UNIVERSIDAD TÉCNICA DE BABAHOYO
FACULTAD DE ADMINISTRACIÓN, FINANZAS E INFORMÁTICA

PROCESO DE TITULACIÓN

NOVIEMBRE 2020 - MAYO 2021

EXAMEN COMPLEXIVO DE GRADO O DE FIN DE CARRERA

PRUEBA PRÁCTICA

INGENIERÍA EN SISTEMAS

PREVIO A LA OBTENCION DEL TITULO DE INGENIERO EN SISTEMAS

TEMA:

ESTUDIO DE LAS APLICACIONES WEB EMPRESARIALES, DESARROLLADAS EN EL LENGUAJE DE PROGRAMACIÓN JAVA, EN LOS FRAMEWORKS HIBERNATE Y SPRING.

EGRESADO:

VIEJO CAVERO DIEGO ARMANDO

TUTOR:

ING. RUIZ PARRALES IVAN RUBEN

AÑO 2021

INTRODUCCIÓN

Hoy en día en las prácticas de programación es muy común el uso de frameworks que permitan agilizar el proceso de la programación, son muchos los ámbitos en los que se puede usar un framework, ya sea para programación web, programación móvil entre otros, los frameworks representan una herramienta muy importante en el proceso de desarrollo grandes aplicaciones, ya que como es de conocimiento en los programadores entre más grande es la aplicación y más módulos esta posea, será más complejo realizar el mantenimiento respectivo de las mismas.

El objetivo de este trabajo es estudiar la manera en que las aplicaciones empresariales han implementado los frameworks actualmente más conocidos del mundo de desarrollo de java como lo son Hibernate y Spring, estos son utilizados mayormente para la creación de aplicaciones web, aunque también se pueden desarrollar aplicaciones de escritorio de manera más sencilla utilizando patrones de desarrollo muy conocidos como el MVC (Modelo Vista Controlador).

El presente trabajo se realizó siguiendo los lineamientos determinados en la línea de investigación de Sistemas de información y comunicación, emprendimiento e innovación, y en la sub línea de investigación que comprende las redes y tecnologías inteligentes de software y hardware, en la investigación se utilizó como técnica la recopilación documental y bibliográfica.

La comunidad de programadores que usan estos framework hoy en día es bastante extensa, debido a esto la información utilizada en la realización de este trabajo será citada de fuentes bibliográficas y artículos realizados por parte de la comunidad de desarrolladores de java.

Se utilizó el método descriptivo, donde (Ramos Chagoya, 2018) indica que "Su preocupación primordial radica en describir algunas características fundamentales de conjuntos homogéneos de fenómenos, utilizando criterios sistemáticos que permitan poner de manifiesto su estructura o comportamiento". Con este método logramos establecer las cualidades de cada uno de los frameworks y el desempeño que se puede alcanzar integrando estas herramientas al momento de desarrollar aplicaciones empresariales.

Las nuevas formalidades de programación que se han establecido en el mundo hacen que cada vez más desarrolladores independientes y empresas dedicadas al desarrollo de soluciones informáticas hagan uso de frameworks que faciliten el trabajo de los grupos de desarrolladores, esto con el objetivo de que el código escrito sea los más legible posible y ejecutar cambios en el mismo sea sencillo.

Hoy cuando se habla de programación de aplicaciones web empresariales desarrolladas en el lenguaje de programación java se habla de Java Enterprise Edition (JEE), este se ha establecido informalmente como un estándar de desarrollo, las diferentes API's tales como JDBC, Servicios web, XML, etc, estos permiten desarrollar aplicaciones empresariales portables y escalables entre plataformas.

DESARROLLO

En la actualidad existen grandes aplicaciones web desarrolladas en diferentes tipos de lenguajes de programación, las mismas han pasado por un proceso de evolución a lo largo del tiempo, la creación de esta aplicaciones ha implicado la el uso de infraestructuras de desarrollo que permitan a las misma poder funcionar de la mejor manera posible dependiendo de la época en la que fue creada, sin embargo, hoy la mayoría de las grandes aplicaciones web creadas apuntan a el desarrollo de microservicios, existen diferentes lenguajes de programación que pueden desarrollar microservicios que permiten hacer un sitio web autónomo. Los microservicios son considerados una arquitectura de programación que se basa en la creación de varios elementos independientes que cumplen una funcion en específico, a la vez funcionan en conjunto para cumplir una determinada tarea. Otras de las infraestructuras que predomina es la monolítica donde todos los aspectos funcionales del software quedan acoplados en un mismo programa.

Según (López & Maya , 2017) en una aplicación monolítica toda la lógica se ejecuta en un único servidor de aplicaciones. Las aplicaciones monolíticas típicas son grandes y construidas por múltiples equipos, requiriendo una orquestación cuidadosa del despliegue para cada cambio. También se consideran aplicaciones monolíticas cuando existen múltiples servicios de API que proporcionan la lógica de negocio, toda la capa de presentación es una sola aplicación web grande. En ambos casos, la arquitectura de microservicios puede proporcionar una alternativa.

(Lewis & Fowler, 2014) dice que el término "Arquitectura de microservicio" ha surgido en los últimos años para describir una forma particular de diseñar aplicaciones de software como conjuntos de servicios desplegables de forma independiente. Si bien no existe una definición precisa de este estilo arquitectónico, existen ciertas características comunes en torno a la organización en torno a la capacidad empresarial, la

implementación automatizada, la inteligencia en los puntos finales y el control descentralizado de idiomas y datos.

La infraestructura que predominaba el desarrollo de aplicaciones web era la infraestructura monolitica hasta que apareció la infrestructura de microservicios como respuesta a las falencias de la monolítica.

A continuación, se comparan las dos infraestructuras de desarrollo.

