Método de automatización del despliegue continuo en la nube para la implementación de microservicios

Conference Paper · May 2018

CITATIONS
4

READS
2,875

1 author:

Fredy Humberto Vera-Rivera
Universidad Francisco de Paula Santander
35 PUBLICATIONS **108** CITATIONS

Método de automatización del despliegue continuo en la nube para la implementación de microservicios

Fredy H. Vera-Rivera [1]

[1] Universidad del Valle, Cali, Colombia
[1] Universidad Francisco de Paula Santander, Cúcuta, Colombia
fredyhumbertovera@ufps.edu.co
fredy.vera@correounivalle.edu.co

Abstract. La tendencia actual en el desarrollo de aplicaciones orientadas a servicios se centra en los microservicios, como una alternativa para construir las aplicaciones uniendo o componiendo una serie de servicios independientes, autónomos, que pueden ser individualmente desarrollados, probados, configurados, desplegados y escalados. En el presente trabajo de investigación doctoral se pretende crear un modelo de especificación de la arquitectura de aplicaciones basadas en microservicios con el fin de modelar, orquestar, adaptar, componer y evaluar los microservicios que hacen parte de una aplicación. Como resultados preliminares se presenta la caracterización del proceso de desarrollo de la arquitectura de microservicios, se propone un método de automatización del despliegue continuo de microservicios; este método se probó usando el deployment pipeline de Bitbucked, obteniendo la posibilidad de realizar pruebas estáticas, unitarias y funcionales automatizadas y desplegar a un servidor de pruebas y de producción automáticamente, sin intervención humana, el despliegue se hace a una instancia EC2 de AWS. También se presenta la identificación de los desafíos y problemas de investigación existentes en el área de los microservicios.

Keywords: Microservicios, especificación, DevOps, automatización del despliegue continuo, deployment pipeline.

1 Introducción

Cada vez es más el enfoque histórico de la arquitectura de aplicaciones monolíticas (en la que se crea una sola aplicación integrada que contiene la mayoría de sus funcionalidades y componentes están muy acoplados), comienza a ser reemplazado por diseños basados en Microservicios. Los microservicios son una nueva tendencia de arquitectura de software que está fuertemente influenciada por la computación distribuida, la idea principal es dividir una aplicación en pequeños servicios; cada servicio debe ser lo más independiente posible de los demás. Cada uno se ejecuta en procesos separados, en un contenedor aislado y único, es posible asignar más recursos computacionales al microservicio que más lo necesite, a diferencia de las aplicaciones monolíticas que asignan recursos a toda la aplicación, que en ocasiones son recursos que posiblemente no serán utilizados y aprovechados en su totalidad [1].

Según Lewis y Fowler un microservicio es: “Un enfoque arquitectónico que consiste en construir sistemas para servicios de escala reducida, cada uno en su propio proceso, que se comunican a través de protocolos ligeros" [2]. Los microservicios se presentan como un enfoque de implementación de la Arquitectura Orientada a Servicios (SOA), que pretende mejorar las desventajas y problemas presentados en SOA. La resiliencia, la escalabilidad, la entrega rápida de software y el uso de menos recursos son características esenciales en los sistemas empresariales actuales. La arquitectura de microservicios llegó a cumplir esas expectativas [3], sin embargo, aún se presentan muchos desafíos como la complejidad de tener que gestionar pequeños sistemas distribuidos, la latencia de la red y la falta de fiabilidad, la tolerancia a fallas, la coherencia e integración de datos, la gestión de transacciones distribuidas, las capas de comunicación, el balanceo de carga, la orquestación, el monitoreo y la seguridad [4].

La automatización de la infraestructura usando prácticas de DevOps reducen el esfuerzo manual involucrado en la construcción, implementación y operación de microservicios, lo que facilita la entrega y despliegue continuo. El gobierno descentralizado y la gestión de datos permiten que los servicios sean independientes y evitan que una aplicación estandarice una sola tecnología. Las arquitecturas de microservicio son especialmente adecuadas para las infraestructuras en la nube, ya que se benefician enormemente de la elasticidad y el rápido aprovisionamiento de recursos [5].

