enforce Least Privilege:**
  - To reduce the risk of mistakes and privilege abuse without affecting productivity, organizations should limit privilege access and centrally manage privilege escalation.
  - This is particularly important for service accounts: every one that is used by COTS or CI/CD tools should be provisioned.
- **Monitor User Activity:**
  - Tracking developers' activity throughout the software development lifecycle, knowing who built an image and added it to the Registry, and who made changes to privileges, processes, and policies is necessary to maintain control of the environment.

Types Of DevOps Tools

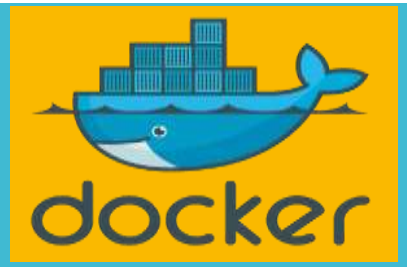
- There are 9 types of DevOps tools which has known before choosing for the project…
- **Collaboration Tools:**
  - This type of tool is crucial to helping teams work together more easily, regardless of time zones or locations.
  - A rapid action oriented communication designed to share knowledge and save time. (See: **Slack, Campfire**).
- **Planning Tools:**
  - This type of tool is designed to provide transparency to stakeholder and participants.
  - Working together, teams can plan towards common goals, and better understanding of dependencies. Bottlenecks and conflicting priorities are more visible. (See: **Clarizen**and **Asana**).
- **Source Control Tools:**
  - Tools of this sort make up the building blocks for the entire process ranging across all key assets. Whether code, configuration, documentation, database, compiled resources and your web site html – you can only gain by managing them in your one true source of truth.(See: **Git, Subversion**).

Types Of DevOps Tools

- **Issue Tracking Tools:**
  - These tools increase responsiveness and visibility.
  - All teams should use the same issue tracking tool, unifying internal issue tracking as well as customer generated ones. (See: **Jira** and **ZenDesk**).
- **Configuration Management Tools:**
  - Without this type of tool, it would be impossible to enforce desired state norms or achieve any sort of consistency at scale.
  - Infrastructure should be treated exactly as code that can be provisioned and configured in a repeatable way. (See: **Puppet, Chef, Salt**).
- **Database DevOps Tools:**
  - The database, obviously, needs to be an honored member of the managed resources family. Managing source code, tasks, configuration, and deployments is incomplete if the database is left out of the equation.(See: **DBmaestro**)

Types Of DevOps Tools

- **Continuous Integration Tools:**
  - Continuous integration tools provide an immediate feedback loop by regularly merging code. Teams merge developed code many times a day, getting feedback from automated test tools. (See: **Jenkins, Bamboo, TeamCity**).
- **Automated Testing Tools:**
  - Tools of this sort are tasked with verifying code quality before passing the build. The quicker the feedback loop works – the higher the quality gets, and the quicker you reach the desired "definition of done".(See: **Telerik, QTP, TestComplete**)
- **Deployment Tools:**
  - In an effective DevOps environment, application deployments are frequent, predictable, and reliable.
  - Deployment tools are essential to checking those boxes. Continuous delivery means that applications can be released to production at any time you want in order to improve time to market, while keeping risk as low as possible. (See: **IBM uDeploy, CA Release Automation, XebiaLabs**)