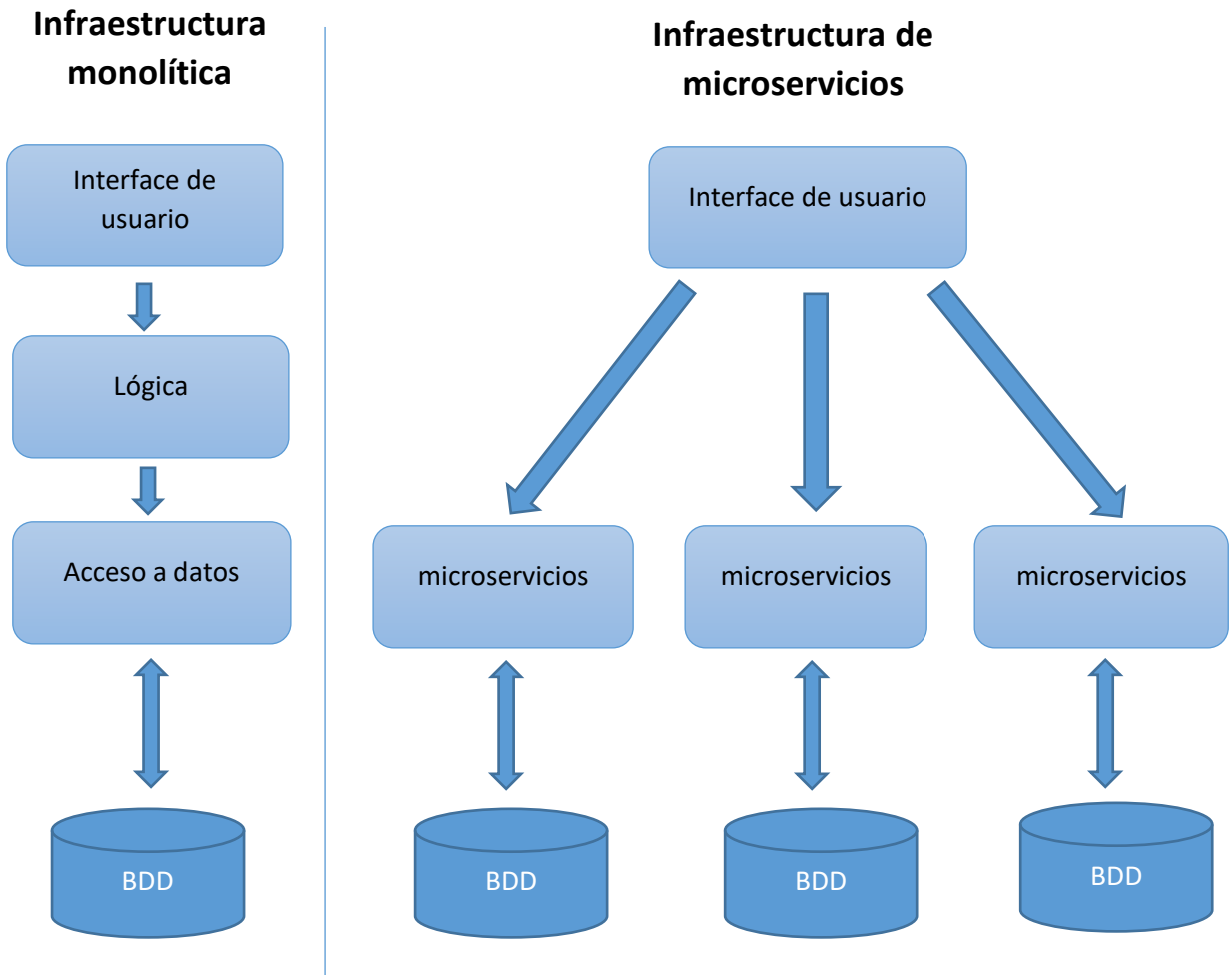


Ilustración 1: Infraestructura monolítica y de microservicios

Elborado por: Diego Viejo

En una arquitectura de microservicios, cada servicio ejecuta un proceso único y generalmente administra su propia base de datos. Esto no solo proporciona a los equipos de desarrollo un enfoque más descentralizado para la creación de software, sino que también permite que cada servicio se implemente, reconstruya, implemente de nuevo y se administre de manera independiente. (Novoseltseva, 2018)

Los microservicios, cuando se implementan correctamente, pueden mejorar enormemente la confiabilidad, la escalabilidad y mantenibilidad del software. Uno de los aspectos más convincentes cuando se compara con una implementación monolítica, por ejemplo, es que un error en un servicio no afecta la capacidad ni el funcionamiento de los otros servicios para seguir trabajando según lo previsto. (GreenSQA, 2019)

A continuación, se manejan algunas de las ventajas de esta arquitectura según (GreenSQA, 2019):

- Capaz de escalar de forma independiente.
- Tolerante a fallos.
- Se puede intercambiar o reescribir fácilmente.
- Fuerte soporte con arquitectura SOLID.
- Adherirse al principio KISS.
- Los Microservicios se pueden reutilizar en diferentes aplicativos de software.

A continuación, se muestra una tabla con las características de cada arquitectura.

Categoría	**Arquitectura de microservicios**	**Arquitectura monolítica**
Código	La aplicación funciona con múltiples fuentes de código, cada microservicio tiene su propia fuente.	La aplicación funciona con una única base de código.
Lenguaje	Al ser independientes, cada microservicio puede desarrollarse en un lenguaje de programación diferente.	Toda la aplicación está desarrollada en un solo lenguaje de programación.
Despliegue	Cada microservicio puede implementarse de manera individual, lo que lo vuelve sencillo de desplegar con un tiempo de inactividad mínimo.	Implementaciones complejas, deben programarse mantenimientos contantes.
comprensibilidad	Cada microservicio es fácil de mantener debido a su facilidad de lectura.	Debido a la longitud del código fuente se vuelve confuso y difícil de mantener.