DevOps es un paradigma emergente que permite integrar al equipo de desarrollado con el personal de operaciones, de ahí su sigla Dev (Developers) – Ops (Operations). Sus prácticas permiten liberaciones rápidas y frecuentes [6]. DevOps es un nuevo término que surge con la colisión de dos nuevas tendencias, infraestructura ágil u operaciones ágiles, y colaboración entre el personal de desarrollo y de operaciones a lo largo de todas las etapas del ciclo de vida del desarrollo desde la codificación hasta despliegue a producción [7].

En este trabajo de investigación doctoral se presenta en la sección 2 el estado del arte, en la sección 3 las preguntas de investigación, posteriormente en la sección 4 se plantea la metodología, en la sección 5 se detallan los resultados obtenidos, se caracteriza el procesos de desarrollo de microservicios y se propone un método de automatización del despliegue continuo para la implementación de microservicios.

2 Estado del arte

Di Francesco, P. Malavolta y P. Lago (2017), evalúan el estado del arte de los microservicios desde las siguientes tres perspectivas: tendencias de publicación, foco de investigación y potencial para la adopción industrial. Afirman que existen pocos trabajos que investigan patrones de diseño y lenguajes arquitectónicos para microservicios. Proponer un lenguaje arquitectónico para microservicios podría ayudar a los

arquitectos en muchas actividades. Existen vacíos de investigación en las áreas relacionadas con atributos de calidad [5].

Hulya V, Murat K y Sinem G (2017) realizan una revisión de literatura para conocer las tendencias de los microservicios, los estándares emergentes y las posibles lagunas de investigación. Concluyen que "los microservicios son un tema de tendencia y la predicción es que veremos una tendencia creciente en el futuro cercano. No hay investigación específicamente dirigida a los puntos débiles de los microservicios, como las transacciones distribuidas" [8]. Las principales tendencias o motivaciones de investigación están relacionadas con el diseño y la funcionalidad seguida de las técnicas de rendimiento y de prueba [8].

N. Alshuqayran, N. Ali y R. Evans (2016), realizan otro estudio, enfocándose en la identificación de desafíos arquitectónicos, los diagramas / vistas arquitectónicos y los atributos de calidad de los microservicios. Observan que el rastreo es uno de los problemas más comunes que enfrentan este tipo de aplicaciones. Los gráficos de dependencia ayudan a los arquitectos a refaccionar y tomar decisiones con confianza. Plantean la necesidad de futuras investigaciones en temas como la seguridad, distribución de carga en la nube, integración continua, gestión organizacional y DevOps, así como la automatización de la gestión de contenedores y su despliegue [9].

Claus Pahl y Pooyan Jamshidi (2016), evidencian la falta de tecnologías comprobadas para realizar una arquitectura de microservicios, falta de soporte de herramientas para automatizar y facilitar el uso de microservicios en la nube. Ausencia de herramientas de monitoreo, patrones arquitectónicos, enfoques experimentales para comparar microservicios empíricamente con otros estilos, herramientas para habilitar DevOps, métodos de creación de software (metodologías, patrones de diseño, mejores prácticas, garantía de calidad, revisión de sistemas, gestión de cambios) para el desarrollo de arquitectura de microservicios [10].

Olaf Zimmermann (2017) resalta los puntos críticos en la literatura de microservicios en forma de preguntas prácticas, como resultado de la revisión de la literatura SOA / microservicios y de discusiones con líderes de opinión industrial, desarrolladores y miembros de la comunidad de informática orientada a servicios. Enuncia estos temas de investigación: (1) Diseño de interfaz de servicio (contratación y control de versiones), (2) Ensamblaje y alojamiento de microservicio. (3) Integración y descubrimiento de microservicios, (4) Gestión de dependencias del servicio. (5) Prueba del servicio y de la aplicación de usuario final / cliente [11].

