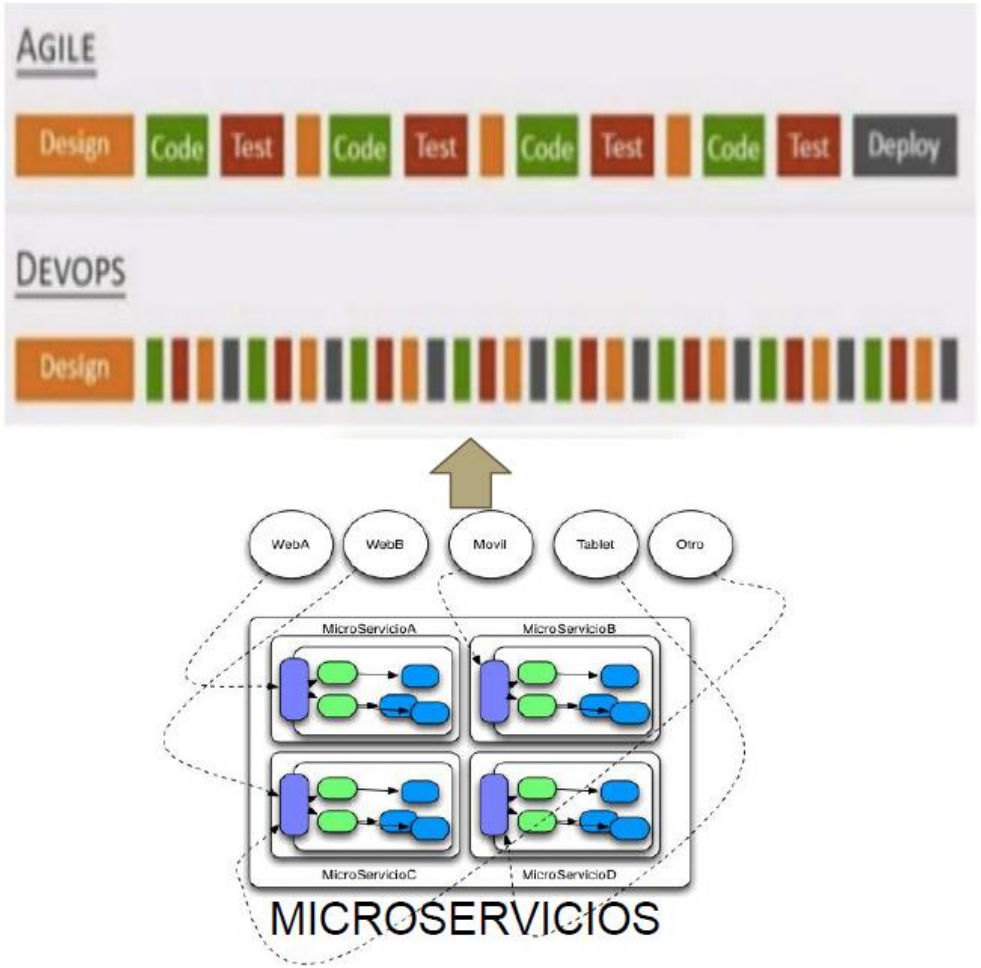
Ingeniería de Sistemas

Desarrollo de aplicaciones basadas en microservicios
2. Arquitectura basada en microservicios

AGENDA

1. Arquitectura de microservicios
2. Evolución de los microservicios
3. Proceso de Desarrollo de microservicios
4. Despliegue continuo
5. Comparativo aplicaciones monolíticas y microservicios
6. Estado del arte de la arquitectura de microservicios
7. Caso de estudio Sinplafut

Arquitectura basada en microservicios



Al aplicar las prácticas ágiles implica una entrega continua y un despliegue continuo para entregar valor al cliente en cada iteración, para hacer esto posible las prácticas de DevOps y los microservicios son fundamentales porque facilitan este proceso.

Microservicios - Definición

"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".

Martin Flower.

Microservicios – Aspectos clave

Hacen una cosa o son responsables de una funcionalidad. Principio de responsabilidad Simple

Cada microservicio puede ser construido por diferente conjunto de herramientas o lenguajes, ya que cada uno es independiente de otros.

Son desacoplados ya que cada microservicio está físicamente separado de los demás.

Independencia entre equipos de desarrollo, cada microservicio es desarrollado por un equipo de desarrollo.

Pruebas más sencillas, entrega y despliegue continuo.

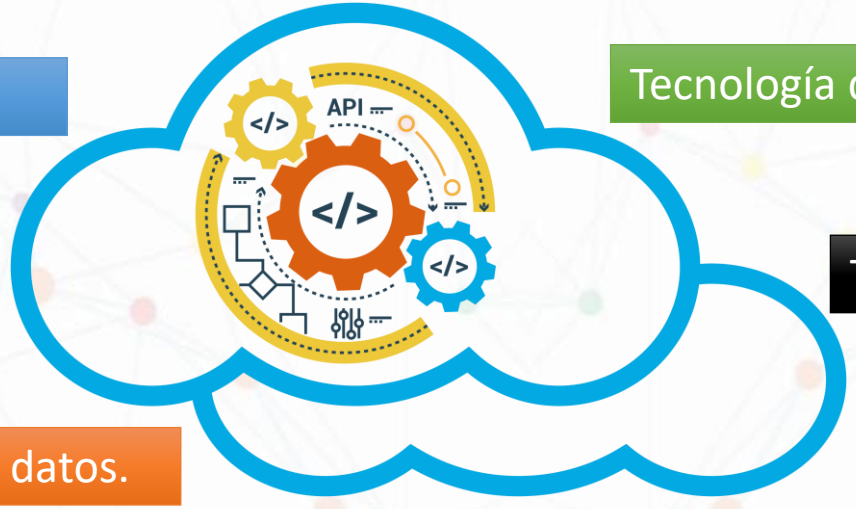
La arquitectura se organiza de acuerdo a capacidades empresariales

Microservicios – Características

Diseño evolutivo.

Servicios autónomos

Gestión descentralizada de datos.

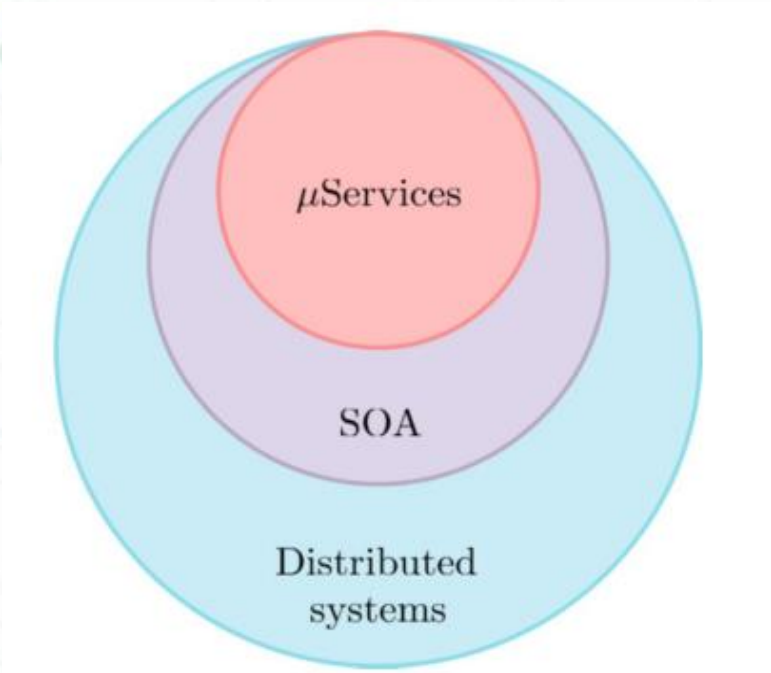


Tecnología de desarrollo heterogénea

Tolerancia a fallos

Automatización de la infraestructura

Arquitectura basada en microservicios



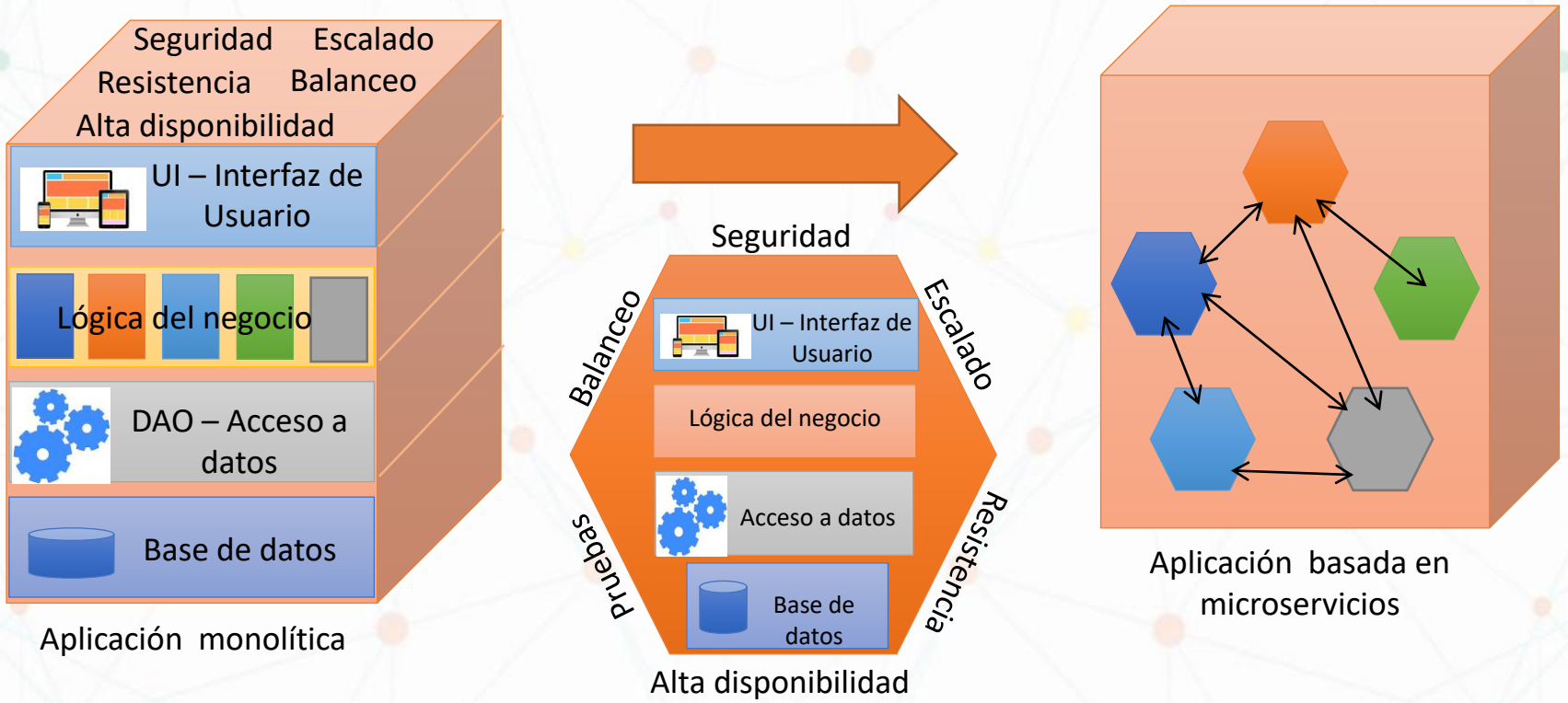
"Overcoming Security Challenges in Microservice Architectures," in *2018 IEEE Symposium on Service-Oriented System Engineering (SOSE)* (T. Yarygina and A. H. Bagge, 2018)

Beneficios:

- Uso de tecnologías heterogéneas
- Resistencia a fallas
- Escalado – Se escala los microservicios que deben ser escalados.
- Despliegue independiente
- Alineamiento organizacional
- Componebilidad y reutilización
- Costo de reemplazo bajo.