Tabla 1: Características de las arquitecturas monolíticas y de microservicios

Elaborado por: Diego Viejo.

Netflix, eBay, Amazon, el Servicio Digital del Gobierno del Reino Unido, Twitter, PayPal, The Guardian y muchos otros sitios web y aplicaciones a gran escala han evolucionado desde la arquitectura monolítica a la de microservicios. (Novoseltseva, 2018)

El primer stack de microservicios que se dio a conocer, o que alcanzó una relativa popularidad, fue el stack de Netflix, que fue adoptado por Spring como su principal solución para Cloud bautizándolo como spring-cloud-netflix y que vio la luz en su primera release en Marzo de 2015. (Rodríguez, 2018)

(Novoseltseva, 2018) dice que en 2001, el sitio web minorista de Amazon.com era un gran monolito arquitectónico. Estaba estructurado en varios niveles, y esos niveles tenían muchos componentes en ellos, pero estaban muy juntos y se comportaban como un gran monolito. Tenían una gran cantidad de desarrolladores trabajando en un gran sitio web monolítico, y aunque cada uno de estos desarrolladores solo trabajaba en una parte muy pequeña de esa aplicación, aún necesitaban ocuparse de coordinar sus cambios con todos los demás que también trabajaban en el mismo proyecto. Cuando agregaban una nueva función o realizaban una corrección de errores, tenían que asegurarse de que el cambio no rompería ninguna otra cosa en ese proyecto.

Spotify brinda servicios a más de 75 millones de usuarios activos por mes, con una duración promedio de sesión de 23 minutos, mientras ejecuta roles empresariales increíblemente complejos entre bastidores. Por lo tanto, Spotify llegó a la conclusión de que si le preocupa el escalado a cientos de millones de usuarios, construye su sistema de manera que escale los componentes de forma independiente. Y Spotify construyó una arquitectura de microservicio con equipos autónomos de pila completa a cargo para evitar el infierno de sincronización dentro de la organización. Estos equipos son autónomos y su misión no se superpone con la misión de otros equipos. (Novoseltseva, 2018)

Se citaron algunos de los casos mas conocidos de grandes aplicaciones mundiales que basan su funcionamiento en la arquitectura de microservicios, los mismos pueden ser creados en diferentes lenguajes de programacion, en la actualidad a estos micorservicios cuando se implementan correctamente, pueden mejorar enormemente la confiabilidad, la escalabilidad y mantenibilidad del software

Java es un lenguaje de programación de propósito general orientado a objetos, que fue diseñado específicamente para tener tan pocas dependencias de implementación como fuera posible permitiendo a desarrolladores escribir un programa y ejecutarlo en cualquier tipo de dispositivo sin tener que compilarlo una y otra vez. (Robledano, 2019)

Sun principios básicos segun (Robledano, 2019) son:

- Simple. Una de las ventajas de Java reside en su sencillez con una moderada curva de aprendizaje. Esto hace que sea el lenguaje más usado en escuelas y universidades para mostrar los fundamentos de la programación.
- Multihilo. Considerando el entorno multithread (multihilo), cada thread (hilo, flujo de control del programa) representa un proceso individual ejecutándose en un sistema. Cada hilo controla un único aspecto dentro de un programa, como puede ser supervisar la entrada en un determinado periférico o controlar toda la entrada/salida del disco. Todos los hilos comparten los mismos recursos, al contrario que los procesos, en donde cada uno tiene su propia copia de código y datos (separados unos de otros).
- Seguro. Java es un lenguaje de programación seguro y estable. Pensado para poder operar en multitud de entornos. Desde el sector más lúdico a aplicaciones empresariales.

- Multiplataforma. Podemos desarrollar nuestro código una única vez y ejecutarlo en cualquier plataforma. Lo que facilita el poder portar nuestro proyecto a diferentes sistemas operativos.

Según (Fontanet, 2016) Java Enterprise Edition, Java EE en adelante, es un conjunto de estándares de tecnología s dedicadas al desarrollo de Java del lado del servidor. La plataforma Java EE consta de un conjunto de servicios, API y protocolos que proporcionan la funcionalidad necesaria para desarrollar aplicaciones basadas en web de varios niveles. Es decir, desarrollaremos aplicaciones empresariales distribuidas, con arquitecturas multicapa, escritas en Java y que se ejecutan en un servidor de aplicaciones.

Spring es un framework basado en J2EE que implementa el modelo MVC. Este framework fue lanzado bajo la licencia Apache 2.0 Licence en junio de 2003 por Rod Jonhson, su creador. Durante su lanzamiento los desarrolladores consideraban Spring un avance con respecto al modelo de programación que implementaba Enterprise JavaBeans (EJB). Gran parte de la popularidad de Spring se basa en la implementación del modelo MVC que hace uno de sus módulos. Spring logró popularizar en su momento buenas técnicas de programación al estar implementado usando distintos patrones de diseño. Es código abierto. (Palumbo & Tunessi, 2013)

En general, Spring aumenta la productividad y reduce la fricción al ofrecernos abstracciones sobre implementaciones de tecnologías concretas. Un ejemplo claro es el de spring-data, que nos permite definir el acceso a base de datos con interfaces Java. Esto lo consigue parseando el nombre de los métodos y generando la consulta con la sintaxis específica para el driver que utilicemos. Por ejemplo, cambiar nuestra aplicación de MySQL a PostgreSQL es tan sencillo como cambiar el driver: Spring se encarga de la sintaxis de forma transparente. (Pahino, 2020)

La inyección de dependencias es un patrón de desarrollo de software donde los objetos no son responsables de inicializar sus dependencias, sino que estas son provistas a través de otro objeto. En el caso de Spring ese objeto es el contenedor IoC el cual es provisto por los módulos spring-core y spring-beans. (Moreno, 2017)