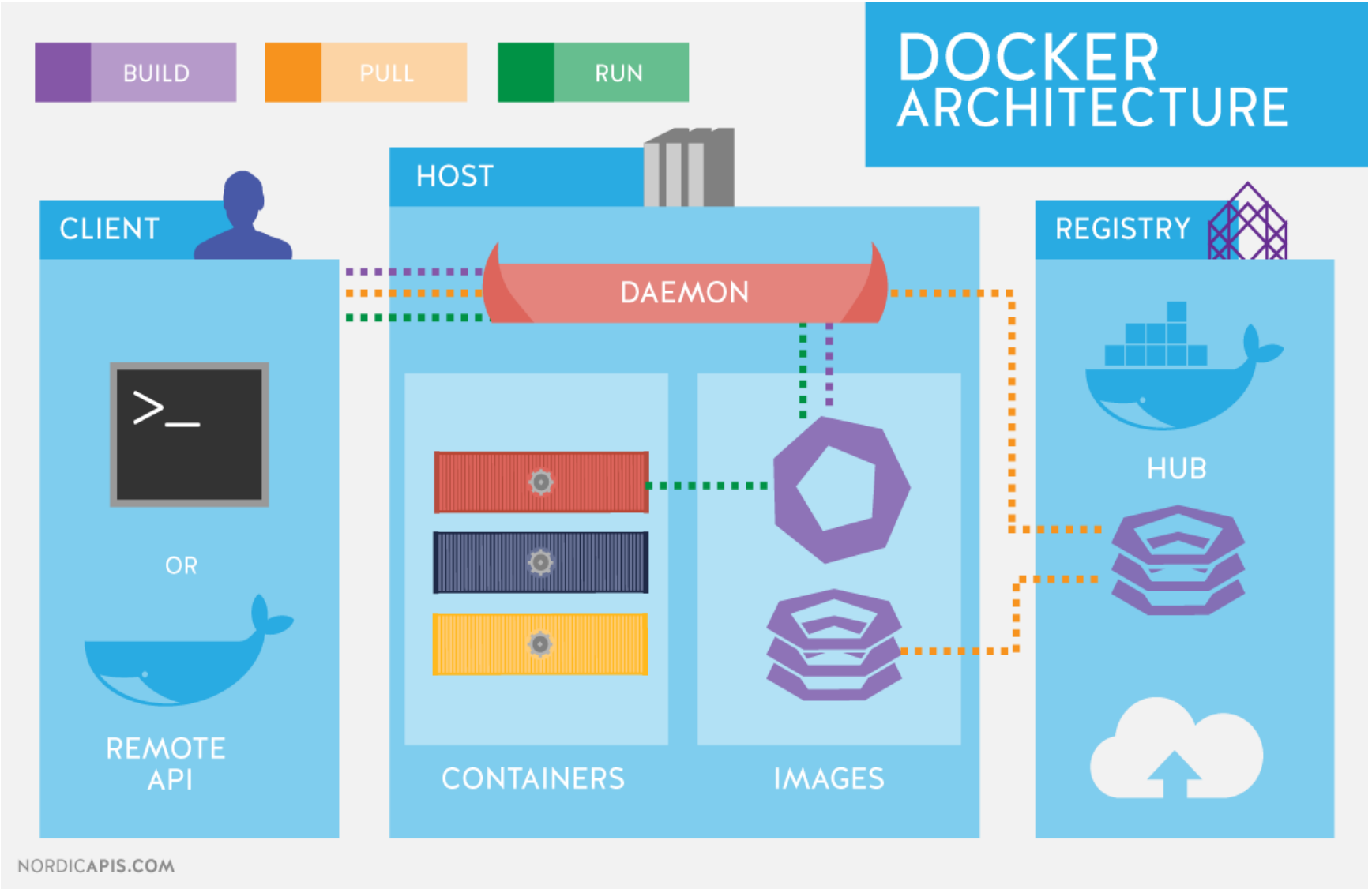


DevOps Tools

Docker

- Docker is at the forefront of the new trend toward containerization.
- It packages together everything that an application needs to run—the code, the runtime, system tools, libraries, etc.—so that applications will operate the same way no matter where they are deployed.
- Containers are more lightweight than virtual machines, and they also offer some security benefits.
- A recent survey conducted by Docker found that 80 percent of enterprises surveyed plan their DevOps implementations around Docker.
- Docker implements a high-level API to provide lightweight containers that run processes in isolation.

DevOps Tools

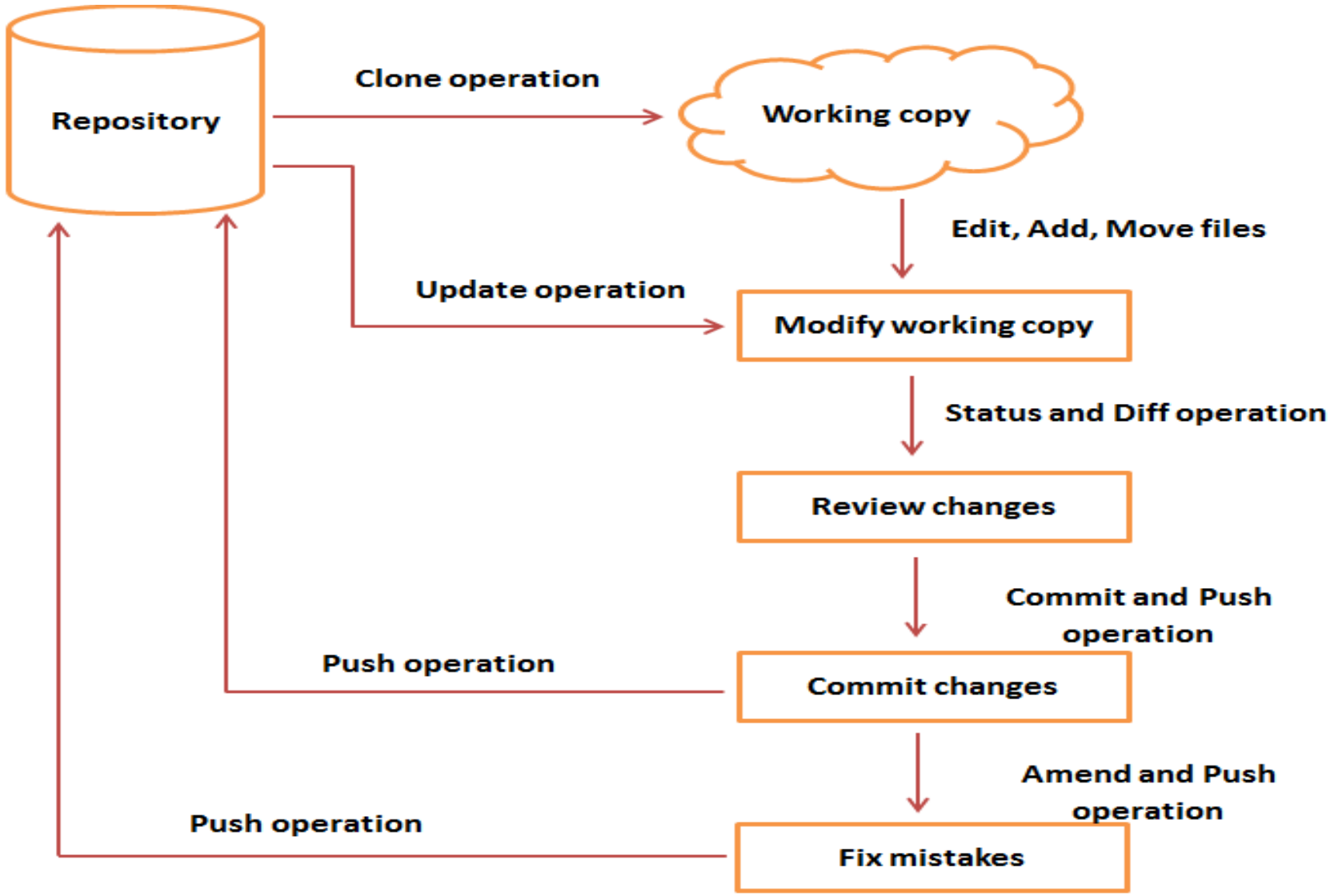


DevOps Tools

Git

- In recent years, Git has become incredibly popular for source code management, particularly as the site GitHub has become more popular for hosting open source projects.
- It stands out from other version control management for the ease with which it handles branching and merging.
- It's also very easy to use with distributed development teams, and it offers fast performance.
- Many DevOps teams use it to manage the source code for their applications.
- Its list of well-known users includes many of the biggest firms in the technology industry, such as Google, Facebook, Microsoft, Twitter, LinkedIn, Netflix, the Linux kernel and many others.

