Unidad 2 / Escenario 4

Lectura fundamental

Uso de contenedores en DEVOPS

Contenido

Palabras clave: desarrolladores, Quality Assurance, tecnología de la información, Docker.

1. DevOps

1.1. ¿Qué es?

DevOps es la aplicación de culturas de desarrollo, prácticas y herramientas que incluye una comunicación, colaboración e integración estrecha entre desarrolladores de software y profesionales de la tecnología de información para obtener proyectos y realizar procesos de forma ágil, basado en procesos y herramientas de calidad. DevOps busca la integración entre el equipo de desarrollo, el equipo de pruebas y el sector de operaciones para un flujo de completa armonía, durante todo el proceso de desarrollo de un producto o servicio.

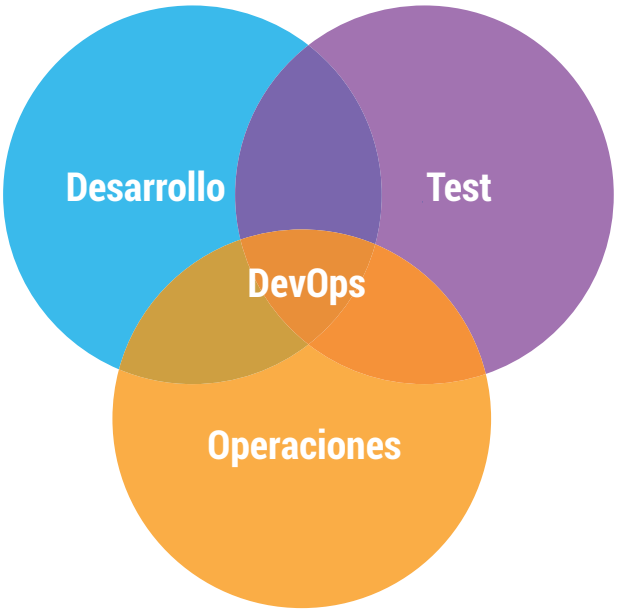

Figura 1. Organización DevOps

Fuente: elaboración propia

1.2. Desarrolladores:

Los desarrolladores son aquellos responsables de diseñar y crear la aplicación en esencia, como producto funcional, DevOps busca una integración más alta por parte de los desarrolladores dentro de las operaciones de la compañía, esto incluye una unión y alineación entre los equipos y un interés a largo plazo por parte de los desarrolladores.

1.3. Control de calidad o Quality Assurance (QA)

El equipo de QA (quality assurance) son responsables de realizar pruebas de contexto para cada plataforma que corra en la aplicación, de ellos depende filtrar los errores, fallos o anomalías que se presente en cada funcionalidad de la aplicación, desde mensajes no convenientes hacia los usuarios hasta “crash”, u operaciones que puedan cerrar la aplicación sin razón alguna.

El equipo de QA debe minimizar la responsabilidad del equipo, a través de la implementación de integración continua y pruebas automáticas, esto ayuda a realizar operaciones repetitivas y a dejar a un lado la suposición de escenarios.

1.4. Operaciones IT (Tecnología de la información)

Operaciones está encargado de mantener y crear toda la infraestructura que envuelve a la aplicación para que esta tenga un correcto funcionamiento. Operaciones como el despliegue monitoreo y manejo de servidores.

1.4.1. Características:

Dentro de las características que esta cultura busca resaltar tenemos: (s.a, s.f)

- **Velocidad:** la integración entre cada uno de los equipos permite mayor velocidad dentro del despliegue de los entregables, de igual forma la constante participación de cada equipo disminuye la realización de operaciones redundantes, debido a que cada persona está alineada con los objetivos y metas, generando un producto de calidad.

¿Sabía qué...?

Amazon libera una versión cada 11,6 segundos de media en ambiente de producción?

- **Entrega ágil:** a medida que se automatizan la compilación y pruebas del producto, es posible entregar resultados en menor tiempo.
- **Confiabilidad:** al aumentar la calidad del desarrollo mediante la automatización de pruebas, cada despliegue tendrá resultados de alta confiabilidad debido a que los errores serán reducidos.
- **Trabajo en equipo y colaboración comunicación:** es importante que todos los departamentos de la compañía estén integrados de manera responsable y colaborativa, se debe tener la idea de unión, no solo con las personas involucradas en el área de desarrollo, sino con cada una de las personas encargadas del proyecto.
- **Mejora continua:** DevOps busca como resultado esta característica, no solo en el proyecto que se está realizando, sino también en cada uno de los procesos de la compañía; cada individuo debe tener como meta la mejora de sus procesos, mejorando la calidad de su trabajo.

2. Jenkins

Jenkins es un servidor de integración continua gratuito que permite la administración de diferentes recursos para mejorar procesos de pruebas, despliegue y administración de resultados, para la integración y el despliegue continuo.

La idea de Jenkins como software base de integración continua, es realizar procesos, definidos como repetitivos, para reducir el desarrollo y horas de producción humana, haciendo uso de la programación de tareas como verificación del estado de los repositorios, generación de builds, la ejecución de pruebas y la generación de reportes de errores.

¿Sabía qué...?

Jenkins posee mas de 1000 plugins para la integración de herramientas y funcionalidades, tanto para el desarrollo web como el desarrollo móvil.

Básicamente esta herramienta integró dentro de una sola herramienta un ecosistema funcional para el desarrollo bajo la integración continua y el despliegue continuo.