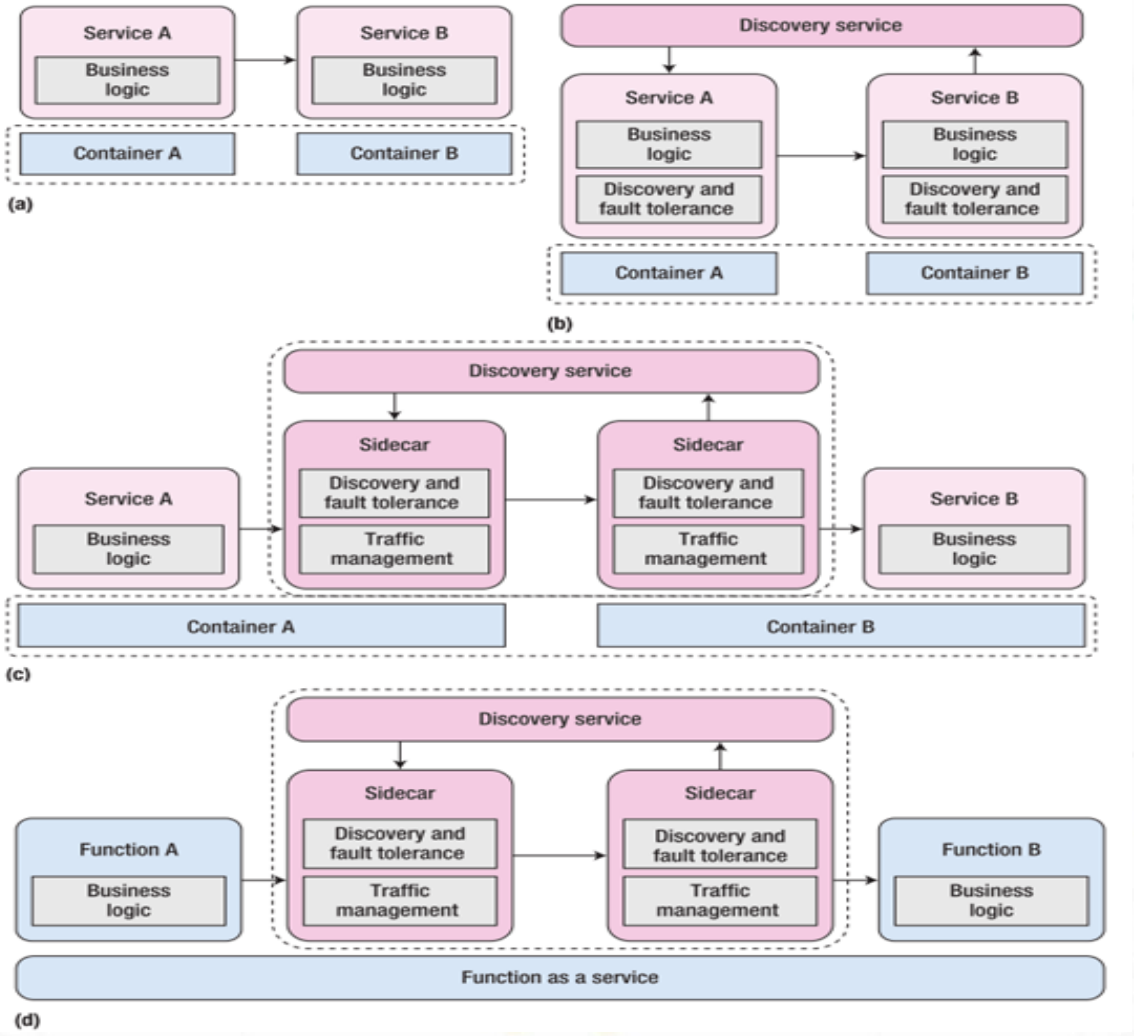
Building Microservices. O'Reilly Media, Inc. (S. Newman, 2015)

Arquitectura basada en microservicios



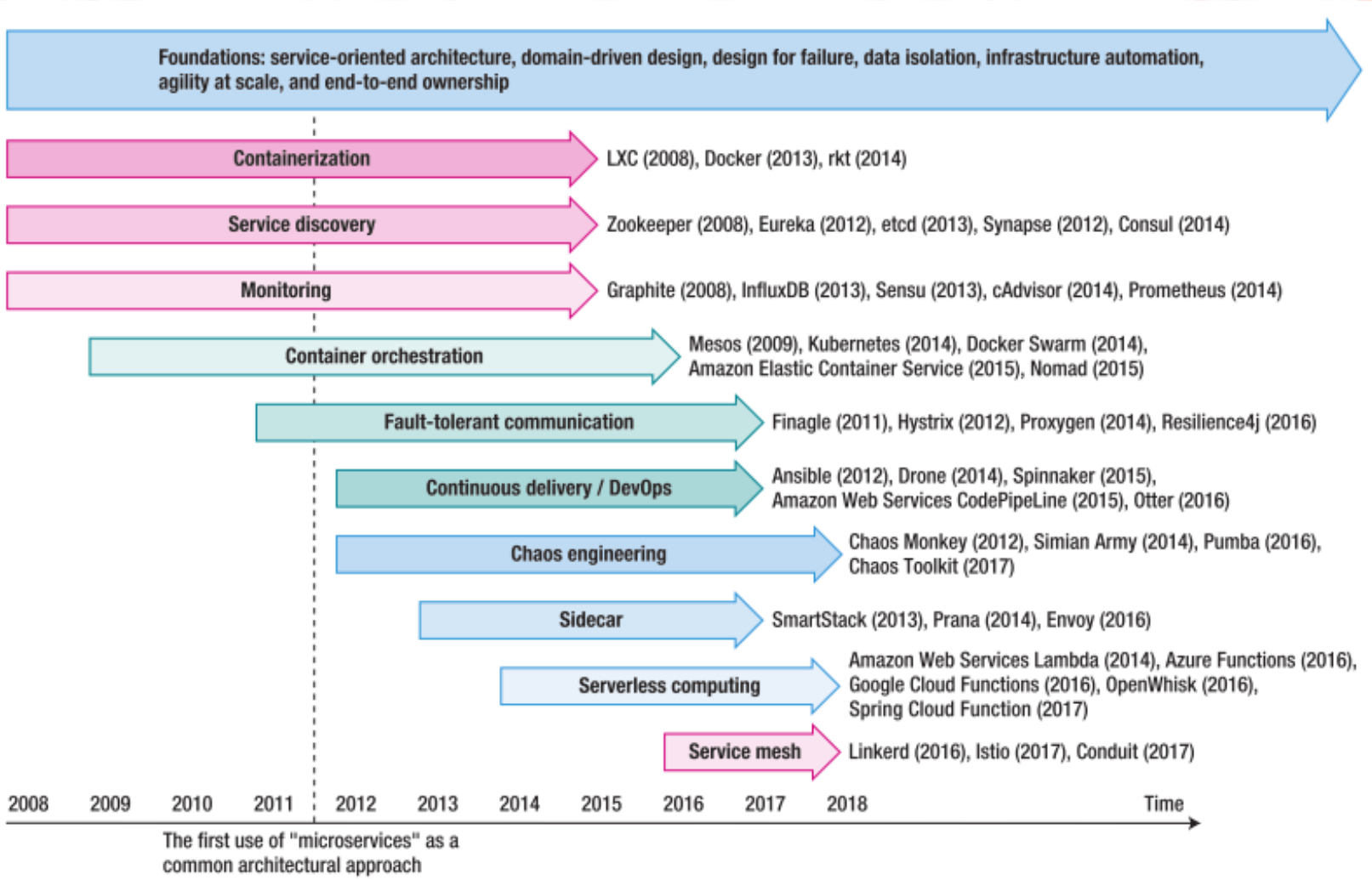
Los microservicio son servicios pequeños y autónomos que trabajan en conjunto. Enfocados en hacer una cosa bien hecha. Siguen el "Principio de responsabilidad simple" que dice "Reúnan las cosas que cambian por la misma razón, y separe aquellas cosas que cambian por diferentes razones" (S. Newman, 2015)

Generación de la Arquitectura basada en microservicios



Generaciones de la arquitectura de los microservicios. (P. Jamshidi et. al, 2018.)

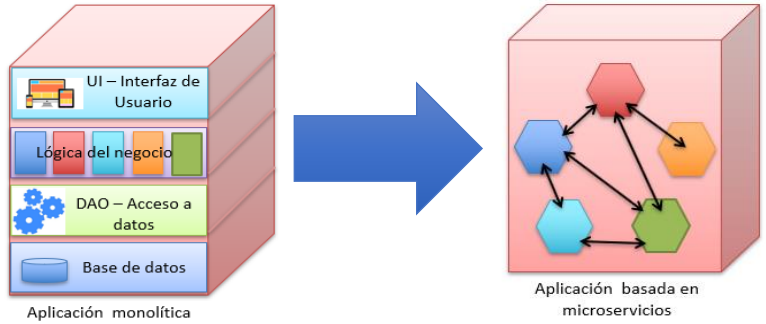
Evolución tecnológica



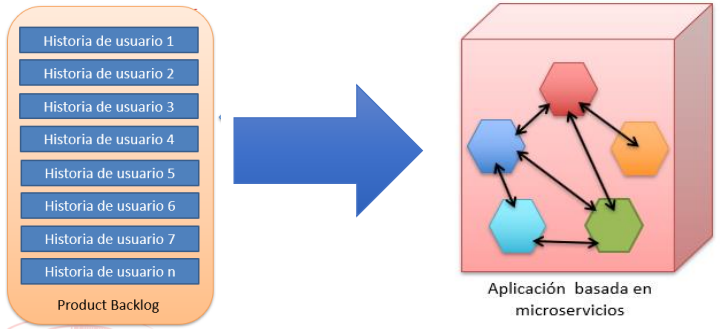
Línea de tiempo de evolución tecnológica de los microservicios. (P. Jamshidi et. al, 2018.)