DevOps Tools



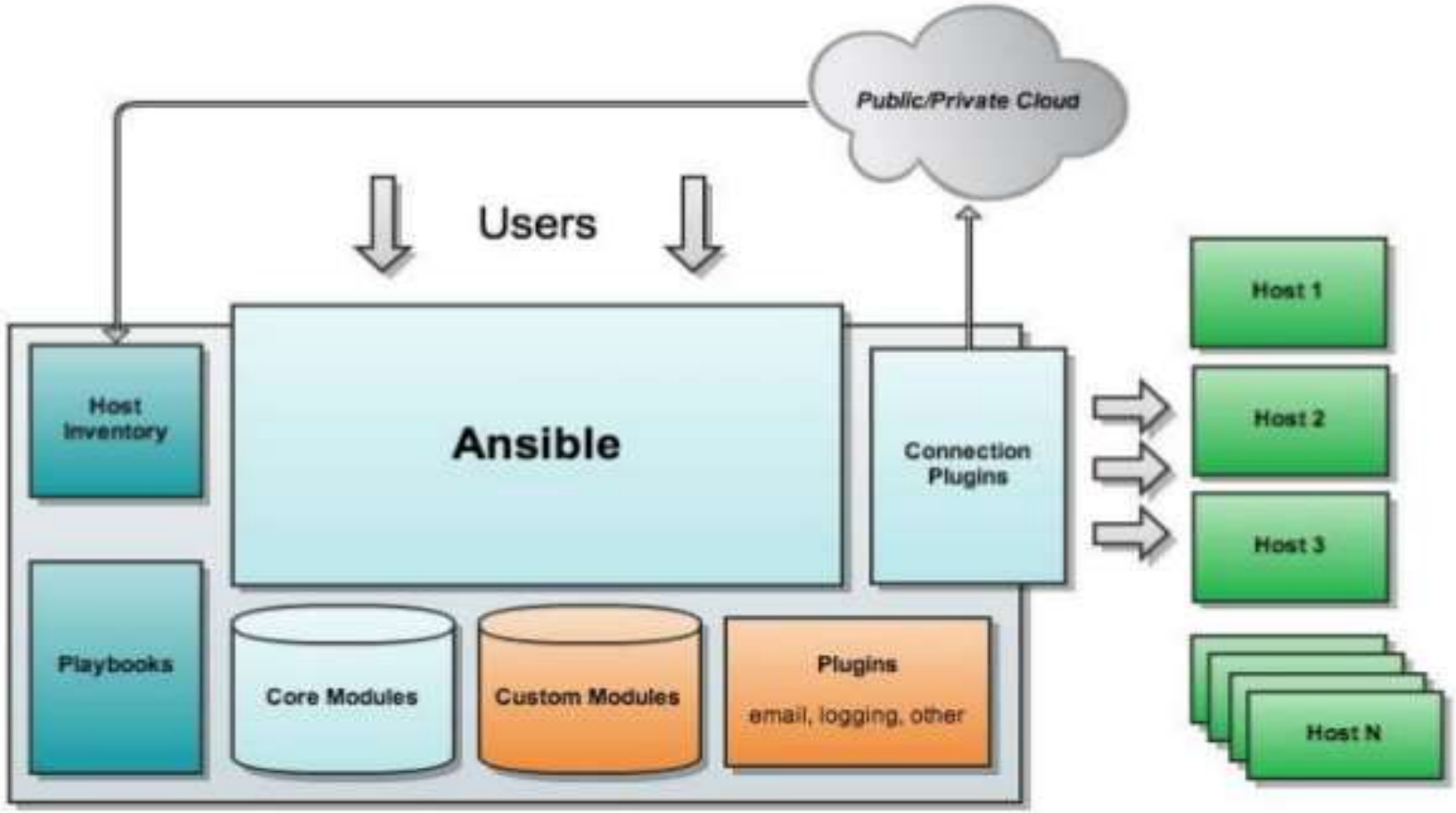
DevOps Tools

Ansible

- Owned by Red Hat, Ansible automates many common IT operations tasks, such as cloud provisioning, configuration management and application deployment.
- It integrates with a lot of other popular DevOps tools, including Git, JIRA, Jenkins and many others.
- The software has been downloaded more than 5 million times, and it has more than sixteen thousand stars on GitHub.
- The free open source version is available on GitHub, and Red Hat offers three paid versions—self-support, standard and premium—with prices that vary based on the number of nodes in production and the level of support needed.
- Ansible has two types of servers: controlling machines and nodes. First, there is a single controlling machine which is where orchestration begins.