Spring es más conocido es por la inyección de dependencias, sin embargo el Framework está dividido en diversos módulos que podemos utilizar, ofreciéndonos muchas más funcionalidades:

- Core container: proporciona inyección de dependencias e inversión de control. (Pahino, 2020)
- Web: nos permite crear controladores Web, tanto de vistas MVC como aplicaciones REST. (Pahino, 2020)
- Acceso a datos: abstracciones sobre JDBC, ORMs como Hibernate, sistemas OXM (Object XML Mappers), JSM y transacciones. (Pahino, 2020)
- Programación orientada a Aspectos (AOP): ofrece el soporte para aspectos. (Pahino, 2020)
- Instrumentación: proporciona soporte para la instrumentación de clases. (Pahino, 2020)
- Pruebas de código: contiene un framework de testing, con soporte para JUnit y TestNG y todo lo necesario para probar los mecanismos de Spring. (Pahino, 2020)

A continuacion se muetra una imagen con la estructura de Spring Framework:

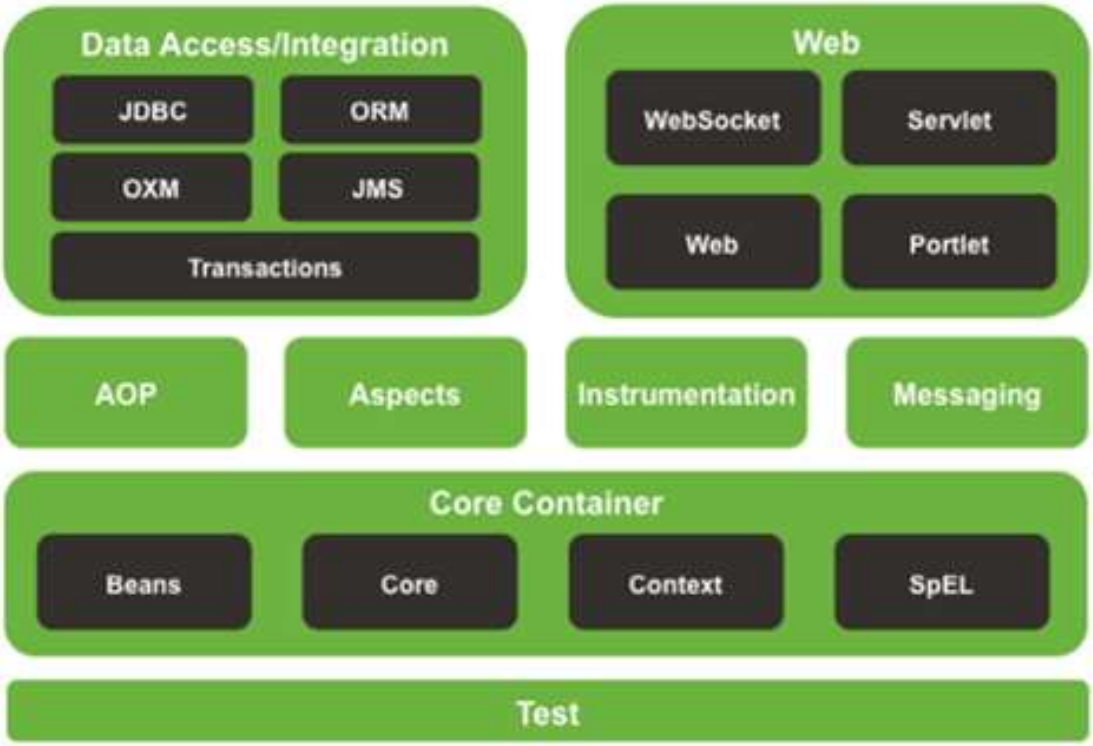


Ilustración 2: Estructura de Spring Framework

FUENTE : (Cuervas, 2019)

Hibernate es una herramienta de mapeo objeto-relacional (ORM) bajo licencia GNU LGPL para Java, que facilita el mapeo de atributos en una base de datos tradicional, y el modelo de objetos de un aplicación mediante archivos declarativos o anotaciones en los beans de las entidades que permiten establecer estas relaciones. Todo lo dicho, que suena a vendedor de seguros, se resume en que agiliza la relación entre la aplicación y nuestra base de datos SQL, de un modo que optimiza nuestro flujo de trabajo evitando caer en código repetitivo. (Equipo Geek, 2019)

Un ORM (Object Relational Mapping o Mapeo Objeto-Relacional en castellano) es una herramienta que nos permite mapear, o lo que es lo mismo, convertir los objetos de tu aplicación a un formato adecuado para ser almacenados en cualquier base de datos,

creándo para ello una base de datos virtual donde los datos disponibles en nuestra aplicación quedan vinculados con la base de datos final. Lo que podemos obtener de la definición anterior, es que además de convertir, los ORM nos ayudan a eliminar todo el lenguaje tedioso de sentencias SQL necesario para realizar las acciones CRUD (Create, Read, Update, Delete) en nuestro código, ya que es el propio ORM quien se encarga de ello. (Romanos, 2019)

Spring Cloud Netflix Eureka es un microservicio REST (Representational State Transfer) que se utiliza con el objetivo de registrar y localizar los microservicios existentes en el ecosistema, informar de su localización, estado y alguna métrica operacional. Facilita el balanceo de la carga entre las instancias de los microservicios y la tolerancia a fallos. En Netflix, es un componente crítico en la distribución de la carga en capas intermedias. En resumen la función de Eureka es registrar las diferentes instancias de microservicios existentes, su localización, estado, metadatos. (z, 2020)

Como vemos al momento de crear grandes aplicaciones web de caracteristicas empresariales son muchas las tecnologias que se ven inmersas, incluso varios lenguajes de programacion pueden dar origen a una misma aplicación web, en este estudio nos cenytramos en la forma en que el lenguaje de programacion Java se ha mantenido a la vanguardia y sigue actualizando herramientas que le permitan competir con lenguajes de programacion del lado del servidor como php o JavaScript, Srping Framwork se ha convertido hoy en día una herramienta indispenzable si se desea desarrollar en el lenguaje de programacion Java, este framework posee caracteristicas que han logrado que el lenguaje se mantega en constante evolucion.