Como trabajos relacionados se encuentran Balalaie, Heydarnoori y Jamshidi (2016) presentan las lecciones aprendidas y buenas prácticas de la migración de una aplicación móvil comercial a microservicios, adoptando DevOps y despliegue en la nube [12]. Villamizar y otros realizan una evaluación del rendimiento de una aplicación monolítica en contraste a una aplicación basada en microservicios desplegadas en la nube [13]. Di Francesco, identifica las propiedades clave en la arquitectura de micro-

servicios, propone MicroArt una herramienta ágil para el diseño de arquitecturas basadas en microservicios [14]. Oberhauser propone Microflows un enfoque para automatizar el modelado y la implementación del flujo de trabajo en un entorno de microservicio dinámico [15]. Xu, Chengzhi y otros presentan CAOPLE un lenguaje de programación para microservicios SaaS [16] y CIDE un ambiente integrado de desarrollo para microservicios usando el lenguaje CAOPLE [17].

3 Preguntas de investigación

Al usar microservicios se puede desplegar una aplicación grande como un conjunto de aplicaciones pequeñas que pueden ser desarrolladas, implementadas, ampliadas, manejadas y monitoreadas de manera independiente. La agilidad, la reducción de costos y la escalabilidad granular, trae algunos desafíos y la complejidad de manejar sistemas distribuidos [13]. En el presente trabajo de investigación doctoral se plantean las siguientes preguntas de investigación: **PI-1:** ¿Qué desafíos y problemas de investigación presenta el desarrollo de aplicaciones que usan microservicios? **PI-2:** ¿Cómo es el proceso de desarrollo de aplicaciones que usan microservicios? **PI-3:** ¿Cómo especificar las relaciones existentes entre los microservicios que hacen parte de una aplicación y qué características debe incluir esa especificación? **PI-4:** ¿Cómo demostrar que el modelo de especificación propuesto proporciona una mejora al proceso de desarrollo de aplicaciones que usan microservicios?

Se han resuelto las dos primeras preguntas de investigación: Para resolver la pregunta PI-1, se hizo una búsqueda en las bases de datos científicas más importantes en el área de la computación como son IEEE y Scopus, se seleccionaron y revisaron un total de 95 artículos. Como resultado se refleja una inmadurez en este campo y una continua evolución, los retos de investigación se centran en la definición del nivel de granularidad del microservicio, en la seguridad, en la orquestación, el monitoreo, en la definición de técnicas, procesos, modelos, herramientas y buenas prácticas a nivel de diseño, implementación y mantenimiento de los microservicios. Para resolver la PI-2 se caracterizó el proceso de desarrollo de aplicaciones basadas en microservicios en el cual las prácticas de DevOps juegan un papel fundamental, resaltando la automatización de pruebas, la entrega continua y el "pipeline" de despliegue continuo como esenciales. Se propone un método de automatización del despliegue continuo que fue probado usando el Bitbucked pipeline, el cual permite realizar pruebas automáticas y despliegue a un contenedor de pruebas y otro de desarrollo en una instancia EC2 de Amazón Web Services.

4 Metodología

A continuación se detalla la metodología que se está llevando a cabo, identificando las fases, métodos, actividades y resultados esperados, se puede apreciar en la fig. 1. Los métodos que se van a usar en cada fase son: revisión sistemática de literatura, análisis de casos de estudio, experimentación práctica, realizar estudio de caso y vali-

dación experimental en la cual se busca comparar el modelo propuesto con otros similares. También se resaltan los resultados a obtener en cada fase y al realizar cada fase se resuelven las preguntas de investigación propuestas.