DevOps Tools

Ansible architecture

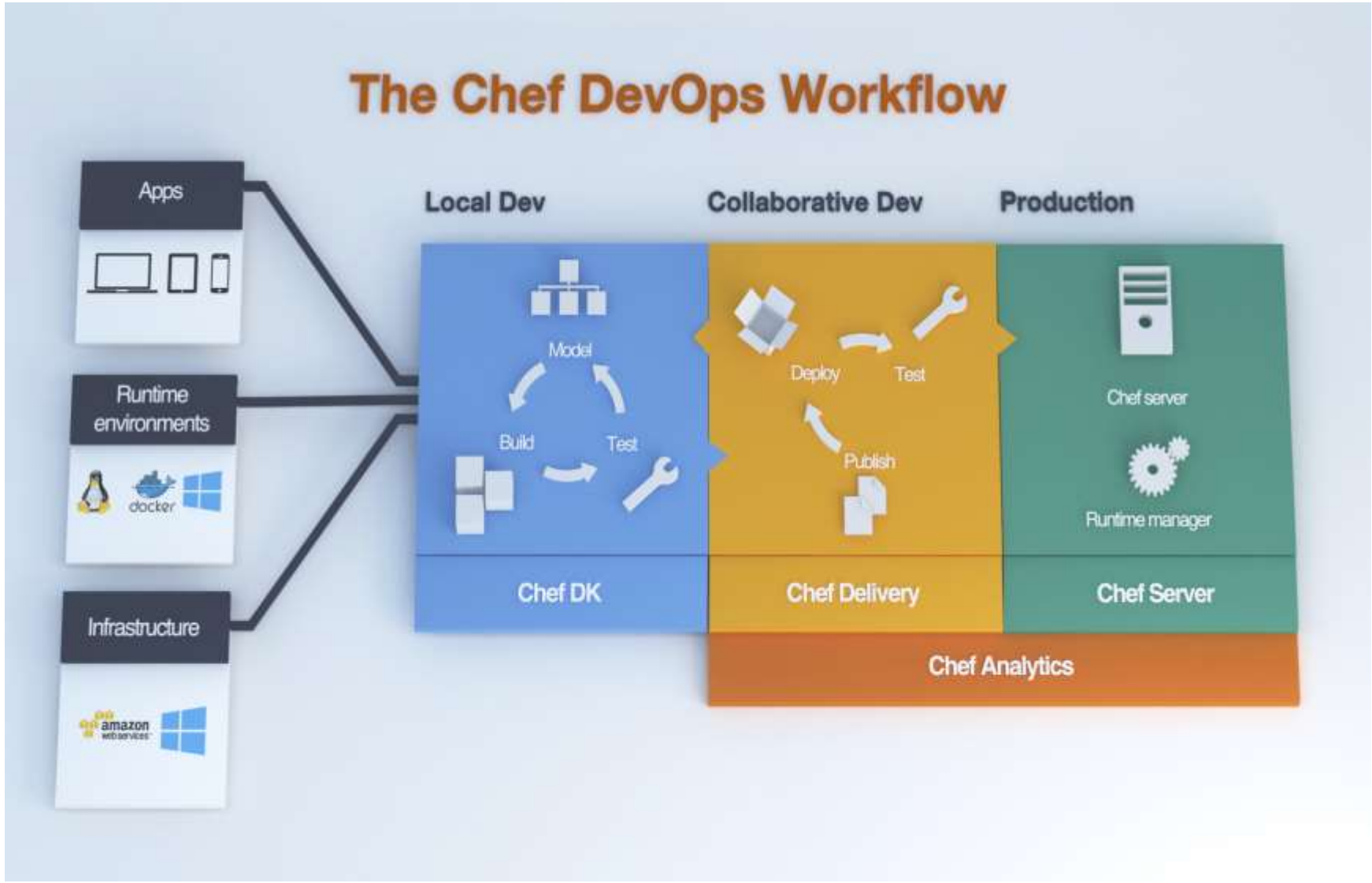


DevOps Tools

Chef

- Another option for infrastructure automation, Chef makes it possible to manage both cloud and traditional environments with a single tool.
- It promises to accelerate cloud adoption while maintaining high availability.
- Quite a lot of documentation and technical resources are available on the Chef site, including many resources designed to help enterprises transition to DevOps and scale their DevOps implementations.
- The company also offers a paid version of Chef called Chef Automate, as well as two other open source projects: InSpec, which focuses on security and compliance, and Habitat, which makes it possible to deploy apps in any environment, including the cloud, bare metal or containers.