A continucacion se presenta un cuadro comparativo de las principales caracteristicas de los frameworks Hibernate y Spring en el desarrollo de aplicaciones web empresariales.

FRAMEWROKS		
	SPRING	HIBERNATE
FUNCION PRINCIPAL	Spring es completamente modular y soporta diferentes tecnologías como la inyección de dependencias, eventos, recursos, i18n, validación, enlace de datos, conversión de tipo, SpEL.	Herramienta ORM, de mapeo de objeto de relaciones.
ACCESO A DATOS	Soporte DAO, JDBC, ORM, Marshalling XML.	Utiliza su propio lenguaje de consulta denominado HQL.
PARADIGMAS	Programación orientada a aspectos (AOP): permite la implementación de rutinas transversales.	Permite a la aplicación manipular los datos en la base de datos operando sobre objetos, con todas las características de la POO.

Tabla 2: Cuadro comparativo de frameworks Hibernate y Spring

Elaborado por: Diego Viejo.

Ahora presentamos una tabla donde se establecen algunas de las ventajas y desventajas de usar los framework Hibernate y Spring en el desarrollo de aplicaciones web empresariales:

FRAMEWORKS			
	SPRING	HIBERNATE	ÁMBITO EMPRESARIAL
VENTAJAS	Completo soporte de transacciones.	Nos permite desarrollar mucho mas rápido.	Spring ya no solo se encarga de la inyeccion de dependencias, ha liberado proyectos que abarcan mucho ambitos: desarrollo de aplicaciones web, aplicaciones web reactivas, seguridad, servicios web, microservicios, Android,etc. Spring Framework proporciona un modelo integral de programación y configuración para aplicaciones empresariales modernas basadas en Java, en cualquier tipo de plataforma de implementación. Hibernate y Spring son técnologias muy complementarias
	Permite pruebas unitarias y de integracion.	Permite trabajar con la base de datos por medio de entidades en vez de Querys.	
	Lógica aplicable con POJO.	Nos ofrece un paradigma 100% orientado a objetos.	
	Gestion pragmática de transacciones.	Mejora el mantenimiento del software.	
DESVEN-TAJAS	Configuracion xml mas compleja.	No ofrece toda la funcionalidad que ofrecería tirar consultas nativas.	
	No se puede evaluar si un objeto ha sido bien inyectado más que en tiempo de ejecución.	El performance es mucho mas bajo que realizar las consultas por JBDC.	
	El contenedor de Spring no es ligero.	Puede representar una curva de aprendizaje mas grande.	

Tabla 3: Ventajas y desventajas de los frameworks Hibernate y Spring.

Elaborado por: Diego Viejo.

CONCLUSIONES

Las aplicaciones web hoy en día constituyen una nueva forma de hacer negocios rentables, cada día son más las grandes aplicaciones que brindan servicios y se hacen conocidas a nivel mundial.

Entre más grandes son las aplicaciones más son los leguajes de programación que se integran para satisfacer las necesidades de los clientes, así mismo cada vez son más las nuevas soluciones que surgen a partir de las necesidades que se van desarrollando a lo largo del tiempo.

Los framework constituyen una herramienta muy importante para los programadores, ya que establecen nuevas formas de desarrollo, estos a su vez son apoyados por la comunidad de programadores, por ende, su crecimiento y mejoramiento es siempre constante.

Las soluciones que pueden brindar una aplicación web pueden ser muchas, todo depende de las necesidades que surgen a lo largo del tiempo, entre más funciones o necesidades satisfaga una aplicación web, mayor será la complejidad de su código fuente, es ahí donde los frameworks juegan un papel importante ya la principal funcion de los Frameworks es hacer que el proceso de desarrollo sea mucho más fácil y escalable.

Actualmente en la práctica dela programación es muy común implementar e integrar varias herramientas entre ellas frameworks, así mismo, si se usa la infraestructura de microservicios incluso se pueden integrar diferentes tipos de lenguajes de programación para aprovechar las ventajas que puede ofrecer cada uno en ámbito determinado.

Bibliografía

Cuervas, J. (18 de Diciembre de 2019). ¿Qué es Spring Framework? Características I. Obtenido de atsistemas: https://www.atsistemas.com/blog/qu-es-spring-framework-caractersticas-i

Equipo Geek. (06 de Agosto de 2019). ¿Qué es Java Hibernate? ¿Por qué usarlo? Obtenido de if geek then: https://ifgeekthen.everis.com/es/que-es-java-hibernate-por-que-usarlo

Fontanet, B. (12 de Julio de 2016). Java EE y el desarrollo web: Un enfoque de aprendizaje. Obtenido de Fundesem Business Topics: https://www.fundesem.es/bt/publicacion-java-ee-y-el-desarrollo-web-un-enfoque-de-aprendizaje

GreenSQA. (17 de Enero de 2019). Arquitectura de Microservicios vs Arquitectura Monolítica. Obtenido de GreenSQA: https://greensqa.com/arquitectura-de-microservicios-vs-arquitectura-monolitica/

Lewis, J., & Fowler, M. (25 de Marzo de 2014). Microservices. Obtenido de martinFowler.com: https://martinfowler.com/articles/microservices.html

López , D., & Maya , E. (20 de Julio de 2017). Arquitectura de Software basada en Microservicios para Desarrollo de Aplicaciones Web. Obtenido de Dspace: https://dspace.redclara.net/bitstream/10786/1277/1/93%20Arquitectura%20de%20Software%20basada%20en%20Microservicios%20para%20Desarrollo%20de%20Aplicaciones%20Web.pdf

Moreno, J. (06 de Diciembre de 2017). Tipos de inyección de dependencias con Spring. Obtenido de proitc solution: https://proitcsolution.com.ve/inyeccion-de-dependencias-spring/

Novoseltseva, E. (05 de Marzo de 2018). BENEFICIOS Y EJEMPLOS DE LA IMPLEMENTACIÓN DE LOS MICROSERVICIOS. Obtenido de Apiumhub: https://apiumhub.com/es/tech-blog-barcelona/los-microservicios/#:~:text=Casos%20de%20%C3%A9xito%20y%20ejemplos,monol%C3%ADtica%20a%20la%20de%20microservicios.