Enfoque de desarrollo de aplicaciones basadas en microservicios

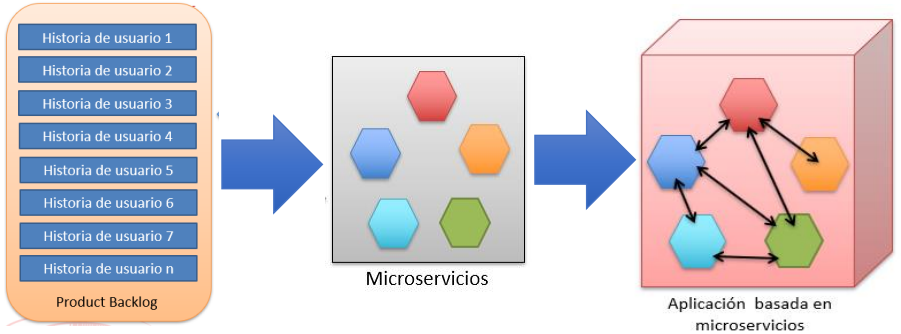
Migración de aplicación monolítica a microservicios



Desarrollo de aplicación sin microservicios ya desarrollados



Desarrollo de aplicación con microservicios ya desarrollados

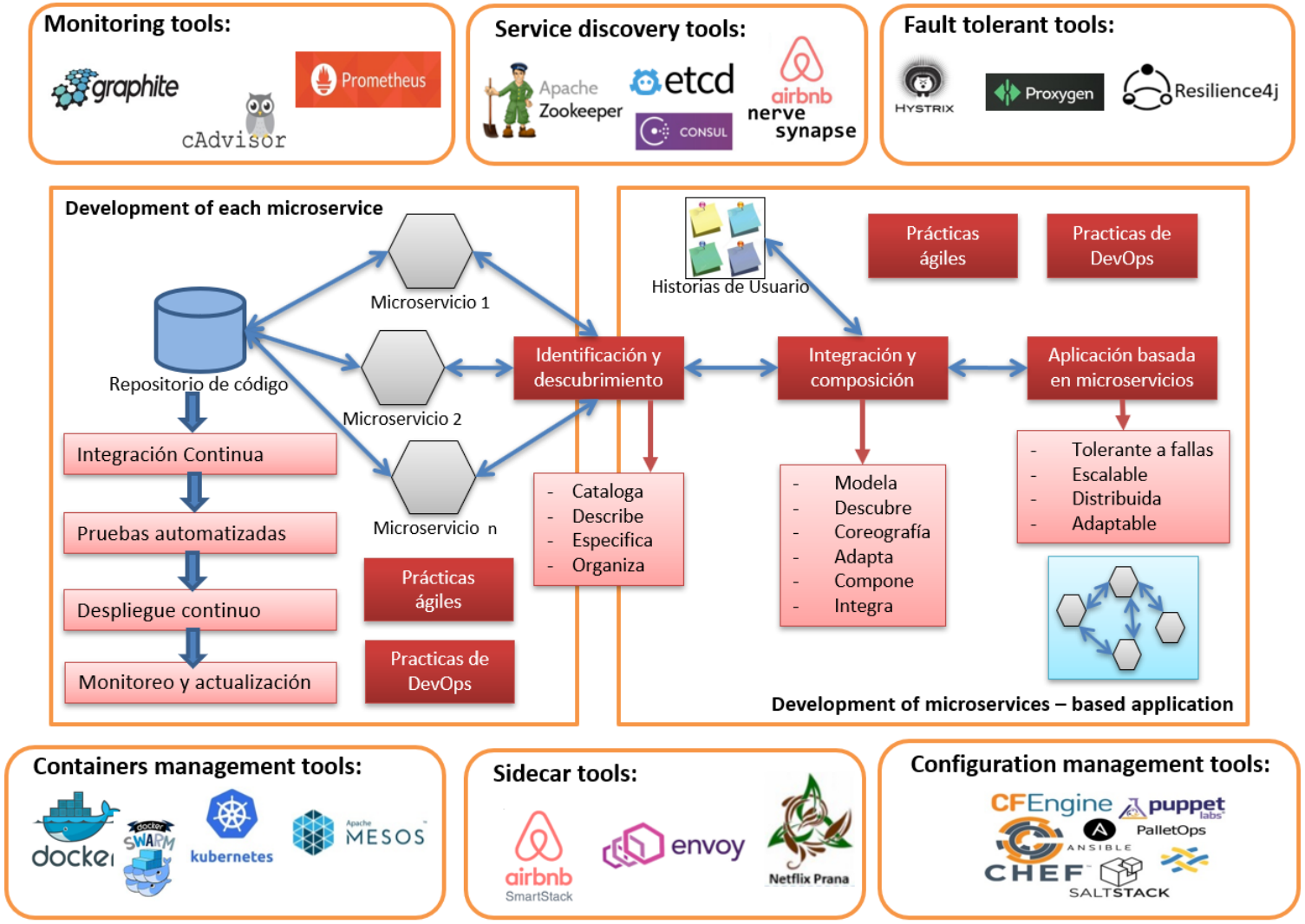


Proceso de desarrollo de aplicaciones basadas en microservicios

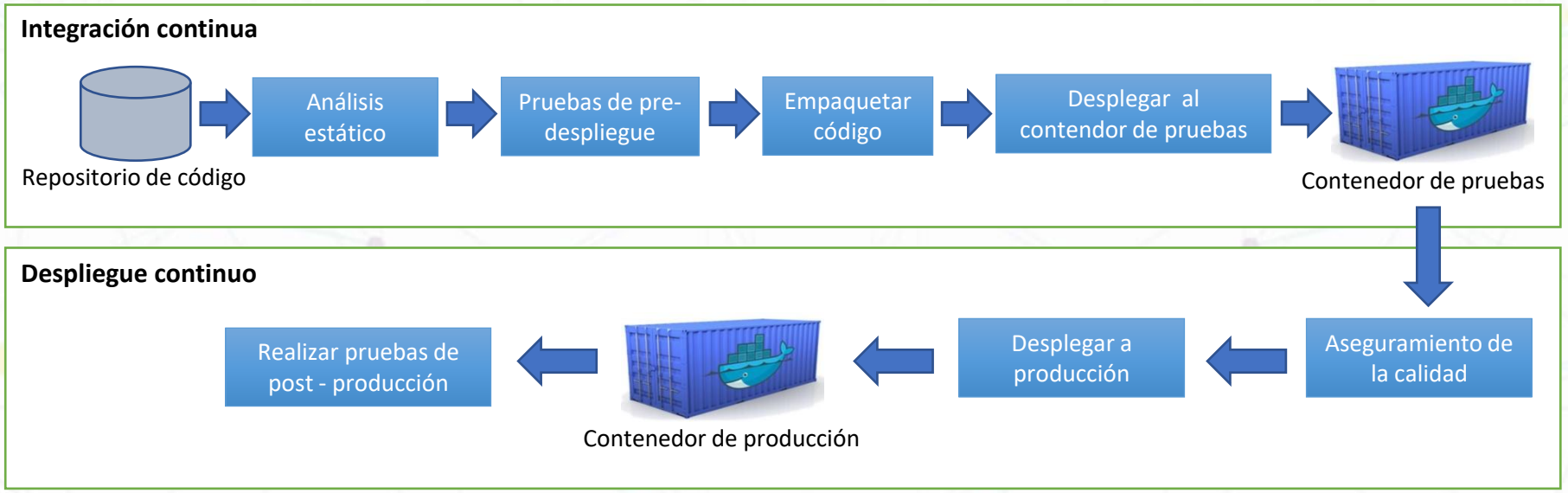
El proceso de desarrollo de aplicaciones basadas en microservicios se divide en dos partes fundamentales:

1. El desarrollo de cada microservicio.
2. El desarrollo de la aplicación basada en microservicios.

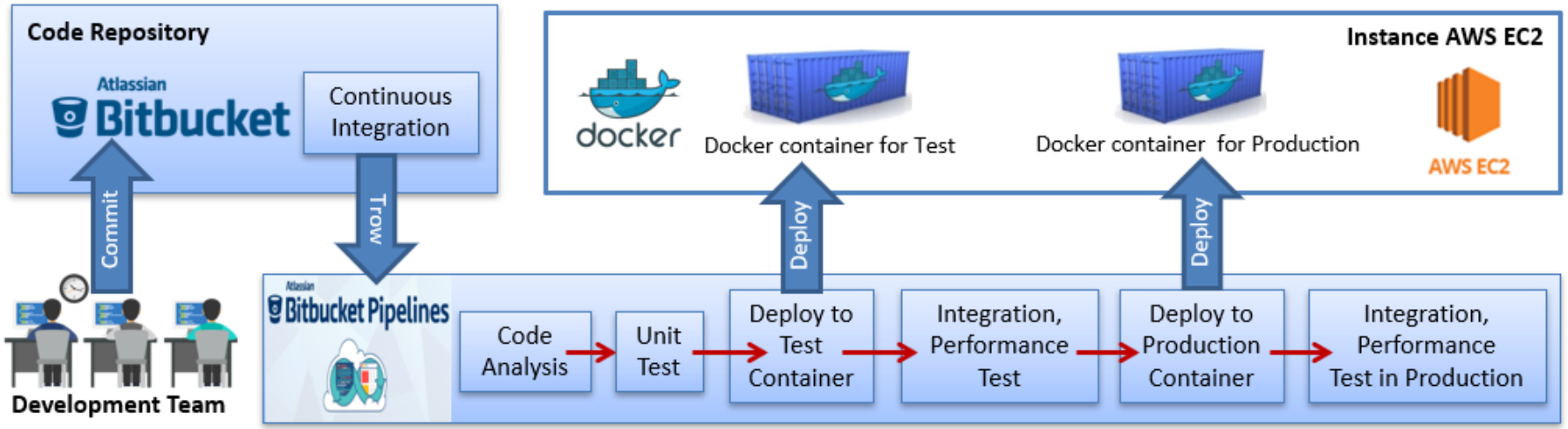
Las prácticas ágiles y DevOps son fundamentales en la arquitectura de microservicios.