Figura 2. Logo Jenkins
Fuente: Jenkins (2017)

Jenkis permite un ecosistema entero de herramientas que ayudan a una adecuada integración bajo cinco principales categorías, basadas bajo el proceso de desarrollo que se presenta habitualmente dentro del desarrollo de software (figura 3), cada operación de mejora y corrección de código pasa por estas 4 fases, desde la fase de desarrollo, hasta la de producción.

Figura 3. Operaciones de realizacion de Cambios
Fuente: elaboración propia

2.1. Code & Commit

Jenkins permite la integración de herramientas para la administración de código y gestión de repositorios; herramientas como git a través de GitHub o Bitbucket entre otros. De igual forma brinda soporte a herramientas de interface de ambiente de desarrollo como virtual studio, eclipse, etc.

Dentro de los principales entornos de desarrollo se tiene:

- Virtual Studio
- Netbeans
- Eclipse
- Xcode (Para desarrollo de aplicaciones para dispositivos de Apple)
- Android Studio

De igual forma, git, como herramienta para la administración de repositorios, permite el uso, mediante herramientas como:

- Github
- Bitbucket
- Gitlab
- Coding

2.2. Build y Configuracion

Jenkins integra de igual forma herramientas para la construcción de proyectos, herramientas como maven para administrar la documentación y entorno de un proyecto, integraciones con amazon, gradle, entre otros. Sin duda alguna, la integración con docker, como herramienta de gestión de contenedores y ambientes virtuales, es uno de los plugins más valiosos que posee este servidor de integración continua debido a su facilidad para la generación de esclavos, e información detallada que puede ser generada en cada construcción o despliegue.

Dentro de las principales herramientas para la construcción y configuración de proyectos, se tiene:

Maven	Visual studio
Gradle	Ibm rational
Docker	Amazon
Chef	Ansible
Gant	

2.3. Scan y Test

Para la realización de entregas automáticas, Jenkis posee herramientas como Fitnesse, JUnit para la ejecución de pruebas automáticas sobre nuestros ambientes de desarrollo. Dentro de las principales herramientas para la realización de pruebas, tenemos:

JUnit	Cucumber
Sonar	Sauce
Gerrit	Retline13.com
BlazeMeter	Arquillian
FitNesse	

2.4. Release

La publicación de nuestras herramientas de forma automática puede ser generada a través de la integración de Jenkins y herramientas como Serena y uDeploy. Dentro de las principales herramientas para hacer release se tiene:

UDeploy
Serena
CollabNet
MidVision
XL Release

2.5. Deploy

Como última fase de desarrollo puede estar controlada por Jenkins con herramientas como docker swarm, amazon webservices, kubernetes entre otros. Dentro de las principales herramientas para la realizacion de Deploy, se tienen:

Pivotal	Kubernetes
Docker	WebSphere
Vmware	Azure
Google Engine	OpenStack

2.6. Docker en Jenkins

Para la administración de contenedores y el despliegue de los mismos, Jenkins permite mediante plugins la integración de los mismos, con demás herramientas dentro del desarrollo de software.

El objetivo de este plugin de docker es poder utilizar el host de docker para, de forma dinámica, proporcionar un esclavo con el fin de ejecutar una sola compilación para analizar los resultados y posterior a esto eliminar el mismo.

Cómo mejorar...

Es importante tener instalado en su servidor de integración continua el plugin JClouds para poder implementar el plugin de Docker.

En caso de que el contenedor haga parte de una red de contenedores es posible analizar el comportamiento de este en conjunto a través de docker swarm. Para configurar este plugin basta con obtener un contenedor que esté en ejecución, establecer las características de configuración en Jenkins para crear un sistema que tenga instalado un servidor ssh y un JDK, crear un usuario para que se pueda iniciar sesión y almacenar la imagen con un identificador conocido.

Referencias

Ángel, E. (2011). ***maquikvit.blogspot.*** Recuperado de: http://maquikvit.blogspot.com.co/2011/08/maquina-virtual-de-proceso.html

Amazon (s.f.) Recuperado de: https://aws.amazon.com/devops/what-is-devops/.

Castro, E. (2014) Recuperado de: https://www.atsistemas.com/es/novedades/opiniones/Opinion-09042014.

Docker. (2017). Recuperado de: https://www.docker.com/what-container

Fuente, T. d. (2015).Recuperado de: https://blyx.com/2015/05/22/buenas-practicas-de-seguridad-en-docker/

Kin (2011). ***Webadicto***. Recuperado de: http://webadicto.net/maquina-virtual-de-sistema-caracteristicas-beneficios-desventajas/

Valdés, B. (2016). ***Osandnet.*** Recuperado de: http://www.osandnet.com/maquina-virtual-caracteristicas-tipos/

Referencias de imágenes

Figura 2. Jenkins (2017) ***Logo Jenkins*** [Imagen]. Recuperado de`https://jenkins.io/

Texto aclaratorio: en este material se ha tomado el logo de la marca Jenkis con fines netamente educativos.

INFORMACIÓN TÉCNICA

FACULTAD DE
INGENIERÍA, DISEÑO
E INNOVACIÓN

Módulo: Énfasis Profesional I (Integración continua)

Unidad 2: Integración continua y manejo de contenedores

Escenario 4: Uso de contenedores en DEVOPS

Autor: Eduardo Enrique Oliveros Acosta

Asesor Pedagógico: Amparo Sastoque Romero

Diseñador Gráfico: Paola Andrea Melo

Asistente: Eveling Peñaranda