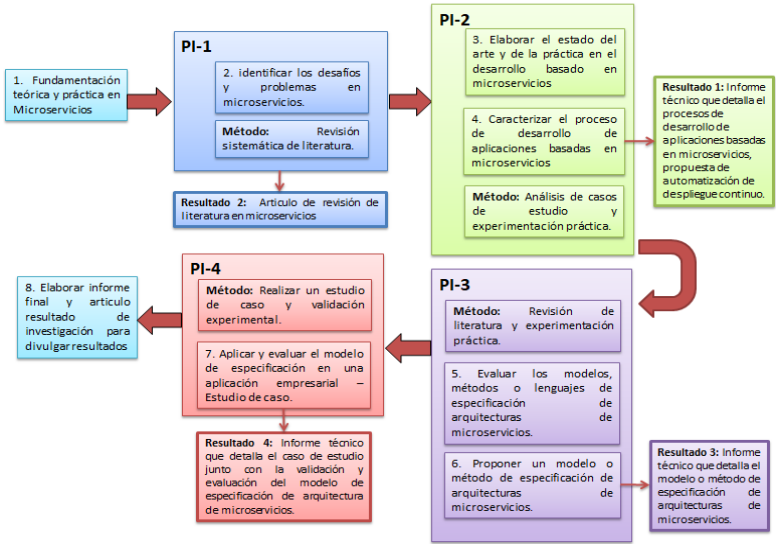

Fig. 1. Metodología de la investigación

5 Resultados

Con este trabajo de investigación doctoral se pretende proponer un modelo para especificar la arquitectura de microservicios, luego a partir de este modelo se podría probar diferentes características como son la calidad del servicio, el rendimiento, las fallas y los costos; de tal forma que el arquitecto o desarrollador pueda encontrar una solución óptima para su implementación. En la fig. 2 se presenta la solución propuesta y la caracterización del proceso de desarrollo de aplicaciones basadas en microservicios.

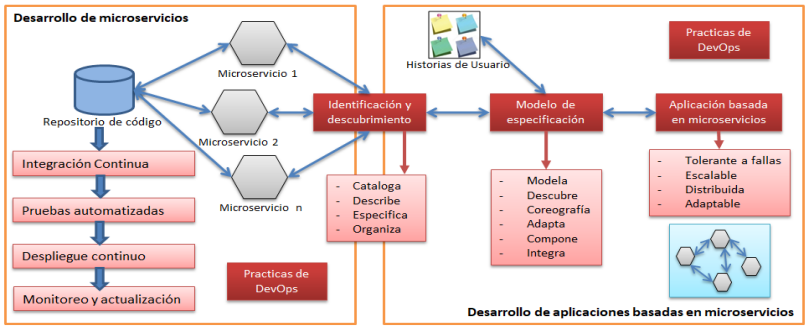

Fig. 2. Solución propuesta – Caracterización del proceso de desarrollo de microservicios

El proceso de desarrollo de aplicaciones basadas en microservicios se divide en dos partes fundamentales, primero el desarrollo de cada microservicio y segundo en el desarrollo de aplicaciones que usan esos microservicios. En la fig. 2 se pueden apreciar las partes fundamentales de ese proceso de desarrollo. El desarrollo de cada microservicio lo realiza un equipo de desarrollo separado de los demás de forma autónoma e independiente, inclusive usando lenguajes de programación diferentes, gestores de bases de datos diferentes y posiblemente ubicados en diferentes partes del mundo. Para cada microservicio se realiza una integración continua, pruebas automatizadas, despliegue continuo, monitoreo y actualización individual. Las prácticas de DevOps son esenciales para el proceso de desarrollo de los microservicios, DevOps tiene como objetivo reducir el tiempo entre el compromiso de un cambio del sistema y colocar el cambio en producción, al tiempo que garantiza una alta calidad [18].

En la fig. 3 se presenta la automatización del "deployment pipeline" propuesta, garantizando integración y despliegue continuo; una vez se confirma (commit) un cambio en el código fuente del microservicio se ejecutan las siguientes acciones automáticas: (1) Análisis estático: se revisa el código fuente para asegurar que cumplen con un estándar, (2) pruebas de predespliegue: se realzan pruebas unitarias, (3) Empaquetar código, (4) Desplegar al contenedor de pruebas (Instancia EC2 de AWS), (5) Aseguramiento de la calidad: se realizan pruebas funcionales, pruebas de carga y rendimiento, todo con el fin de asegurar la calidad del microservicio desarrollado, (6) Desplegar a producción, si pasa todas las pruebas se despliega al contenedor de producción (Instancia EC2 de AWS), (7) Pruebas de post – producción, se ejecutan prueba adicionales para garantizar que la nueva versión funciona adecuadamente en el contender de producción . En caso que se encuentre una falla se cancela el despliegue y se envía una notificación. Este método se basa en lo propuesto por Viktor Farcic en [19]