DevOps Tools

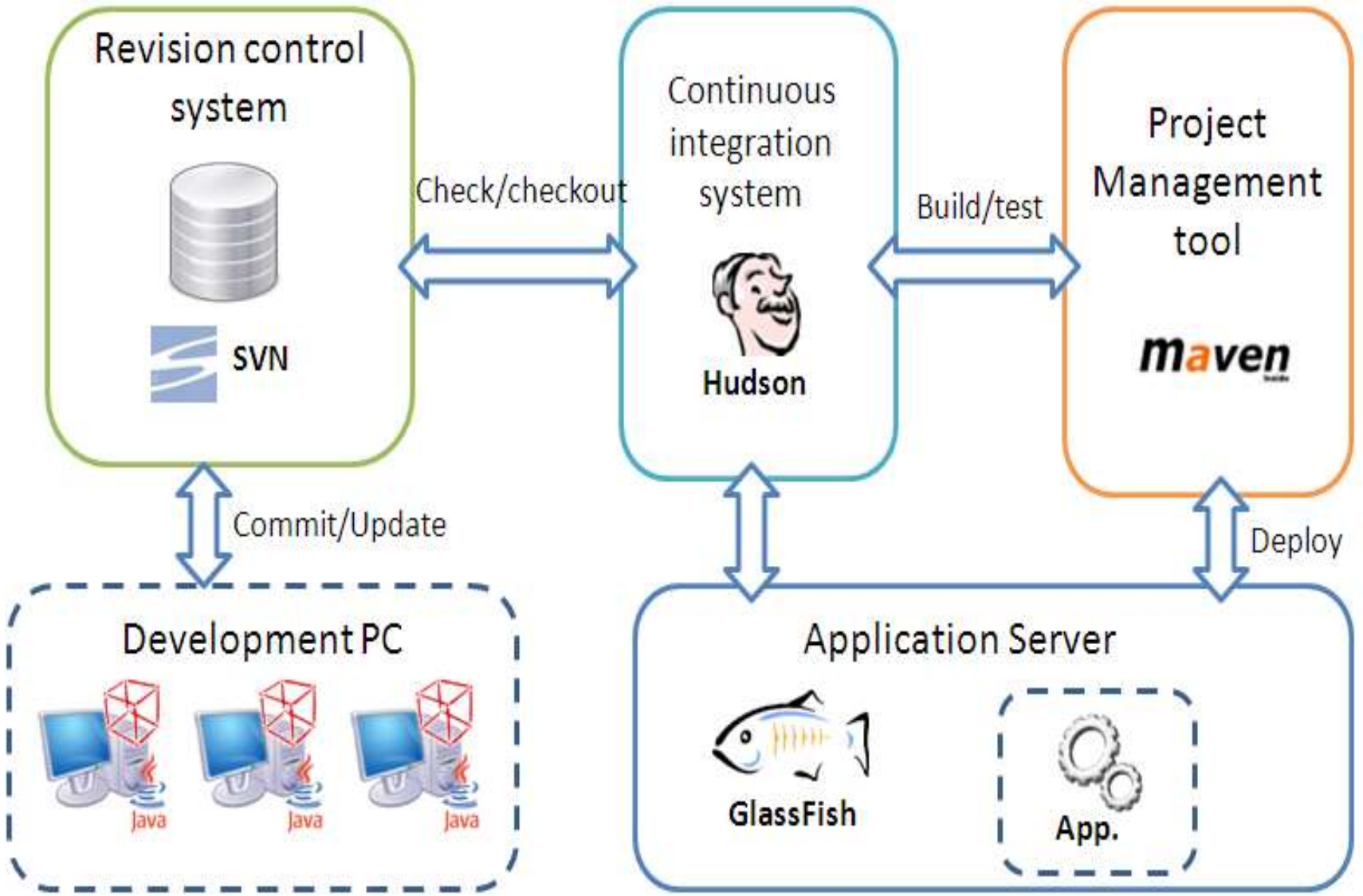


DevOps Tools

Hudson

- When Oracle bought Sun, it declared its intention to trademark the Hudson name, and development began on a commercial version.
- Continuous integration is an integral part of the DevOps approach, and Hudson.
- It is a tool for monitoring and managing continuous integration and testing.
- Its key features include easy installation and configuration, change set support, real-time notifications of test failures, file fingerprinting and support for a wide variety of source code management systems, build tools, testing frameworks, code analysis tools, application servers and other DevOps tools.
- Hudson is managed by the Eclipse Foundation, and there is a huge library of plug-ins that extend its capabilities.

