Dockers

Dennis Jiménez Campos, Mario Carranza Castillo, Roy Chavarría Esquivel, Álvaro González

*Instituto Tecnológico de Costa Rica: Ing. en Computadores*

[alejimenezc@hotmail.com](mailto:alejimenezc@hotmail.com) , [mariocc0499@gmail.com](mailto:mariocc0499@gmail.com) , [alvgonzalezb@gmail.com](mailto:alvgonzalezb@gmail.com), [roychav99@gmail.com](mailto:roychav99@gmail.com)

**ABSTRACT: For this investigation, our group selected the “Dockers” topic from DevOps. From all the DevOps technologies we thought Dockers was one of the most important ones as its usage in containers establishes “the foundation for DevOps collaboration” [1]. This investigation was made using information from the internet as it’s probably one of the most plentiful sources of information.**

**KEYWORDS: Container, Developers, Operators**

**RESUMEN: Para esta investigación, nuestro grupo seleccionó el tema “Dockers” de DevOps. De todas las tecnologías de DevOps pensamos que Dockers era una de las más importantes ya que su uso en contenedores establece las bases para la colaboración DevOps [1]. Esta investigación fue hecha usando información de la internet, ya que esta es probablemente una de las mayores fuentes de información**

**PALABRAS CLAVE: Contendor, Desarrolladores, Operadores**

I. INTRODUCCIÓN

DevOps (Development Operations) es una de las estrategias de colaboración más importantes para los equipos de trabajo en proyectos programados ya que permite la colaboración entre los equipos de desarrollo (Encargados de generar código) y el de operaciones (Encargado de implementarlo en un entorno específico). Para Esta investigación se abarcará la tecnología Dockers perteneciente a DevOps. Durante la misma se discutirán sus conceptos básicos, aplicación, así como su importancia dentro de DevOps

*A. Justificación*

La tecnología Dockers fue seleccionada debido a su importancia para la realización de proyectos utilizando DevOps, el uso de contenedores facilita la realización de un proyecto en diferentes entornos de programación, lo que permite eliminar la típica excusa de “funcionaba en mi computadora” [2]

*B. Objetivo General*

El objetivo general de nuestra investigación es expandir el conocimiento sobre DevOps, para así formar mejores y más eficientes equipos de trabajo en un futuro.

II. ESTADO DEL ARTE

*A. Desarrolladores*

En un equipo de trabajo, el desarrollador es usualmente quien pone la presión sobre los operadores a la hora de generar productos. En un equipo de trabajo, el desarrollador no depende de nadie para continuar desarrollando código, lo que usualmente se traduce en presión para los operadores.

Un desarrollador de software es “un especialista en informática capaz de concebir y elaborar sistemas informáticos” [2], ellos utilizan varios lenguajes de programación con el fin de cumplir cierta tarea de la forma más eficiente posible

Sabiendo esto, es necesario para el desarrollador entender que el código generado es para solucionar un problema específico, sin embargo, su uso como un todo puede darle problemas a los operadores, quienes trabajan en un ambiente diferente.

*B. Operadores*

En un equipo de trabajo, el Operador debe mantener la estabilidad del producto, así como integrar el paquete de código recibido de los desarrolladores. Los errores son frecuentes durante la realización de estas actividades, lo que obliga al operador a dejar de recibir paquetes de código

Un operador o administrador de sistemas se encarga de mantener todo en orden dentro de una empresa, por lo cual debe dominar los diferentes ambientes de programación. El operador ajusta paquetes de código y se asegura del correcto funcionamiento de este [3].

Sabiendo esto es necesario para el operador el poder comunicarse correctamente con el desarrollador. Ya que este no dejará de enviar paquetes de código mientras ignora cualquier error o problema que se le presenta al operador

*C. Contenedores*

Durante el desarrollo de productos, los desarrolladores y los operadores requieren una forma de conectar o unificar los trabajos de ambos de una forma en la que no se produzcan conflictos.

Los desarrolladores son “dueños de lo que haya dentro del contenedor” [4] mientras que el operador puede concentrarse en sus tareas sin preocuparse por el interior de estos.

Los contenedores son, en resumen, una parte esencial para DevOps y una solución práctica para el problema de conflictos.

III. DESARROLLO

Dockers funciona mediante el uso de contenedores, las cuales se comportan como máquinas virtuales más ligeras que contienen código y todo lo necesario para que este funcione adecuadamente. Gracias a esto su principal ventaja es que “la aplicación funcionará de la misma manera en cualquier servidor como funcionaba en la máquina del desarrollador [5]. Dado que los contenedores están diseñados para que el código pueda funcionar en cualquier computadora sin la necesidad de instalar nada más, se elimina el problema de errores o problemas que suceden en un equipo y en otro no.

Cuando no se esta usando un contenedor este se convierte a una imagen del contender, la cual es un paquete más liviano y ejecutable. Al ser más liviano, se facilita la transferencia del contender entre el desarrollador y el operador. Una vez que la imagen del contenedor se ejecuta en el Docker Engine este se convierte a un contenedor listo para ejecutar la aplicación que trae consigo.

A diferencia de las maquinas virtuales que contiene ciertas librerías, binarios y un sistema operativo de huésped, los contenedores solo contienen la aplicación en sí, todo lo demás necesario para abrir la imagen del contenedor se mantiene en el programa encargado de abrir las imágenes del contenedor, lo cual ayuda a reducir tiempos y tamaño.

Los contenedores solo necesitan ser enviados del desarrollador al operador para que este lo pueda instalar y usar, ya que al contener todas las configuraciones, archivos y librerías necesarios para que el código funcione adecuadamente, el operador no necesita configurar ni instalar nada más para

Mediante el uso de Dockers se permite una mejor comunicación entre el desarrollador y el operador al evitar problemas de compatibilidad que pueden ralentizar la producción, ya que cualquier error o problema puede ralentizar toda la línea de trabajo.

Las maquinas virtuales pueden llegar a tardar varios minutos de ejecución para que puedan ser usados, mientras que los contenedores al tener usos más específicos y delineados son más livianos por lo cual solo necesitan unos cuantos segundos, lo cual, reduce el tiempo perdido esperando a que el programa se ejecute. Este tiempo extra puede no ser muy significativo si se abren 2 o 3 aplicaciones, pero si es un proyecto muy grande que requiera el uso de varias aplicaciones creadas previamente por los desarrolladores, el tiempo acumulado puede llegar a ser bastante significativo, retrasando y dificultando la entrega del trabajo.