Pahino, R. (31 de Marzo de 2020). ¿Qué son Spring framework y Spring Boot? Tu primer programa Java con este framework. Obtenido de campus mvp: https://www.campusmvp.es/recursos/post/que-son-spring-framework-y-spring-boot-tu-primer-programa-java-con-este-framework.aspx

Palumbo, F., & Tunessi, S. (octubre de 2013). Arquitectura de Servicios Web basada en modelos: especificación gráfica y derivación automática del código. Obtenido de sedici: http://sedici.unlp.edu.ar/bitstream/handle/10915/63225/Documento_completo.pdf-PDFA2u.pdf?sequence=1&isAllowed=y

Ramos Chagoya, E. (1 de Julio de 2018). Métodos y técnicas de investigación. Obtenido de gestiopolis: https://www.gestiopolis.com/metodos-y-tecnicas-de-investigacion/

Robledano, Á. (12 de Agosto de 2019). Qué es Java: Principios básicos y evolución. Obtenido de OpenWebinars: https://openwebinars.net/blog/que-es-java/

Rodríguez, A. (07 de Mayo de 2018). Microservicios 2.0: Spring Cloud Netflix vs Kubernetes & Istio. Obtenido de Paradigma Digital: https://www.paradigmadigital.com/dev/microservicios-2-0-spring-cloud-netflix-vs-kubernetes-istio/

Romanos, M. (04 de Enero de 2019). ¿Qué es un ORM? Obtenido de el blog: https://www.dreams.es/transformacion-digital/desarrolladores-paginas-web/que-es-un-orm

z, K. (04 de Abril de 2020). Netflix y la nueva arquitectura de microservicios. Obtenido de Joseph a software developer: https://josephcodes.dev/2020/11/04/netflix-y-la-nueva-arquitectura-de-microservicios/

ANEXOS

ANEXO 1: Encuesta.

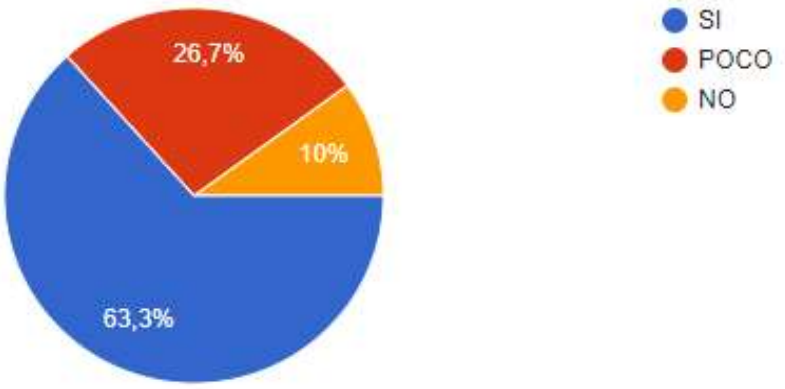
UNIVERSIDAD TECNICA DE BABAHOYO

FACULTAD DE ADMINISTRACIÓN, FINANZAS E INFORMÁTICA

Tabulación de Encuesta dirigida a personas de la comunidad de programadores.

1. **¿Usa el lenguaje Java para el desarrollo de aplicaciones web?**

Ítems	**Frecuencia**	**porcentaje**
Si	19	63,3%
Poco	8	26,7%
No	3	10%
Total	**30**	**100%**

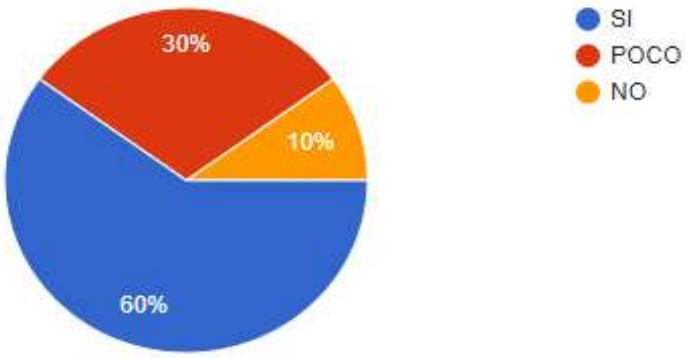


Fuente: Comunidad de programadores.

Análisis: el lenguaje de programación java tiene una importante acogida en el desarrollo web.

2. **¿Hace uso de frameworks de java para el desarrollo web?**

Ítems	Frecuencia	porcentaje
Si	18	60%
Poco	9	30%
No	3	10%
Total	**30**	**100%**

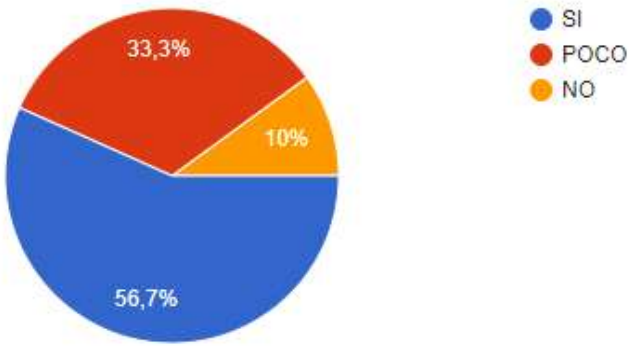


Fuente: Comunidad de programadores.