DevOps Tools

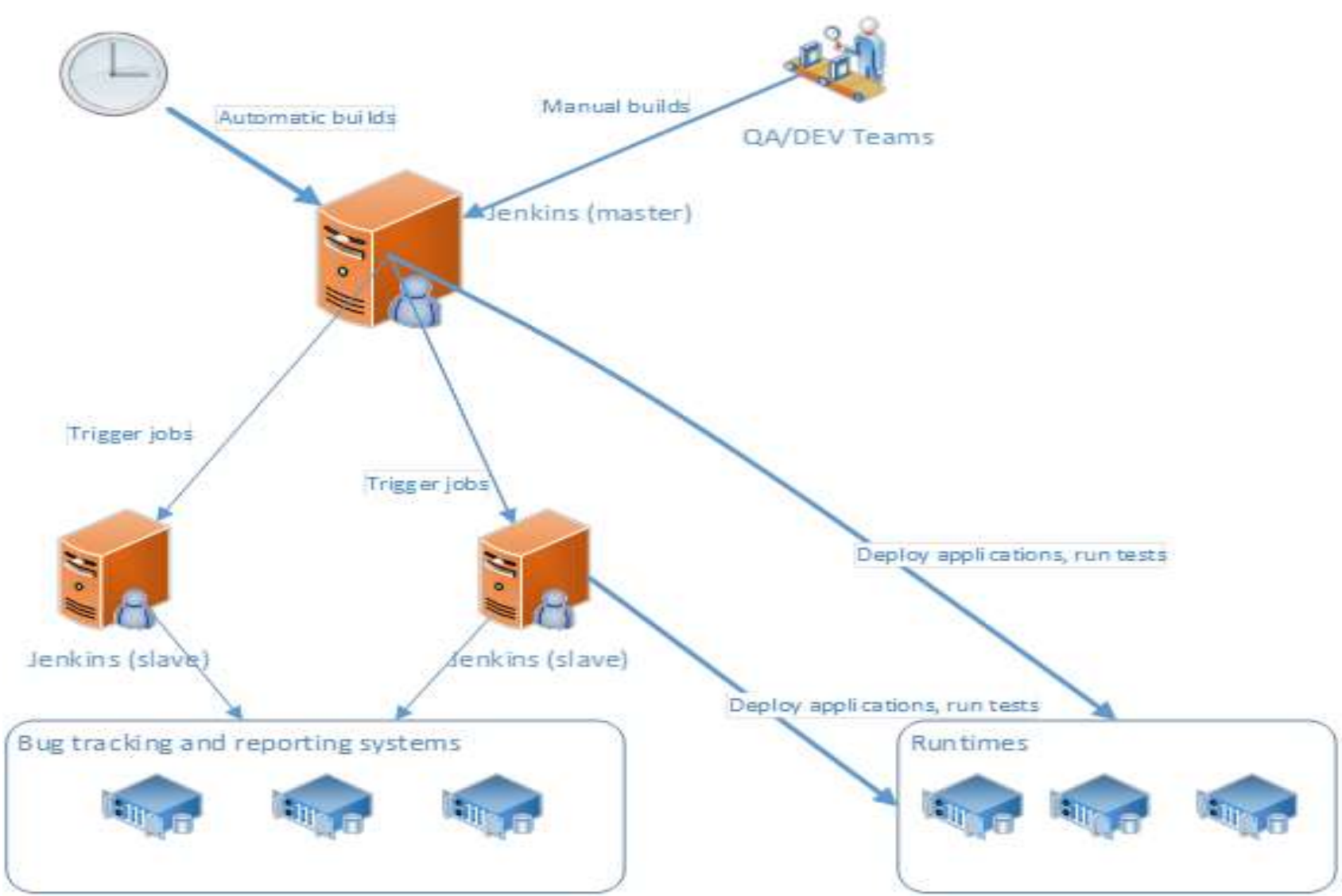


DevOps Tools

Jenkins

- The "leading open source automation server," Jenkins was forked from Hudson and offers many of the same capabilities.
- It boasts easy installation and configuration, hundreds of plugins, extensibility and a distributed architecture that allows it to speed the process of testing.
- It has a very active user community with lots of scheduled events that offer opportunities to learn more about the software.
- There is also plenty of documentation on the website, including a blog that is updated regularly.
- Jenkins Pipeline is a suite of plugins which supports implementing and integrating continuous delivery pipelines into Jenkins.
- Pipeline provides an extensible set of tools for modeling simple-to-complex delivery pipelines "as code".

DevOps Tools



DevOps Tools

OneOps

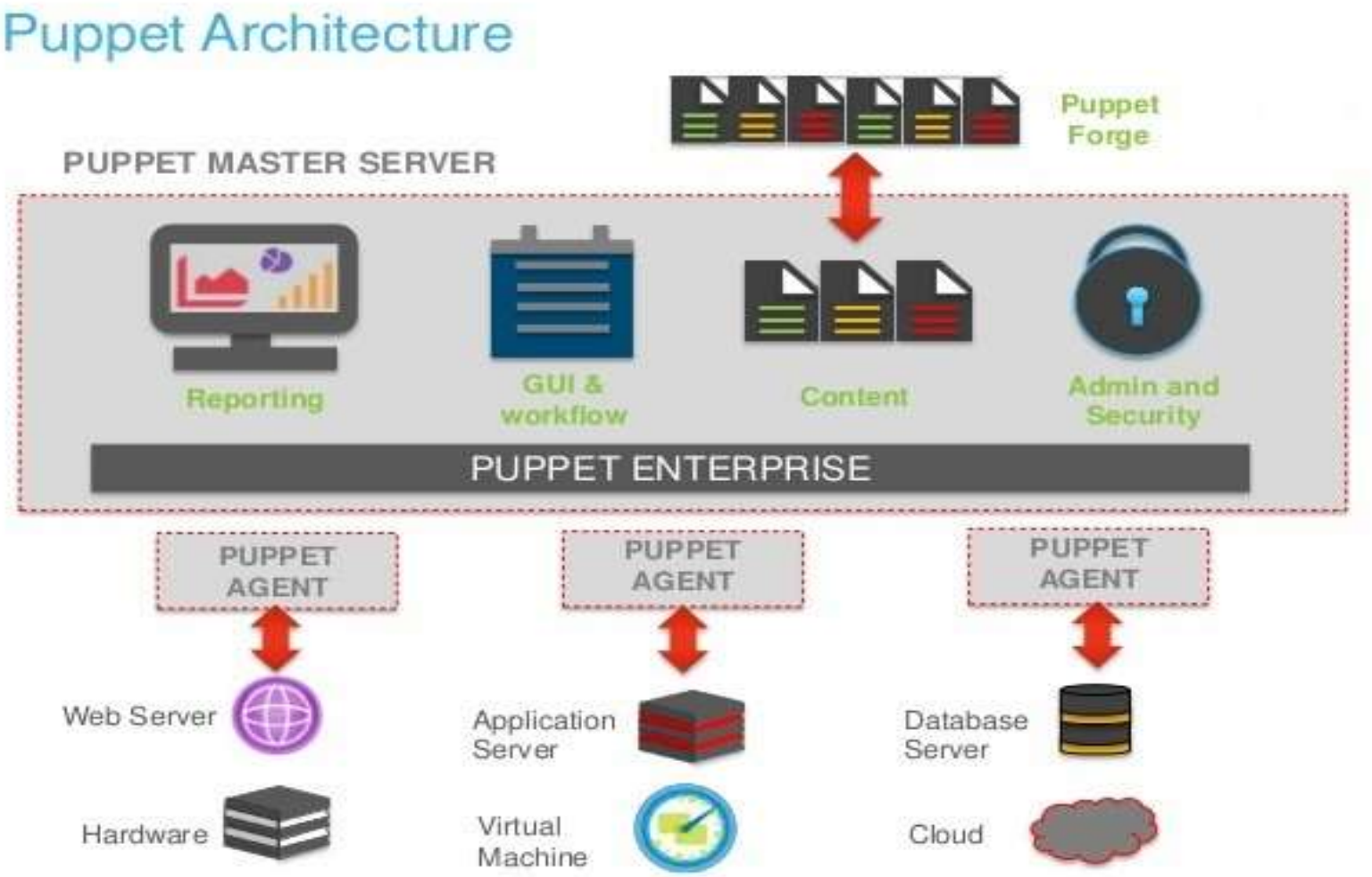
- Released by Walmart Labs as an open source tool earlier this year, OneOps is the newest open source DevOps tool in this slideshow.
- It brings together cloud management and application lifecycle management capabilities with the goal of helping DevOps teams write and launch applications more quickly.
- It also makes it easy to switch among multiple cloud providers, helping prevent vendor lock-in and providing greater flexibility.
- It offers high availability, self-healing and auto-replace capabilities, automatic scaling and integration with many other continuous delivery and automation tools.
- As well as support for all the major public cloud services.

