UNIVERSIDAD TECNOLÓGICA NACIONAL – FACULTAD REGIONAL CÓRDOBA

DEPARTAMENTO DE SISTEMAS

INGENIERIA Y CALIDAD DE SOFTWARE

Trabajo Conceptual 2 - Charla TED: "Prácticas continuas y formas de despliegue"

Despliegue continuo (CD - Continuous Deployment) y formas de despliegue

CURSO: 4K1

GRUPO: 4

ESTUDIANTES: Aguilera, Guadalupe - 85352

Benegas, Catalina - 85616

Garnero, Constanza - 85648

Lavra, Andrés - 86147

Romero, Manuel - 87210

FECHA DE ENTREGA: 08 de noviembre de 2023

Speech / Guion

Buenos días a todos. Soy Manuel Romero, miembro del grupo 4, y voy a utilizar este tiempo que me brinda la Cátedra de Ingeniería y Calidad de Software de la Universidad Tecnológica Nacional para hablarles sobre una de las revoluciones silenciosas que está transformando la industria del software: **el despliegue continuo.**

Construir software es algo desafiante y complejo. Trabajamos y trabajamos en funcionalidades, pero el momento decisivo y clave es cuando estas funcionalidades son entregadas al cliente. Desplegar el software es todo un espectáculo con fuegos artificiales: mucho estruendo, pero no siempre sale como esperamos.

En un enfoque tradicional, el despliegue es como lanzar un cohete. Se espera a que todo esté perfecto, revisando bien cada detalle, se cruzan los dedos y se lanza la actualización. Rezando que no haya problemas. Y cuando algo sale mal, problemón, nos olvidamos de ponerle nafta.

El despliegue continuo es un enfoque más elegante, más moderno, ágil, en vez de esperar hasta el final para lanzar el cohete y ver qué sucede, los cambios en el código de una aplicación se publican automáticamente en el entorno de producción, funcionalidad tras funcionalidad. Si hay algún fallo, es más fácil corregirlo, porque los desarrolladores tienen más fresca la idea de en lo que trabajaron.

Si todavía no les convence que el despliegue continuo es una buena idea miren empresas como Facebook, Netflix, Amazon, IBM y hasta Google que lo utilizan. Google, incluso nos ofrece una herramienta llamada “Kubernetes” (Kubernetis en yankee) que es muy utilizada en la actualidad para el despliegue continuo.

¿Y cómo se hace para que no explote todo el sistema después de cada cambio, si el despliegue se hace apenas se termina de desarrollar la funcionalidad? Se prueba. Se definen una

serie de pruebas y de forma automática se comprueba si las nuevas actualizaciones pasan con éxito todas estas pruebas. Y si las pasan, listo, el sistema envía las actualizaciones directamente a los usuarios del software.

El despliegue continuo agiliza el tiempo de comercialización al eliminar el desfase entre la programación y la entrega de valor al cliente, que suele ser de días, semanas o incluso meses. Un software que entrega valor más rápido al cliente tiene más calidad y permite a las empresas ampliar su producción en software. Y si bien el costo inicial para lograr un correcto despliegue continuo es alto, se ahorra dinero al evitar pruebas manuales costosas y evitar retrasos.

Muy lindo todo, pero ¿cómo hacemos despliegue continuo? Para esto existen distintas técnicas. Por ejemplo, el despliegue **"All-at-once"** que se utiliza para desplegar un nuevo código de aplicación en un conjunto de servidores existentes reemplazando todo el código en una sola acción de despliegue, requiriendo un tiempo de inactividad de los servidores. En caso de que el despliegue falle se tiene que volver a desplegar el código viejo en todos los servidores, por eso tiene un cierto riesgo asociado.

Otra forma es el **Rolling** o despliegue progresivo en el cual los cambios se van implementando de manera progresiva. Las dos versiones de software, la nueva y la antigua, se ejecutan a la vez para hacer el cambio más suave. También está la técnica **Canary**, una variante del rolling, que consiste en desplegar la nueva versión de software en un porcentaje muy pequeño de servidores al principio, evaluando el comportamiento del nuevo software. Si se comporta bien en esos servidores, entonces se va desplegando de a poco en el resto de los servidores.

De la mano del concepto de la **infraestructura inmutable**, que establece que una vez que algo está funcionando nunca podrá ser modificado, se suma otra técnica que plantea que el despliegue inicie un conjunto completamente nuevo de servidores con una nueva versión de código

de aplicación. De acá surge otra forma de despliegue, el **Green/Blue** que también requiere la creación de otro entorno y que una vez que el nuevo entorno está listo el tráfico se desplaza a este nuevo despliegue, iniciando el estado “verde” del mismo y dejando al entorno viejo inactivo, en estado “azul”, para en casos emergencia, poder volver al mismo.

Con todo lo que les acabé de contar, seguro tienen un lío en la cabeza. Escuchamos muchos términos que parecen ser lo mismo. Integración continua, entrega continua, despliegue continuo… Todo continuo. Pero quiero que quede clara la diferencia entre estos tres conceptos. Imagínense que estamos construyendo un super robot muy inteligente, Chef-Botcito, para preparar postres y tortas.

Primero, la **Integración Continua** sería como si cada vez que alguien tiene una nueva idea brillante para mejorar el Chef-Botcito, esa idea se integra rápidamente. No esperamos hasta que todo esté listo; en su lugar, cada pequeña mejora se fusiona en el proyecto tan pronto como está lista. Es como si el Chef-Botcito estuviera aprendiendo trucos nuevos todo el tiempo.

Con la **Entrega Continua**, imaginemos que hemos entrenado a nuestro Chef-Botcito para hacer un tremendo Lemon Pie, la mejor, o una chocotorta también está buena, o torta oreo ufff, bueno, no importa, cada vez que hay una nueva receta o mejora en la forma en que se hace esa torta, el Chef-Bot entrega esa torta a la puerta de tu casa. Podés decidir si lo aceptás o no a ese cambio, haciendo referencia a que se necesita de intervención humana para su despliegue final, pero siempre está ahí, listo para llegar a tu puerta en cualquier momento.

Finalmente, el **Despliegue Continuo** es como si el Chef-Botcito no solo te entregara la torta, sino que te patea la puerta, entra a tu casa, pasa a la cocina, te pone la torta en la mesa y te sirve una porción sin que tengas que hacer nada. No hay necesidad de que intervengas, la torta mejorada simplemente se despliega y está lista para disfrutar. Entonces podemos decir que, de

cierta forma, el despliegue incluye la integración continua y va un paso más allá de la entrega continua.

Para cerrar, les dejo esta metáfora clarísima: el despliegue continuo es como ese amigo que nunca llega tarde a la joda, incluso si no fue invitado, pero siempre trae el fernet, la coca y el hielo y hasta te trae torta. En el mundo del software, ¿quién no quiere tener un amigo así?

Muchas gracias por escucharme. Si tienen alguna duda, se las respondo rápido porque uso despliegue continuo.

Referencias

- Rossel Sander. (2017). Continuous Integration, Delivery and Deployment, Editorial Packt
- Deployment methods - Practicing Continuous Integration and Continuous Delivery on AWS
- Infraestructura Inmutable. Qué es y por qué deberíamos implantarla en nuestra organización - Adictos al trabajo
- ¿Qué es el despliegue continuo? | IBM
- ¿Qué es el despliegue continuo? | TIBCO Software