Integración continuo y despliegue continuo

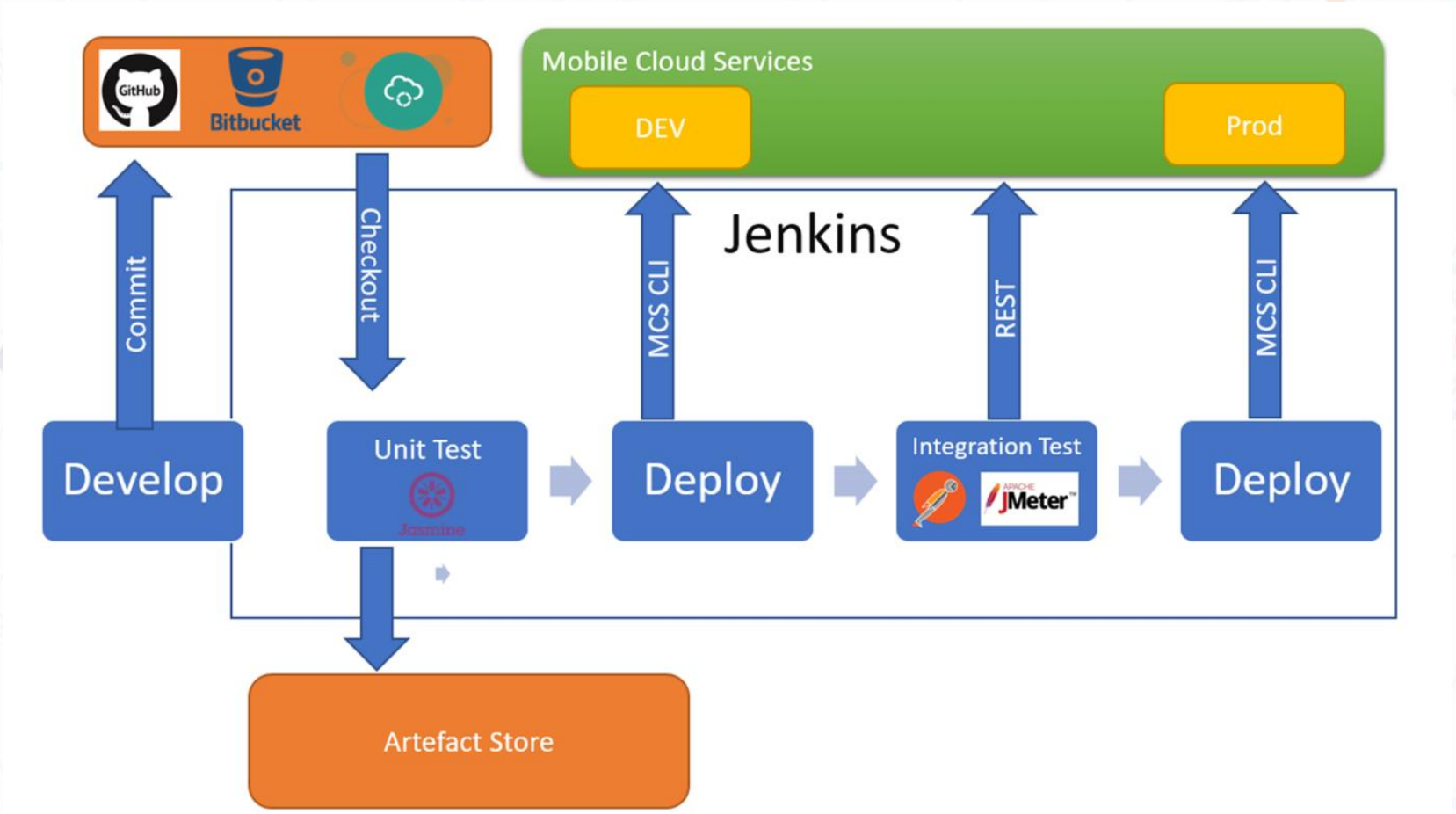


Automatización del despliegue continuo y pruebas automatizadas

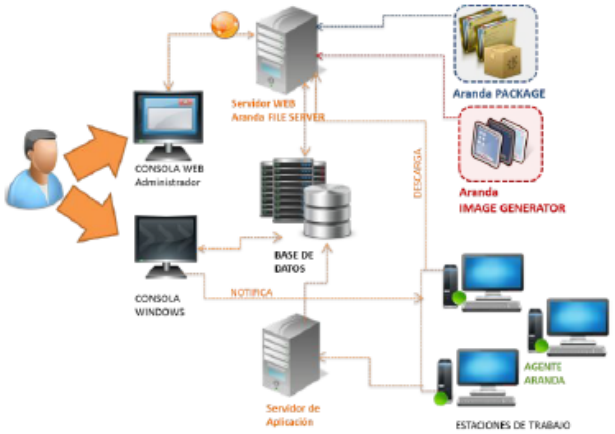


El equipo de desarrollo al realizar un “commit” en el repositorio de código, la “pipeline” de integración continua, pruebas automatizadas y despliegue continuo se activa garantizando que el microservicios sea desplegado en producción cumpliendo con los requerimientos de calidad

Automatización del despliegue continuo y pruebas automatizadas

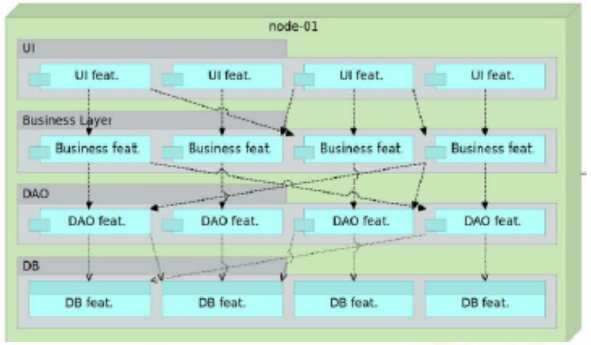


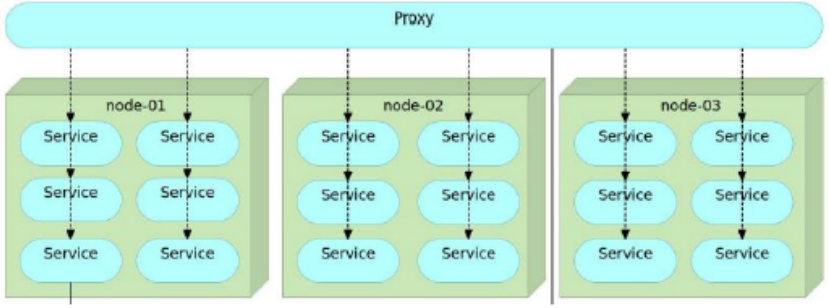
Comparación aplicaciones monolíticas y microservicios

	Monolitica	Microservicios
Complejidad Operacional	Se debe mantener, configurar y desplegar una sola aplicación.	Se debe mantener, configurar y distribuir múltiples aplicaciones más pequeñas independientes pero comunicadas.
	Con el paso del tiempo esta aplicación convive con otras y con diferentes configuraciones, aumenta la complejidad y es difícil de mantener.	Con las herramientas de gestión de la configuración se puede reducir esta complejidad.

Comparación aplicaciones monolíticas y microservicios

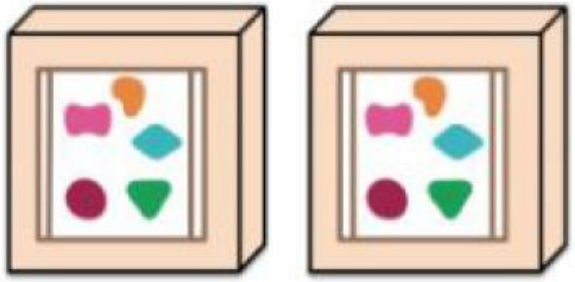
	Monolitica	Microservicios
Llamadas a procesos remotos	Se realizan dentro de la misma aplicación. Dentro del mismo servidor.	El rendimiento se afecta, sobre todo cuando están distribuidos en diferentes servidores.
	Las llamadas internas a través de clases y métodos son más rápidas.	Organizarlos en torno a contextos limitados o funcionalidades específicas.

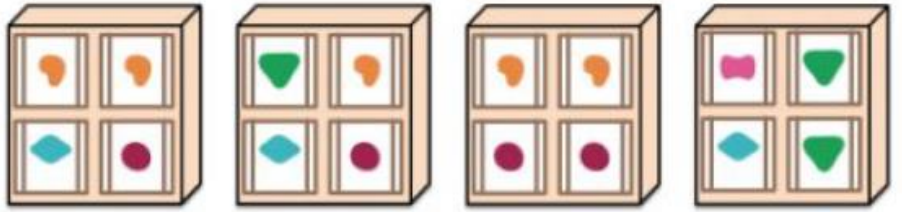




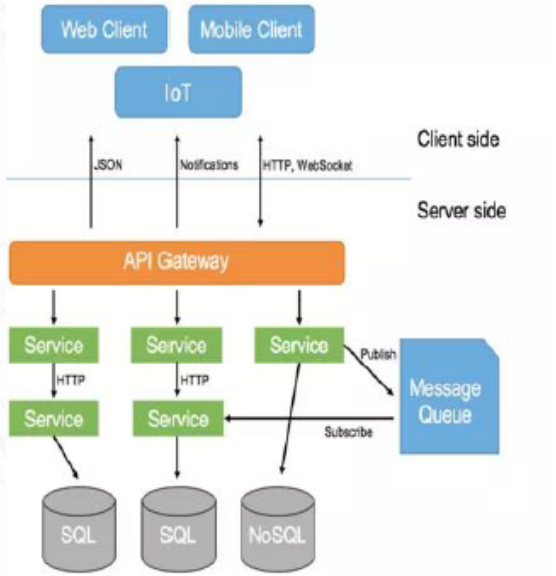
Comparación aplicaciones monolíticas y microservicios

	Monolitica	Microservicios
Escalabilidad	Se duplica toda la aplicación en una nueva máquina.	Se duplica sólo aquellos que necesitan escalamiento.
		Se pueden organizar mejor las cosas y escalar cada servicio por separado.

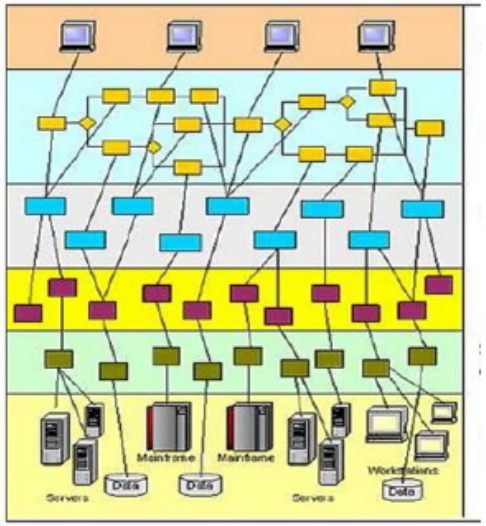




Comparación aplicaciones monolíticas y microservicios

	Monolitica	Microservicios
Innovación	Una vez que se hace la arquitectura inicial, no dejan mucho espacio para la innovación.	Se puede elegir lo que se crea sea la mejor solución para cada servicio por separado.
	Cambiar las cosas toma tiempo, y la experimentación es peligrosa, ya que potencialmente afecta a todo	Los cambios y los experimentos son mucho más fáciles de hacer, todo lo que se hace afecta sólo a uno de muchos microservicios y no al sistema en su conjunto