DevOps Tools

Puppet

- Used by more than 30,000 organizations, Puppet promises "a standard way of delivering and operating software, no matter where it runs."
- It automates deployment to boost agility, reliability and auditability.
- Well-known users of the software include the New York Stock Exchange, 1-800-Flowers.com, Getty Images, Staples and many other large organizations.
- The entire Puppet ecosystem includes more than 40 different projects and 3,100 modules are available through the Puppet Forge.
- In addition to the open source version, it also comes in an enterprise version that has free, standard and premium support tiers.

DevOps Tools



DevOps Tools

Salt

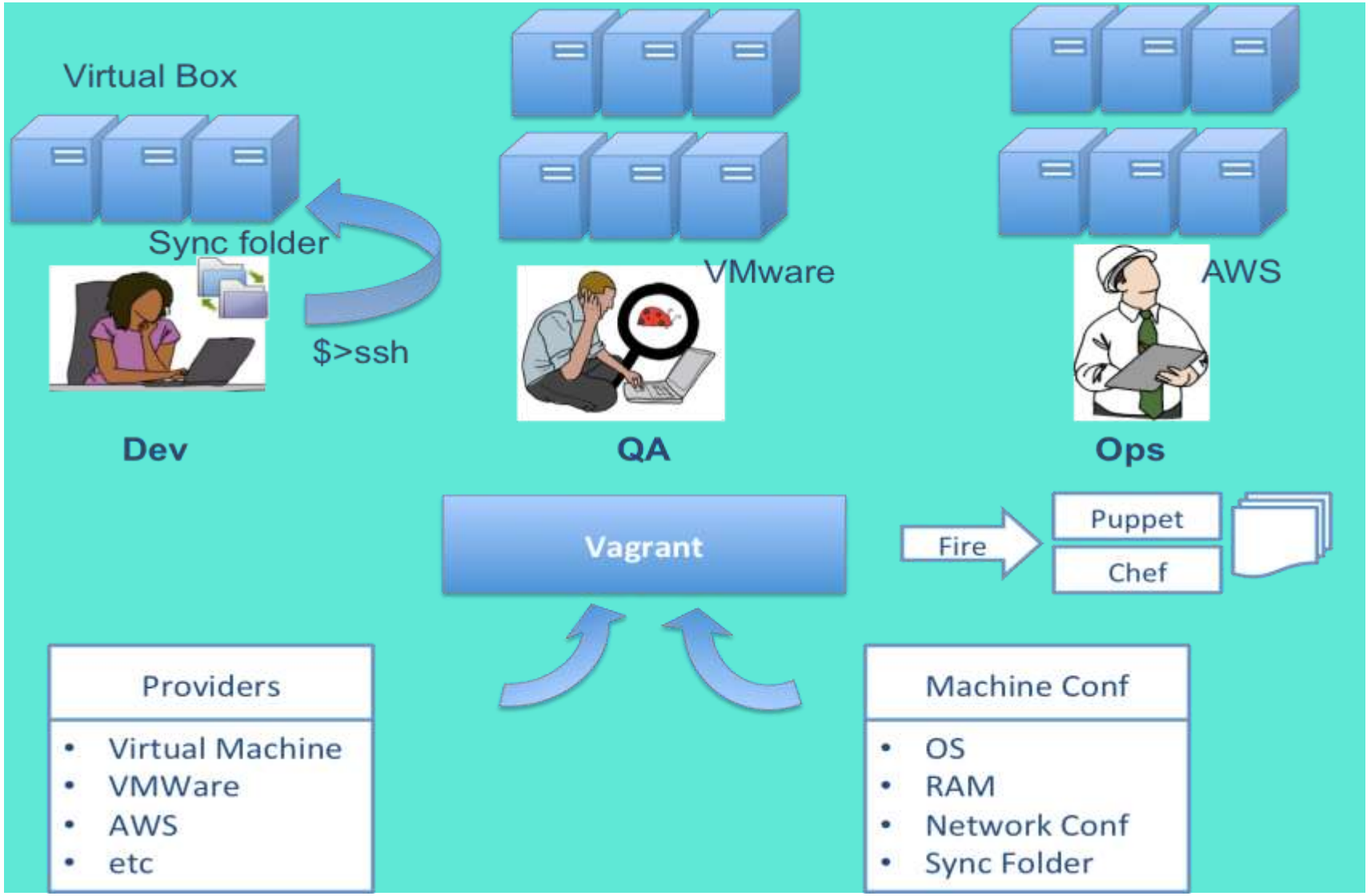
- Another option for IT operations automation, Salt calls itself "the most intelligent, powerful and flexible open source software for remote execution, configuration automation, cloud control and event-driven orchestration."
- First released in 2012, it's used by tens of thousands of organizations.
- It has won numerous awards, including the Best of VMworld 2014 award for virtualization management, an InfoWorld 2014 Technology of the Year Award, and being named a Gartner Cool Vendor in DevOps in 2013.
- The open source version is often referred to as Salt Open, and it also comes in a paid enterprise version called SaltStack Enterprise.