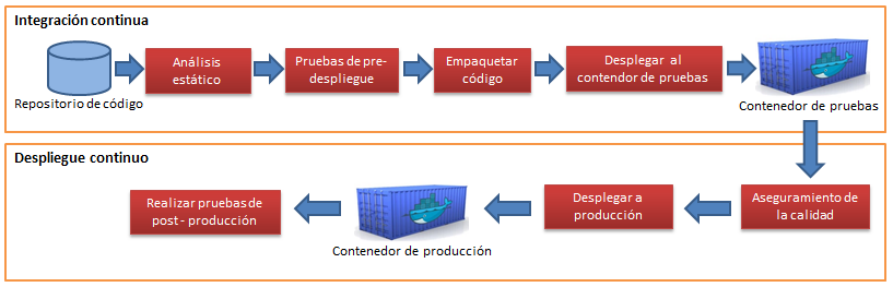

Fig. 3. Automatización del despliegue continúo para la implementación de microservicios

Este proceso de despliegue continuo se realiza para cada microservicio que se esté implementando, se evaluó y probó usando Bitbucked pipeline, se pudo automatizar las pruebas estáticas, funcionales y unitarias de un conjunto de microservicios implementados para un caso de estudio y despliegue automático a una instancia EC2. Los contenedores son fundamentales en el desarrollo de aplicaciones basadas en microservicios. Los contenedores son paquetes autosuficientes que contienen todo lo que necesitamos (con la excepción del kernel), se ejecutan en un proceso aislado y son

inmutables. Ser autosuficiente significa que un contenedor comúnmente tiene los siguientes componentes. (1) Bibliotecas de tiempo de ejecución (JDK, Python o cualquier otra biblioteca necesaria para que la aplicación se ejecute), (2) Servidor de aplicaciones (Tomcat, nginx, otro), (3) Base de datos, (4) Artefacto (JAR, WAR, archivos estáticos, etc.) [19].

La gestión de la configuración y su automatización ahorran tiempo y problemas que surgen a la hora de subir la aplicación a producción. Cualquier persona que administre más de unos pocos servidores puede confirmar que realizar dicha tarea manualmente es una pérdida de tiempo y riesgosa. La gestión de la configuración (CM) existe desde hace mucho tiempo [20]. Ahora con el uso de contenedores y microservicios el número de instancias de desarrollo, prueba y producción se pueden incrementar considerablemente y su administración va ser mucho más tediosa. Herramientas que ayudan a automatizar esta gestión son: CFEngine, Chef y Ansible. La identificación y descubrimiento de servicios son parte fundamental en el desarrollo de aplicaciones que usen los microservicios que fueron creados y desplegados con anterioridad. Este mecanismo permite catalogar, describir, especificar y organizar los microservicios para que puedan ser utilizados. El problema del descubrimiento de servicios es permitir que los consumidores del servicio localicen proveedores de servicios en tiempo real para facilitar la comunicación [20]

6 Conclusiones

El área de investigación en microservicios es relativamente nueva se encuentra en proceso de maduración y evolución. Existen muchos desafíos de investigación por trabajar y resolver. El manejo de seguridad, el nivel de granularidad del microservicio, la orquestación, el monitoreo, la definición de técnicas, procesos modelos, herramientas y buenas prácticas a nivel de diseño, implementación y mantenimiento, son los principales temas y retos de investigación.

El proceso de desarrollo de microservicios se divide en dos partes fundamentales, la primera el desarrollo de los microservicios y la segunda el desarrollo de aplicaciones que usan esos microservicios, siendo las prácticas de DevOps muy importantes. La automatización del despliegue continúo, facilitan las tareas de prueba y despliegue, el uso de contenedores inmutables garantiza tener las mismas características tanto en el servidor de prueba como en el servidor de producción, por lo tanto se reducen considerablemente los errores de las aplicaciones al subirlas a producción. El proceso de despliegue continuo agiliza la puesta en producción de una nueva versión realizando procesos automatizados para probar y garantizar la calidad del software desarrollado.