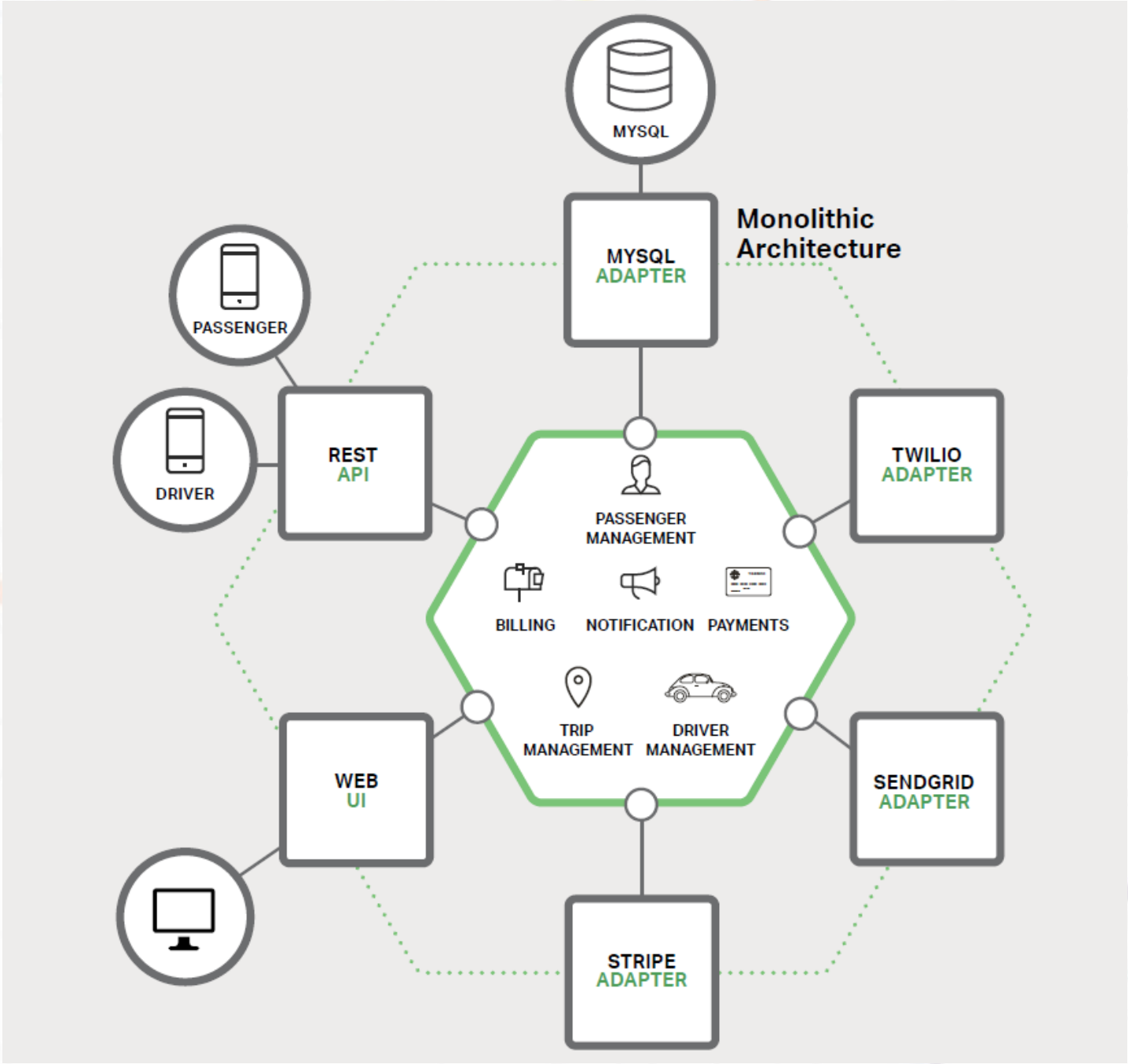
Comparación aplicaciones monolíticas y microservicios

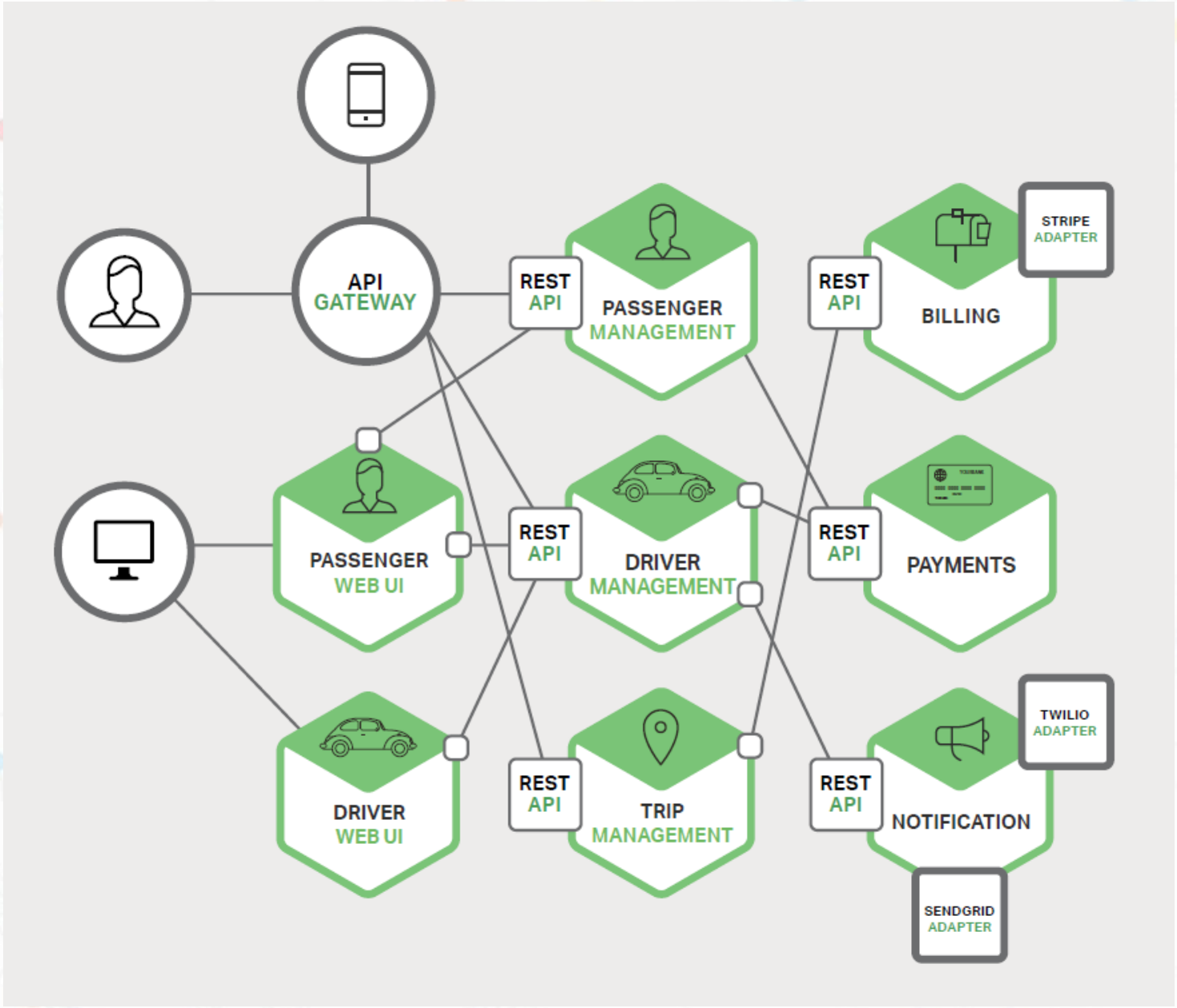
	Monolitica	Microservicios
Tamaño y despliegue	Aplicaciones grandes, enormes. Difícil de entender. El IDE es lento para manejarlas.	Son pequeños, son mucho más fáciles de entender. Los IDE trabajan más rápido con un proyecto pequeño
	Se carga un numero grande de bibliotecas y dependencias. Reinicio y despliegue lento.	Comienzan más rápido ya que no hay servidores enormes ni un número enorme de bibliotecas para cargar. Despliegue rápido.

Comparación aplicaciones monolíticas y microservicios

	Monolitica	Microservicios
Término del compromiso	Obligan a elegir desde el principio la arquitectura y las tecnologías que durarán mucho tiempo.	El compromiso a largo plazo es mucho menor. Cambie el lenguaje de programación en un microservicio y si resulta ser una buena opción, aplíquelo a otros
	Se estamos construyendo algo grande que debe durar mucho tiempo. Un cambio implica cambiar muchas cosas.	Si el experimento falló o no es el óptimo, sólo hay una pequeña parte del sistema que rehacer. El cambio se puede revertir fácilmente.

Arquitectura **monolítica** de un sistema para Taxis, parecido al UBER.



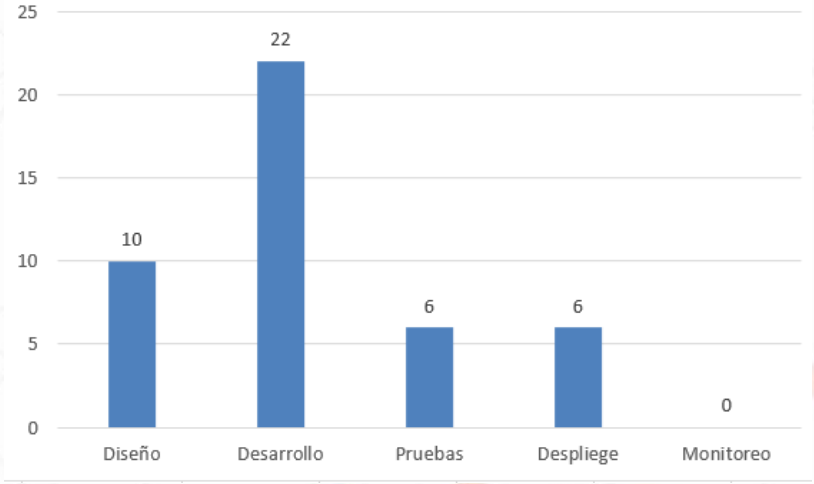


Arquitectura basada en **microservicios** de un sistema para Taxis, parecido al UBER.

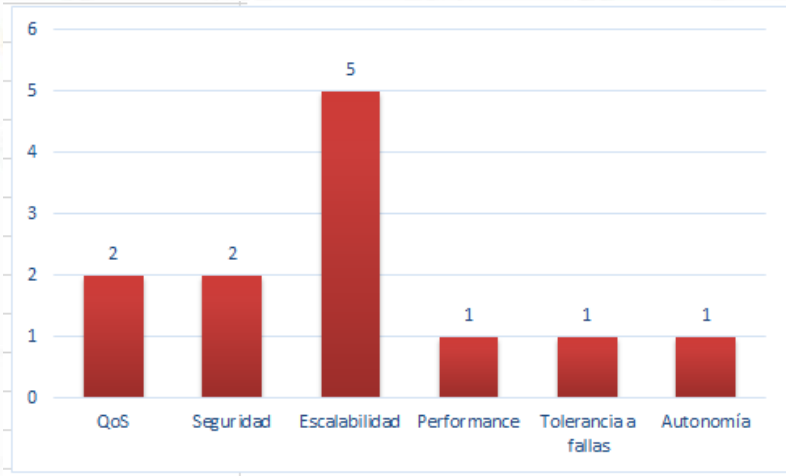
Estado del arte de la arquitectura de microservicios

Se realizó una revisión de literatura para establecer el estado actual del área, se revisaron las tendencias actuales de trabajos de investigación publicados en todo el mundo, 98 artículos, se clasificaron en:

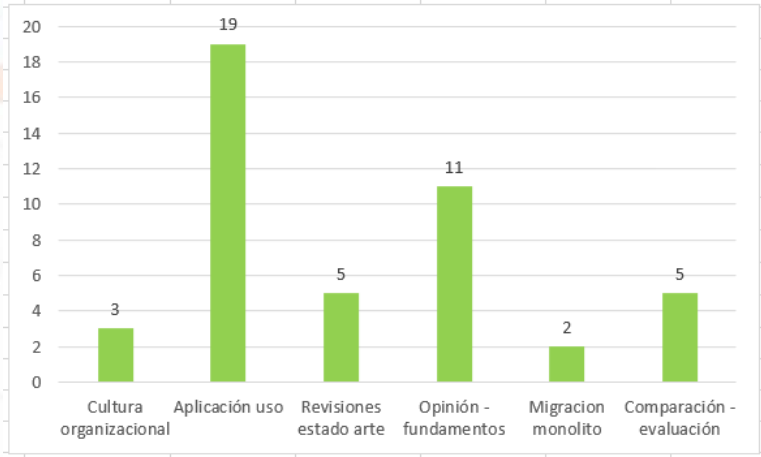
a. Por fase del proceso de desarrollo



b. Atributos de calidad

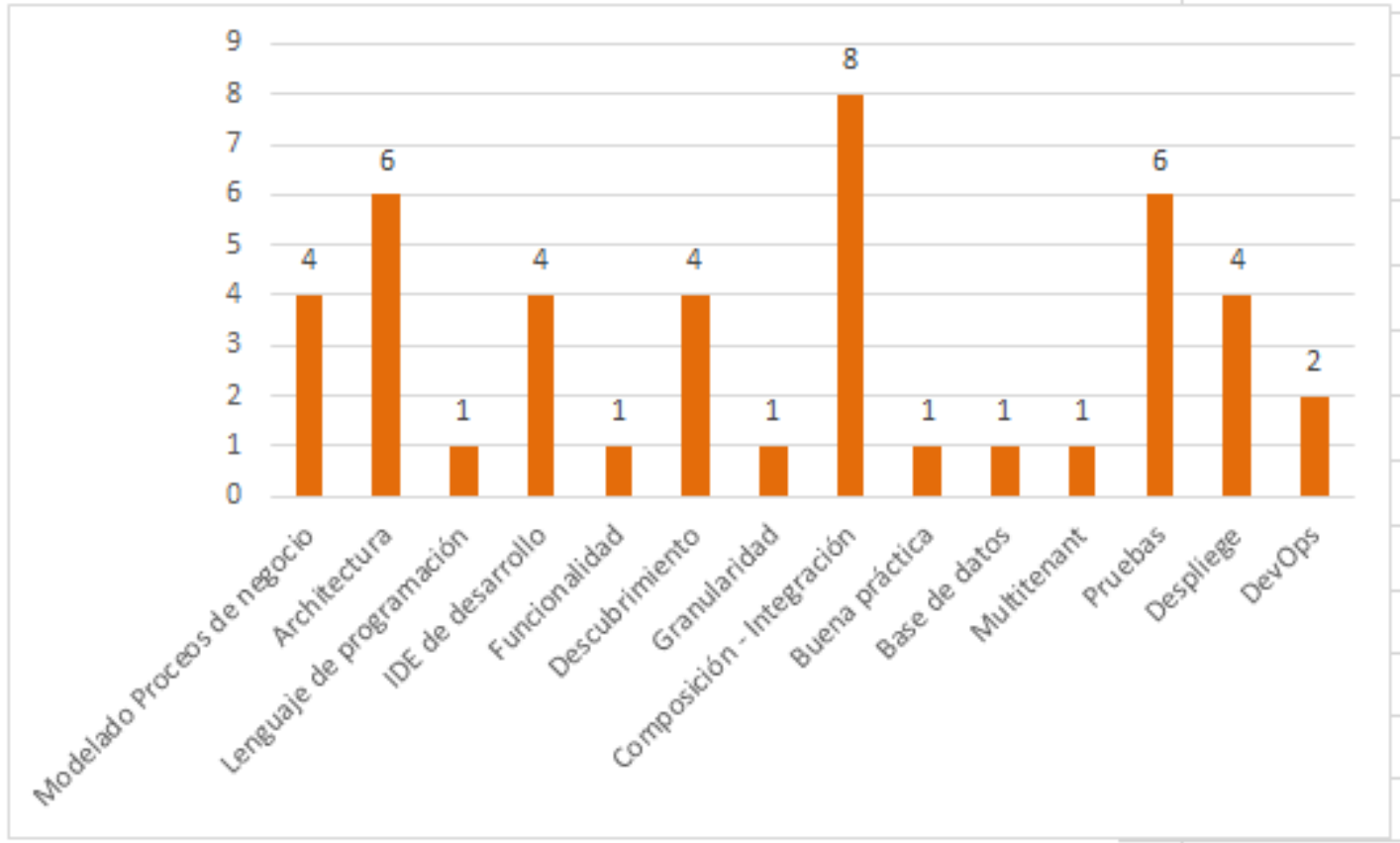


c. Otros aspectos



Estado del arte de la arquitectura de microservicios

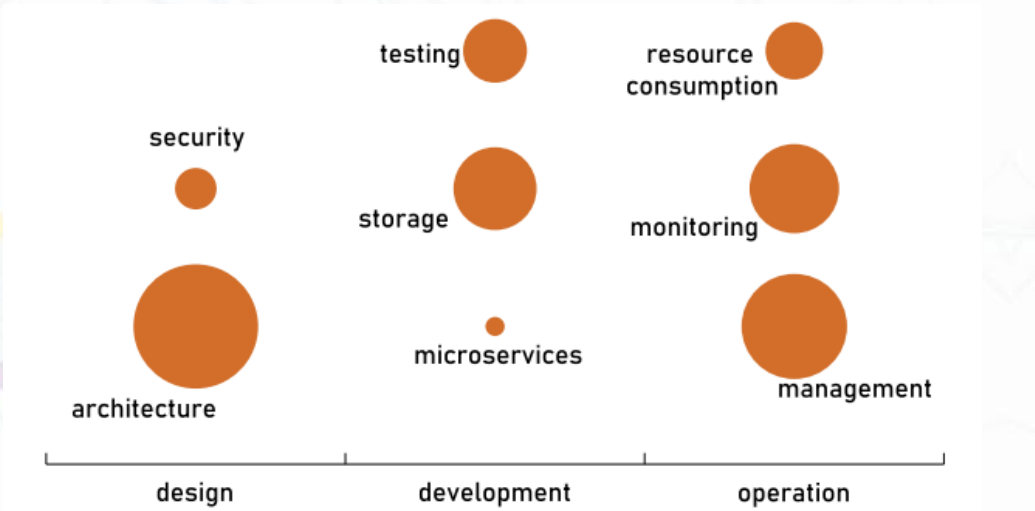
También se clasificaron por área operacional. Aquellos artículos relacionados con las fases del proceso de desarrollo.



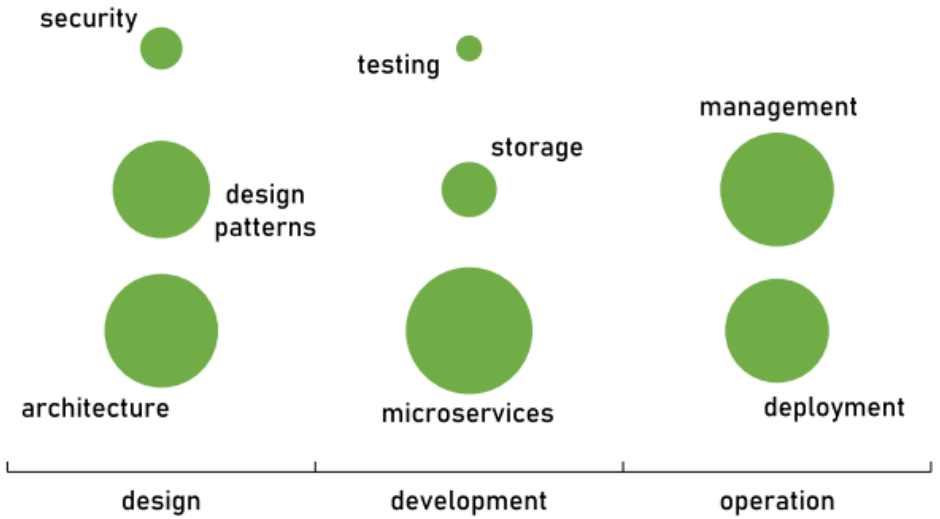
d. Distribución por área operacional

Estado de la práctica de la arquitectura de microservicios

a. Pains – Desventajas - Problemas

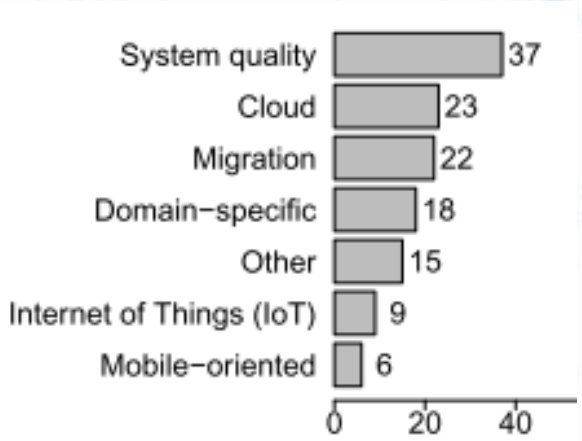


b. Gains – Ventajas - Ganancias



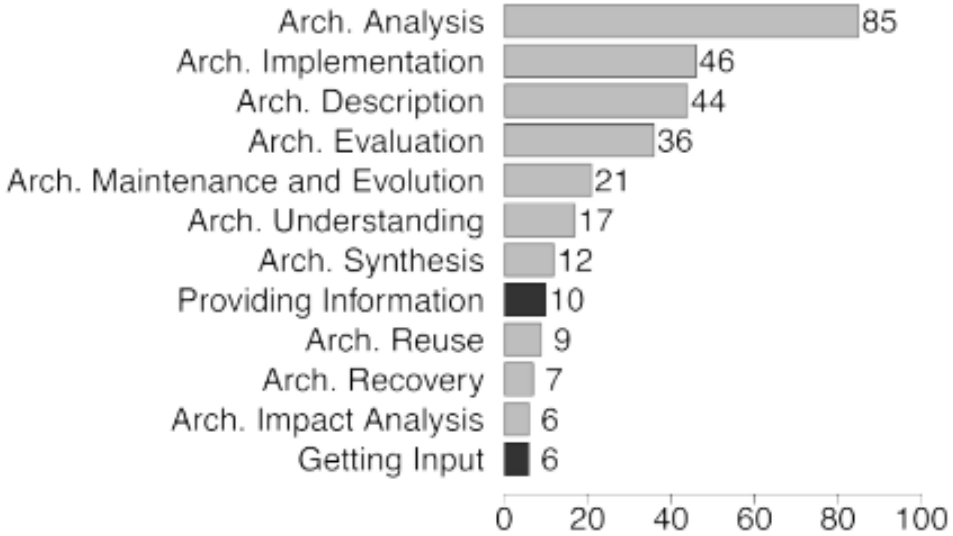
Soldani, J., Tamburri, D.A., Heuvel, W.-J. Van Den, 2018. The Pains and Gains of Microservices: A Systematic Grey Literature Review. J. Syst. Softw. 146, 215–232. https://doi.org/10.1016/j.jss.2018.09.082