DevOps Tools

Vagrant

- Owned by DevOps tool vendor HashiCorp, Vagrant aims to make it easy to set up development environments that are lightweight, portable and reproducible.
- It's a command-line utility for managing virtual machines. Its users include the BBC, Expedia, Yammer, Mozilla, Nokia and others.
- It integrates with Chef, Puppet, VMware, Amazon Web Services and many other DevOps tools and cloud services.
- Paid VMware plug-ins are available through partners, and HashiCorp offers related paid tools for managing DevOps environments.
- Vagrant manages all the necessary configurations for the developers in order to avoid the unnecessary maintenance and setup time, and increases development productivity.
- Vagrant uses "Provisioners" and "Providers" as building blocks to manage the development environments.

DevOps Tools



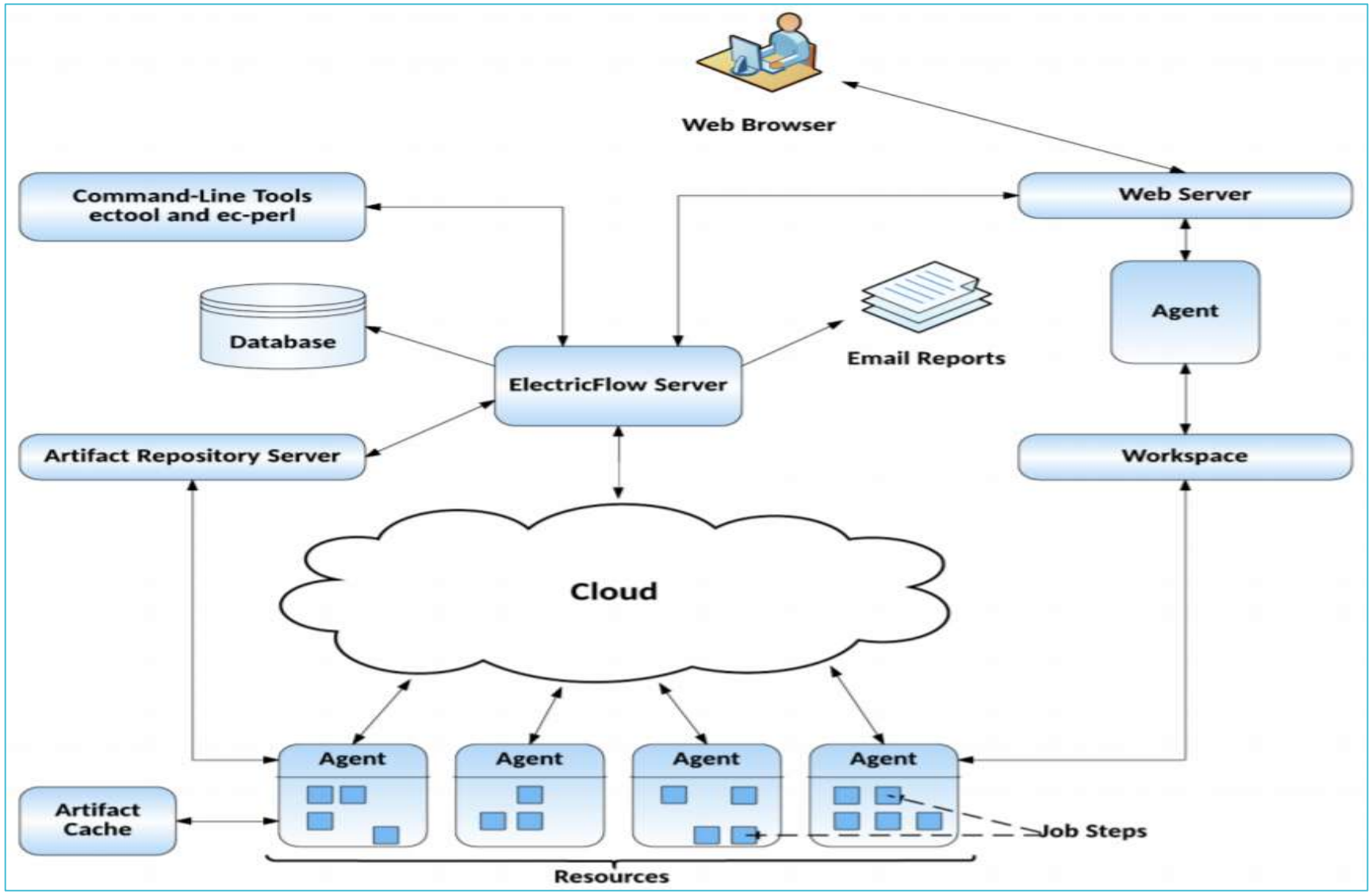
DevOps Tools

ElectricFlow

- ElectricFlow is a single DevOps platform that supports the entire end-to-end software development and delivery process.
- ElectricFlow Deploy is built on top of this powerful platform to automate deployments to accelerate time to market, reduce delivery costs, and increase quality, reliability and traceability.
- Model, automate and scale application deployments to keep up with the pace of the dynamic marketplace.
- Reduce costs and improve productivity and predictability by standardizing Dev and Ops toolchains.
- Orchestrate Docker containers and microservices-based application releases with ease.

Electric Cloud

DevOps Tools



THANKS

If you feel it is helpful and worthy to share with other people, please share the same ☺