Análisis: los frameworks tienden a tener mayor usabilidad.

3. **¿Conoce el framework Spring?**

Ítems	Frecuencia	porcentaje
Si	17	56,7%
Poco	10	33,3%
No	3	10%
Total	**30**	**100%**

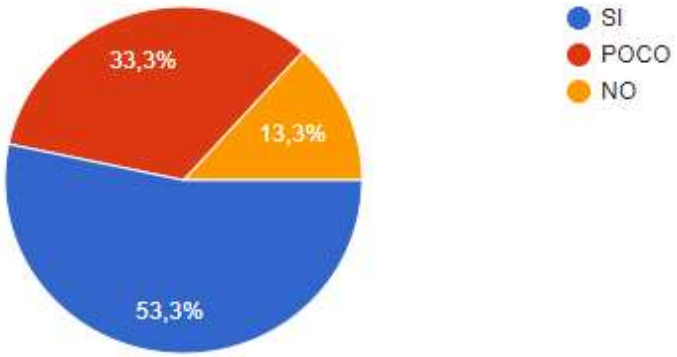


Fuente: Comunidad de programadores.

Análisis: Spring es muy usado por los desarrolladores java.

4. **¿Conoce el framework Hibernate?**

Ítems	Frecuencia	porcentaje
Si	16	53,3%
Poco	10	33,3%
No	4	13,3%
Total	**30**	**100%**

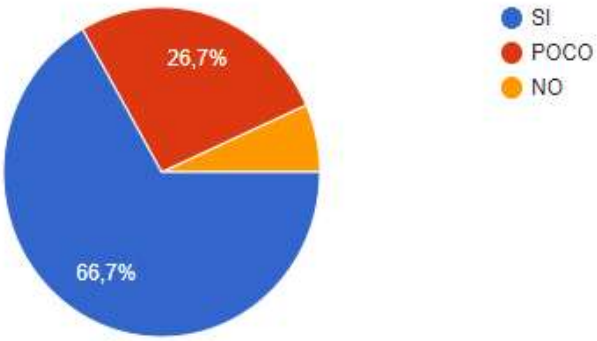


Fuente: Comunidad de programadores.

Análisis: Hibernate es usado para acceso a datos.

5. **¿La complejidad del desarrollo de aplicaciones web disminuye con el uso de frameworks?**

Ítems	Frecuencia	porcentaje
Si	20	66,7%
Poco	8	26,7%
No	2	6,7%
Total	**30**	**100%**

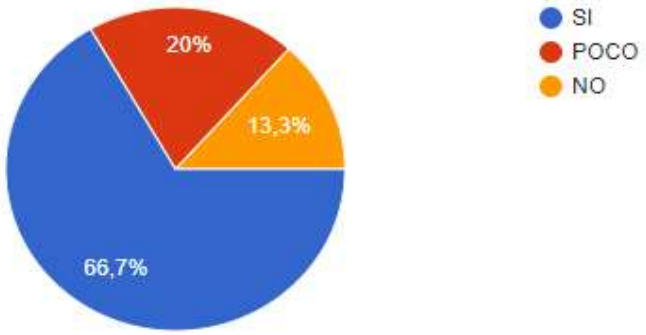


Fuente: Comunidad de programadores.

Análisis: los frameworks tienden a ser más usados.

6. **¿Recomienda el uso de frameworks para el desarrollo de aplicaciones web?**

Ítems	Frecuencia	porcentaje
Si	20	66,7%
Poco	6	20%
No	4	13,3%
Total	**30**	**100%**

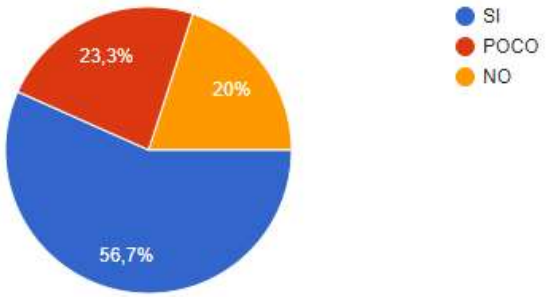


Fuente: Comunidad de programadores.

Análisis: al momento de desarrollar aplicaciones web se usan muchos frameworks.

7. **¿Conoce acerca de la inyección de dependencias?**

Ítems	Frecuencia	porcentaje
Si	17	56,6%
Poco	7	23,3%
No	6	20%
Total	**30**	**100%**

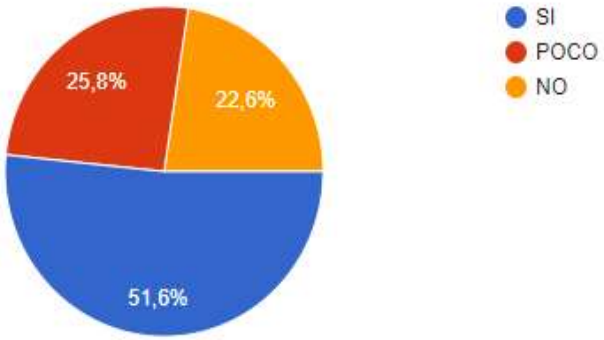


Fuente: Comunidad de programadores.

Análisis: la inyección de dependencias es muy usada.

8. ¿Conoce acerca de HQL?