Referencias

1. J. Thönes, “Microservices,” IEEE Softw., vol. 32, no. 1, 2015.

2. J. Lewis and M. Fowler, “Microservices,” 2014. [Online]. Available: https://martinfowler.com/articles/microservices.html. [Accessed: 13-Sep-2017].
3. T. Salah, M. Jamal Zemerly, Chan Yeob Yeun, M. Al-Qutayri, and Y. Al-Hammadi, “The evolution of distributed systems towards microservices architecture,” in 2016 11th International Conference for Internet Technology and Secured Transactions (ICITST), 2016, pp. 318–325.
4. C. Pautasso, O. Zimmermann, M. Amundsen, J. Lewis, and N. Josuttis, “Microservices in Practice, Part 1: Reality Check and Service Design,” IEEE Softw., vol. 34, no. 1, pp. 91–98, 2017.
5. P. Di Francesco, I. Malavolta, and P. Lago, “Research on Architecting Microservices: Trends, Focus, and Potential for Industrial Adoption,” in 2017 IEEE International Conference on Software Architecture (ICSA), 2017, pp. 21–30.
6. J. Wettinger, U. Breitenbucher, and F. Leymann, “Standards-based DevOps automation and integration using TOSCA,” Proc. - 2014 IEEE/ACM 7th Int. Conf. Util. Cloud Comput. UCC 2014, pp. 59–68, 2014.
7. E. Mueller, J. Wickett, K. Gaekwad, and P. Karayanev, “What Is DevOps? | the agile admin,” 2017. [Online]. Available: https://theagileadmin.com/what-is-devops/. [Accessed: 12-Mar-2018].
8. H. Vural, M. Koyuncu, and S. Guney, “A Systematic Literature Review on Microservices,” Springer, Cham, 2017, pp. 203–217.
9. N. Alshuqayran, N. Ali, and R. Evans, “A Systematic Mapping Study in Microservice Architecture.”
10. C. Pahl and P. Jamshidi, “Microservices: A Systematic Mapping Study,” Proc. 6th Int. Conf. Cloud Comput. Serv. Sci., no. May, pp. 137–146, 2016.
11. O. Zimmermann, “Microservices tenets: Agile approach to service development and deployment,” Comput. Sci. - Res. Dev., vol. 32, no. 3–4, pp. 301–310, 2017.
12. A. Balalaie, A. Heydarnoori, and P. Jamshidi, “Microservices Architecture Enables DevOps: Migration to a Cloud-Native Architecture,” IEEE Softw., vol. 33, no. 3, pp. 42–52, 2016.
13. M. Villamizar, O. Garcés, H. Castro, M. Verano, L. Salamanca, and S. Gil, “Evaluating the Monolithic and the Microservice Architecture Pattern to Deploy Web Applications in the Cloud Evaluando el Patrón de Arquitectura Monolítica y de Micro Servicios Para Desplegar Aplicaciones en la Nube,” 10th Comput. Colomb. Conf., pp. 583–590, 2015.
14. P. Di Francesco, “Architecting microservices,” Proc. - 2017 IEEE Int. Conf. Softw. Archit. Work. ICSAW 2017 Side Track Proc., pp. 224–229, 2017.
15. R. Oberhauser, “Microflows: Automated Planning and Enactment of Dynamic Workflows Comprising Semantically-Annotated Microservices,” Springer, Cham, 2017, pp. 183–199.
16. C. Xu, H. Zhu, I. Bayley, D. Lightfoot, M. Green, and P. Marshall, “CAOPLE: A programming language for microservices SaaS,” Proc. - 2016 IEEE Symp. Serv. Syst. Eng. SOSE 2016, no. 1, pp. 42–52, 2016.
17. D. Liu et al., “CIDE: An integrated development environment for microservices,” Proc. - 2016 IEEE Int. Conf. Serv. Comput. SCC 2016, no. 0, pp. 808–812, 2016.
18. L. Zhu, L. Bass, and G. Champlin-Scharff, “DevOps and Its Practices,” IEEE Softw., vol. 33, no. 3, pp. 32–34, 2016.
19. V. Farcic, The DevOps 2.0 Toolkit: Automating the Continuous Deployment Pipeline with Containerized Microservices. leanpub.com., 2016.
20. J. Stubbs, W. Moreira, and R. Dooley, “Distributed Systems of Microservices Using Docker and Serfnode,” in 2015 7th International Workshop on Science Gateways, 2015, pp. 34–39.