Estado del arte de la arquitectura de microservicios

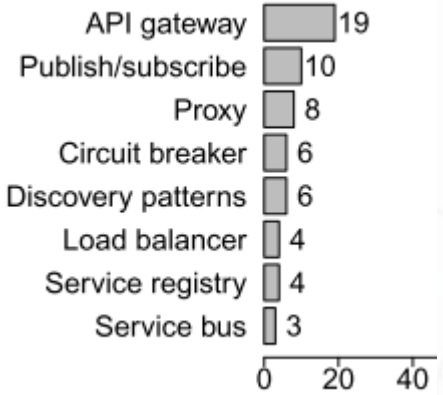


a. Principales áreas de investigación

b. Problemas abordados



c. Actividades de arquitectura



d. Patrones de diseño

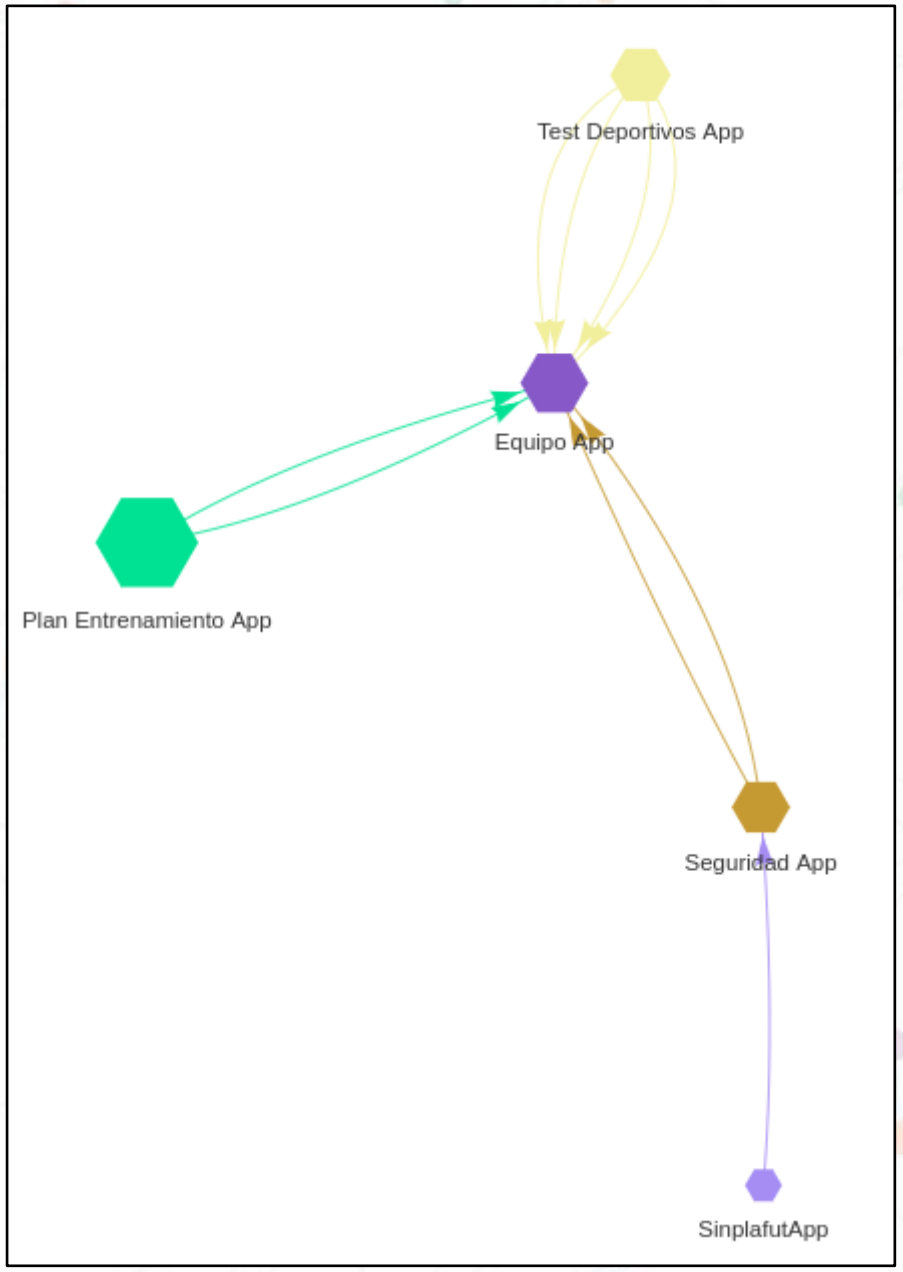
P. Di Francesco, P. Lago, and I. Malavolta, "Architecting with microservices: A systematic mapping study," *J. Syst. Softw.*, vol. 150, pp. 77–97, Apr. 2019.

Caso de estudio Sinplafut

SINPLAFUT
Vera-Rivera, F. H.; Vera-Rivera, J. L.; Gaona-Cuevas, C. M. 2019; https://doi.org/10.1088/1742-6596/1388/1/012026.
Grupos de investigación GIA, GEDI, GRINDER, UFPS, UNIVALLE. Está en desarrollo.
Sistema de gestión de la planificación del entrenamiento deportivo en el fútbol.

Caso de estudio Sinplafut

Sinplafut, una aplicación tipo SaaS que permite realizar la planificación del entrenamiento en el fútbol usando técnicas modernas del entrenamiento deportivo. Sinplafut puede ser utilizado por equipos de fútbol, clubes, preparadores físicos, entrenadores y directores técnicos tanto a nivel aficionado como profesional.



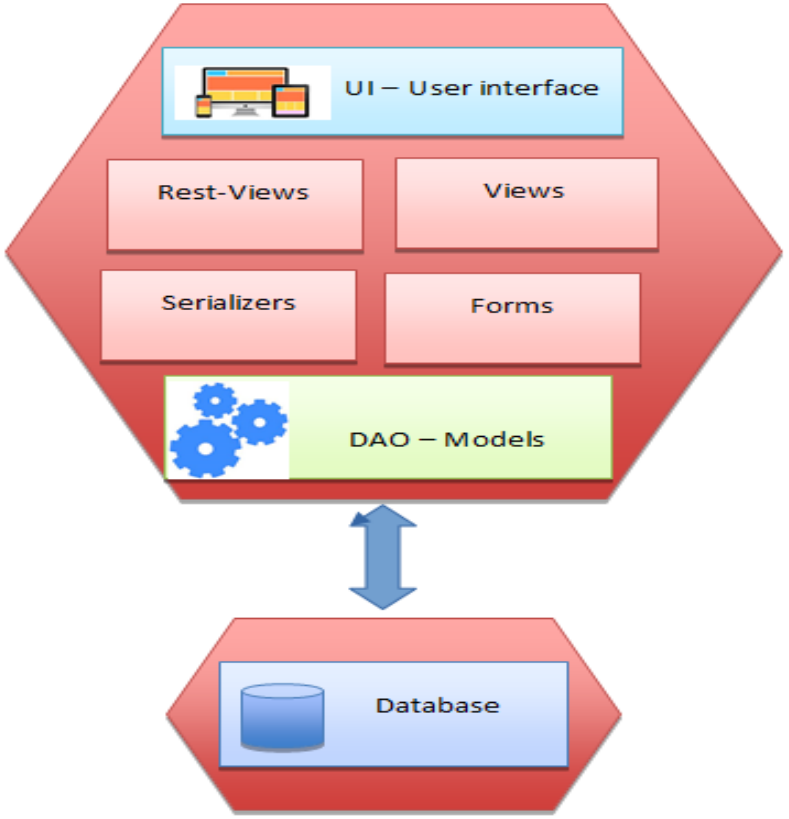
Caso de estudio Sinplafut

HISTORIAS DE USUARIO ÉPICAS		
Nombre	**Número de historias**	**Puntos – Dev. Time**
1. Gestión del club Deportivo	4	7 – 14h
2. Gestión de equipo del club deportivo	6	16 – 32h
3. Gestión de jugadores del equipo	11	33 – 66h
4. Gestión del cuerpo técnico	6	9 – 18h
5. Gestión del plan de entrenamiento	5	39 – 78h
6. Gestión de mesociclos	7	15 – 30h
7. Gestión de microciclos	6	13 – 26h
8. Gestión de sesiones de entrenamiento	6	20 – 40h
9. Gestión de métodos del entrenamiento deportivo	8	30 – 60h
10. Generar cronograma de las sesiones de entrenamiento	1	8 – 16h
11. Generar landing page de Sinplafut	1	10 – 20h
12. Gestión de test deportivos	14	52 - 104h
13. Gestión de usuario y seguridad	17	50 – 100h
Total	**92**	**302 – 604h**

Vera-Rivera FH, Vera-Rivera JL, Gaona-Cuevas CM. 2019. Sinplafut: A microservices – based application for soccer training. In: *5th International Week of Science, Technology & Innovation, Cúcuta, Colombia, 2018. Journal of Physics: Conference Series*. 012026. DOI: 10.1088/1742-6596/1388/1/012026.