Ítems	Frecuencia	porcentaje
Si	16	51,6%
Poco	8	25,8%
No	6	22,6%
Total	**30**	**100%**

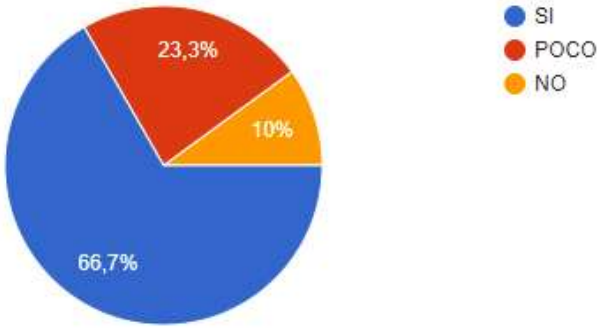


Fuente: Comunidad de programadores.

Analisis: hql es no tiene tanto uso.

9. ¿Conoce la infraestructura de microservicios?

Ítems	Frecuencia	porcentaje
Si	20	66,7%
Poco	7	23,3%
No	3	10%
Total	**30**	**100%**

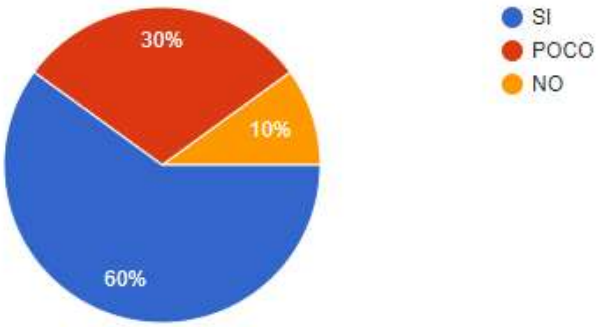


Fuente: Comunidad de programadores.

Análisis: la infraestructura de microservicios es muy implementada.

10. ¿Recomienda usar java como lenguaje backend en el desarrollo de aplicaciones web?

Ítems	Frecuencia	porcentaje
Si	18	60%
Poco	9	30%
No	3	10%
Total	**30**	**100%**



Fuente: Comunidad de programadores.

Análisis: el lenguaje java es muy usado en el lado del servidor.

ANEXO 2: Árbol del problema

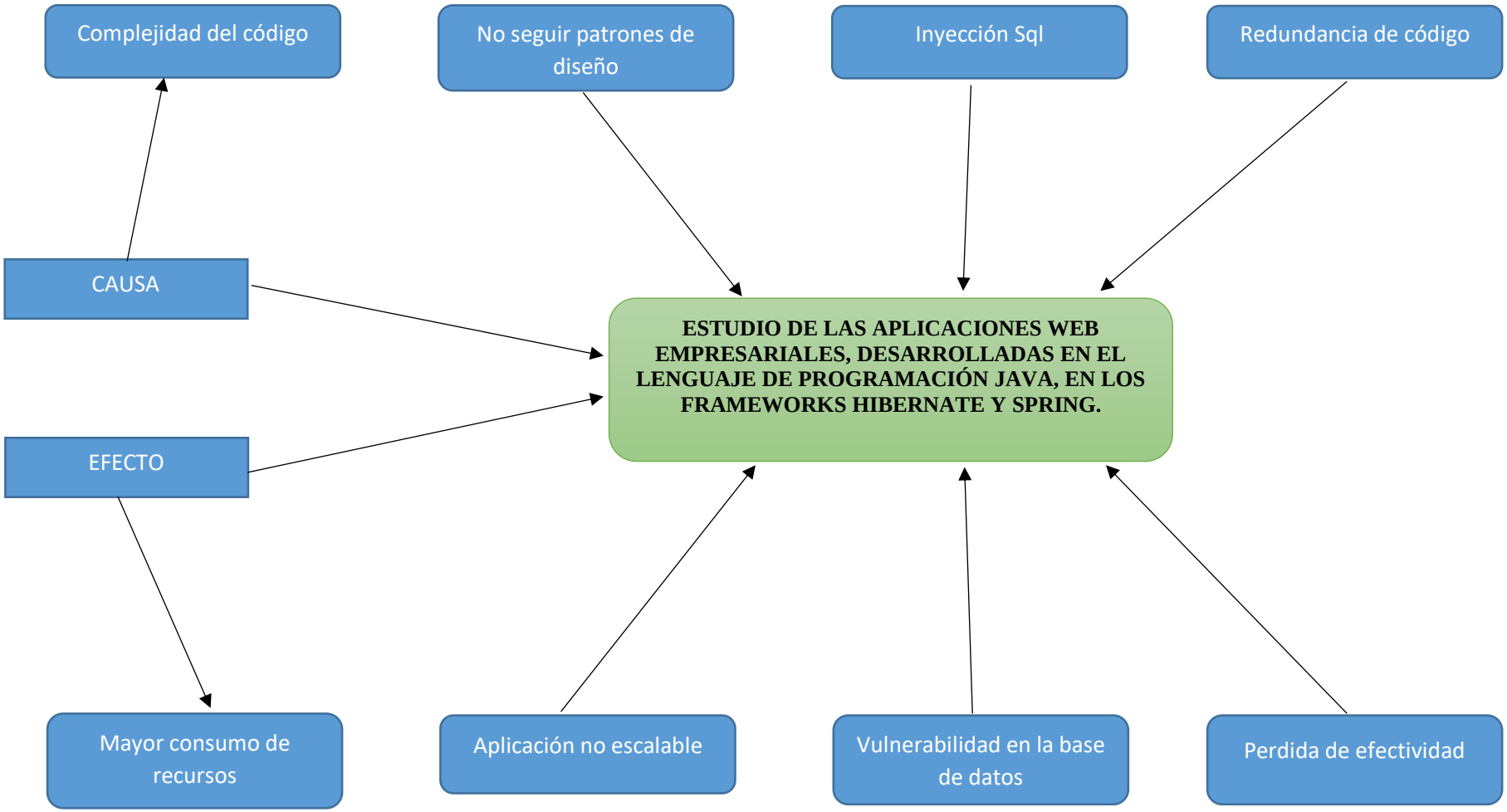


Ilustración 3 árbol del problema

Elaborado por: Diego Viejo