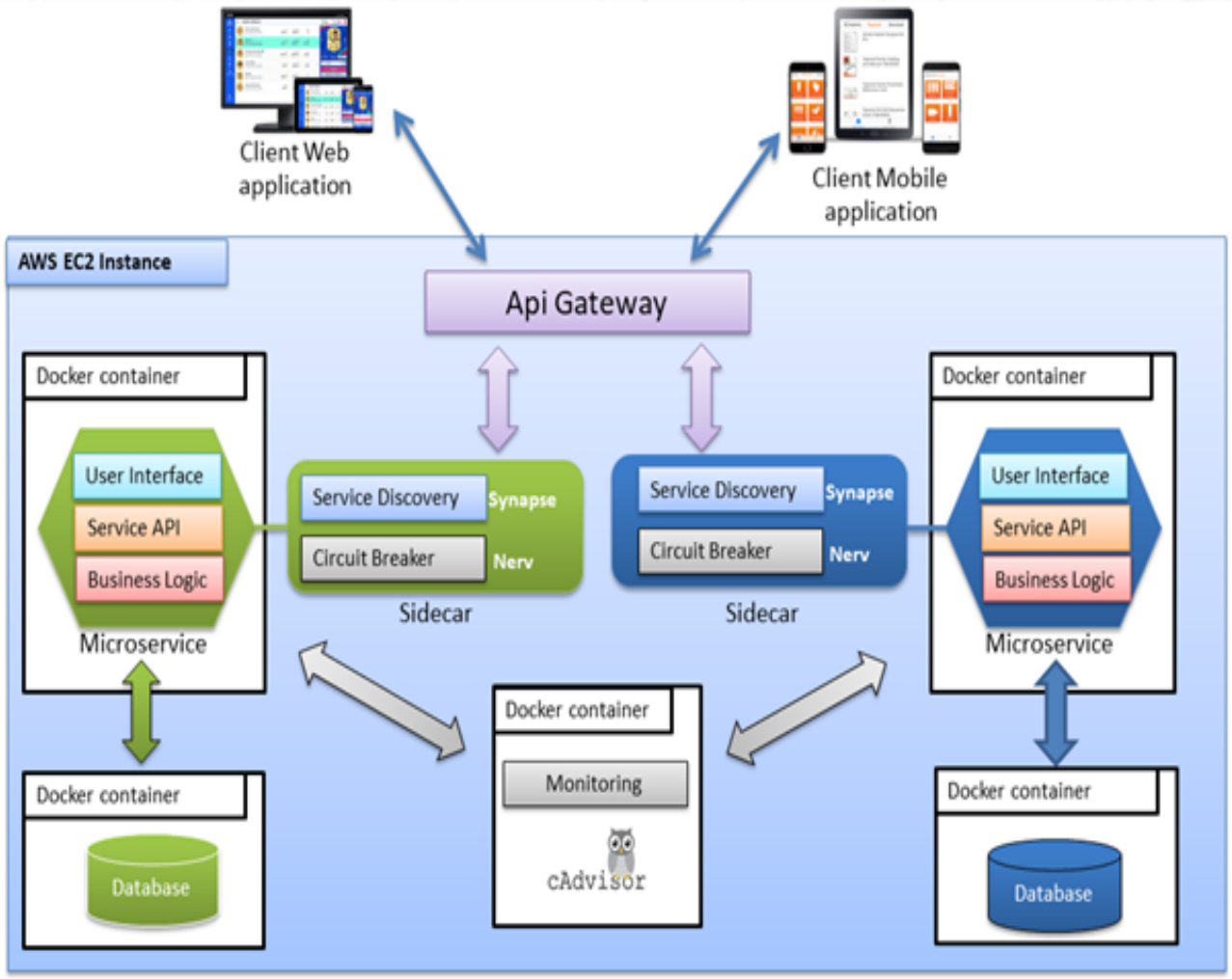
Sinplafut - implementación

1. **sinplafutApp**: Desplegado en: http://sinplafut:8030/. Fron-ent del sistema.
2. **EquipoApp**: Desplegado en: http://sinplafut:8050/. Gestión de equipos, jugadores y cuerpo técnico.
3. **PlanEntrenamientoApp**: Desplegado en: http://sinplafut:8060/. Gestión del plan de entrenamiento y sus sesiones de entrenamiento.
4. **Monitoreo**: Desplegado en: http://sinplafut:8080/. CAdvisor, Google.



Sinplafut - implementación

Arquitectura de los microservicios implementados en Sinplafut



Sinplafut – implementación – pruebas y despliegue automatizado

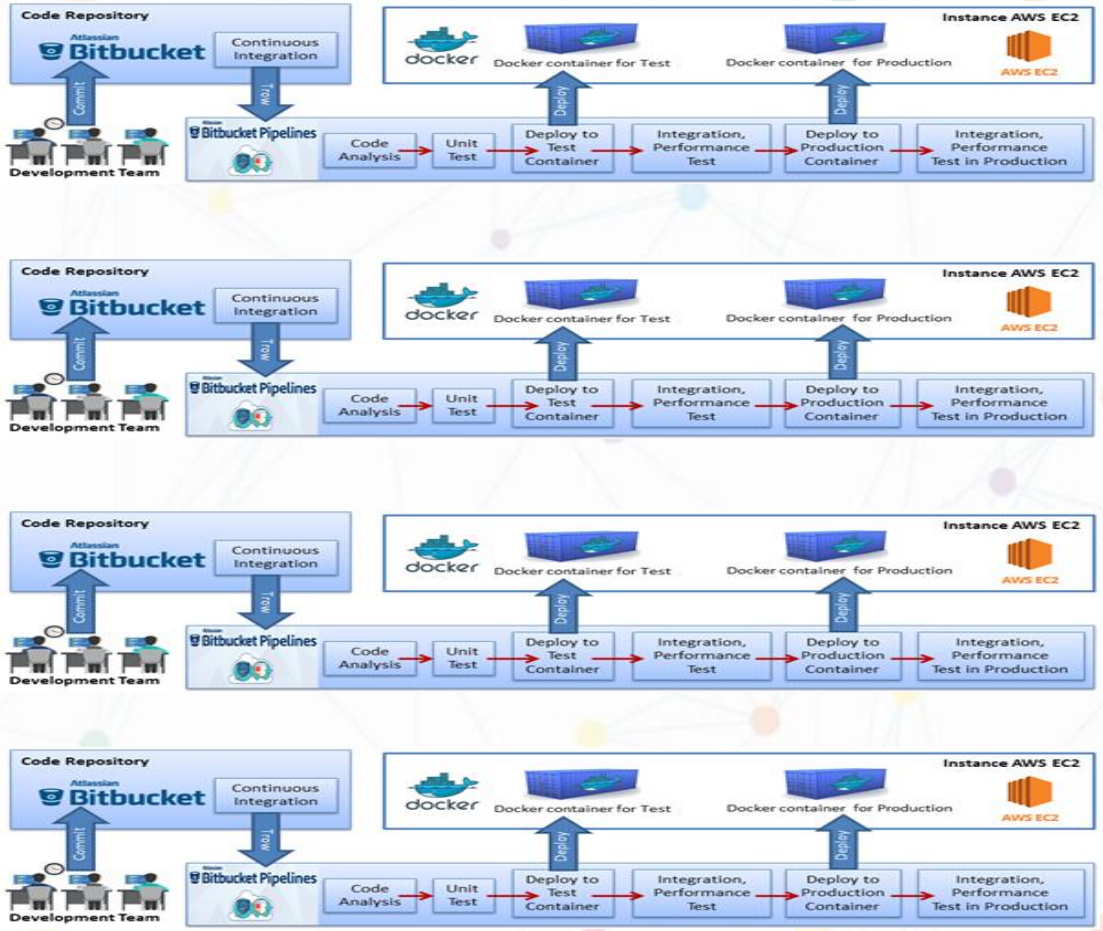
Integración continua, pruebas automatizadas y despliegue continuo para cada microservicios

PlanEntrenamientoApp

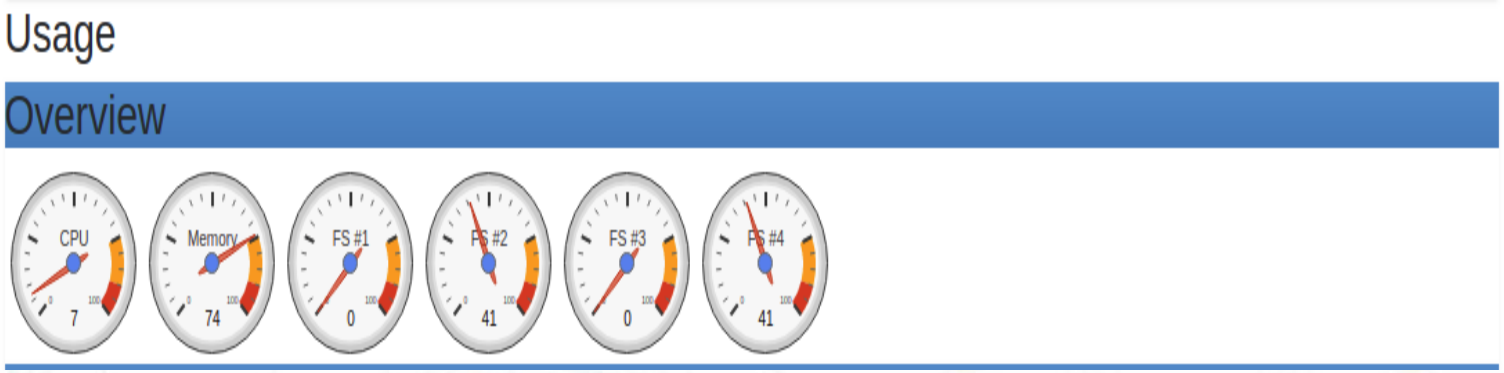
EquipoApp

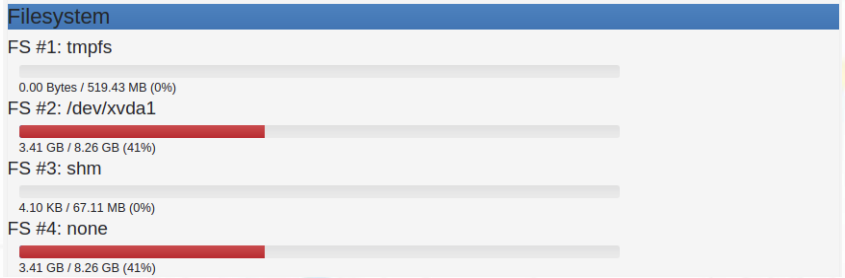
TestDeportivosApp

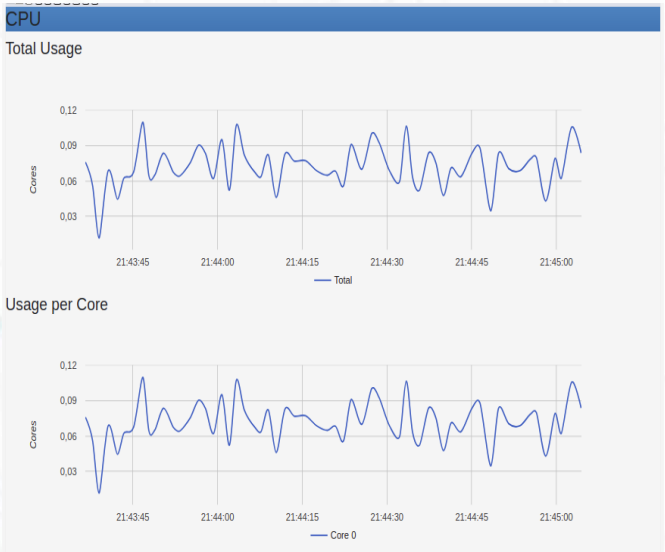
SinplafutApp

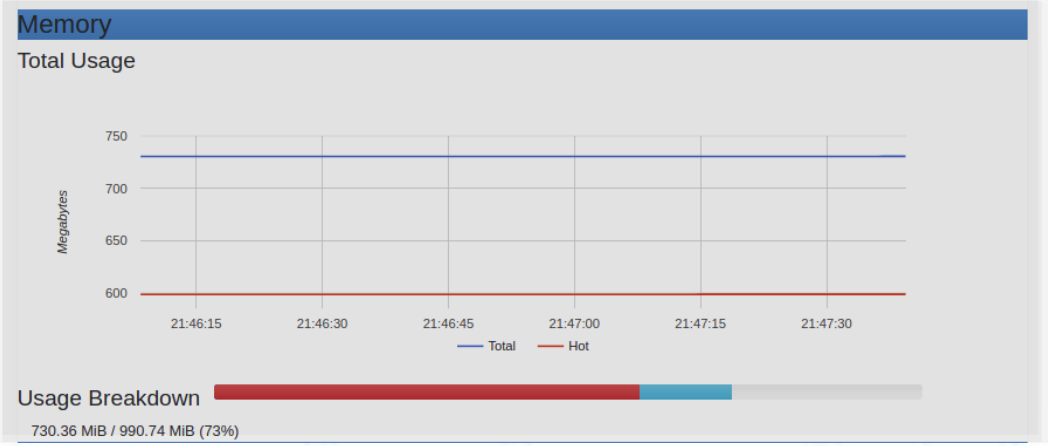


Sinplafut - monitoreo









Actividad en Clase

1. Para el ejemplo del servicio implementado en la clase anterior, investigar cómo convertirlo en un microservicio, empaquetarlo en un contenedor (Docker) para que sea desplegado.

2. Definir su proyecto de clase, establecer la arquitectura del sistema y los microservicios que van a implementar.

MUCHAS GRACIAS

Dr. Fredy H. Vera-Rivera
fredyhumbertovera@ufps.edu.co
freve9@gmail.com
@freve9