IMPLEMENTACIÓN DE ARQUITECTURAS SOA Y MICROSERVICIOS PARA LA INTEGRACIÓN CONTINUA DE APLICACIONES

IMPLEMENTATION OF SOA ARCHITECTURES AND MICROSERVICES FOR CONTINUOUS APPLICATION INTEGRATION

Miriam Zulma Sánchez Hernández
Tecnológico Nacional de México / IT de Morelia, México
miriam.sh@morelia.tecnm.mx

Abel Alberto Pintor Estrada
Tecnológico Nacional de México / IT de Morelia, México
abel.pe@morelia.tecnm.mx

María Yaneth Vega Flores
Tecnológico Nacional de México / IT de Morelia, México
maria.vf@morelia.tecnm.mx

Claudio Ernesto Florian Arenas
Tecnológico Nacional de México / IT de Morelia, México
claudio.fa@morelia.tecnm.mx

Mario Huerta Olivares
Tecnológico Nacional de México / IT de Morelia, México
l18121499@morelia.tecnm.mx

Recepción: 24/noviembre/2022 **Aceptación:** 20/diciembre/2022

Resumen

El desarrollo de software implica una arquitectura que indique su estructura, funcionamiento e interacción entre sus componentes. Actualmente la mayoría de los sistemas desarrollados utilizan una arquitectura monolítica, implicando que toda la funcionalidad del sistema (acceso de datos, interfaz de usuario, lógica, etc.) está implementada en una sola unidad ejecutable; esto ocasiona problemas en aspectos como mantenimiento, escalabilidad y entregas. El objetivo de este artículo es presentar la implementación de arquitecturas SOA y de Microservicios en el desarrollo de software como una propuesta de acoplamiento flexible, con interfaces independientes del hardware, del sistema operativo y del lenguaje de programación,

permitiendo que las unidades funcionales de aplicaciones (servicios) se construyan sobre sistemas heterogéneos, aumentando la agilidad y velocidad para hacer cambios. Lo anterior a través del desarrollo de un software con arquitectura SOA y otro con arquitectura de Microservicios, ejemplificando las diferencias entre estas arquitecturas y la propuesta de una integración más rápida, eficaz y continua de servicios dentro de un software.

Palabras Clave: Aplicaciones, Arquitectura, Microservicios, SOA.

Abstract

Software development involves an architecture that indicates its structure, operation, and interaction between its components. Currently most of the systems developed use a monolithic architecture, implying that all the functionality of the system (data access, user interface, logic, etc.) is implemented in a single executable unit; this causes problems in aspects such as maintenance, scalability, and deliverables. The objective of this article is to present the implementation of SOA and Microservices architectures in software development as a proposal of flexible coupling, with interfaces independent of the hardware, the operating system, and the programming language, allowing that the functional units of the applications (services) are built on heterogeneous systems, increasing agility and speed to make changes. The above through the development of a software with SOA architecture and another with Microservices architecture, exemplifying the differences between These architectures and the proposal of a faster, more efficient, and continuous integration of services within a software.

Keywords**:** *Applications, Architecture, Microservices, SOA.*

1. Introducción

Dentro del desarrollo de software, los sistemas de gestión de información son aquellos que requieren administrar, en muchas ocasiones, grandes cantidades de datos, así también necesitan mantener interfaces con altos niveles de usabilidad para usuarios con diferentes niveles de conocimientos técnicos. Sin embargo, existe una evidente problemática en el desarrollo de aplicaciones de software, ya que la

mayoría utiliza una arquitectura monolítica que incide en diferentes aspectos tanto tecnológicos como de eficiencia en el uso de recursos. Los problemas más evidentes se pueden observar al aplicar procesos de mantenimiento en sistemas complejos, sea por una petición de cambio, nueva funcionalidad o corrección de un fallo, en los cuáles la resolución de un problema o simple cambio implica el redespliegue de toda la aplicación debido a que se tienen todas las funcionalidades en un único paquete incrementando los riesgos de fallos.

Después de analizar la problemática ya planteada, se ha determinado la necesidad de proponer una arquitectura de software con un enfoque que facilite el desarrollo de nuevas aplicaciones, y que permita la creación, el desarrollo y mantenimiento de aplicaciones, evitando la interrupción de actividades de los usuarios que las utilizan. Esto es, una Arquitectura Orientada a Servicios (SOA), o mejor aún, una Arquitectura de Microservicios.

Arquitectura Orientada a Servicios y Arquitectura de Microservicios

La Arquitectura Orientada a Servicios (SOA) permite que los sistemas de información contengan componentes reusables implementados como interfaces de servicio [IBM, 2019]. Por ejemplo, podemos implementar una vista que utilice un buscador que utilice filtros sobre diferentes entidades de nuestra base de datos.

La figura 1 muestra la manera en que la arquitectura orientada a servicios se implementa, y cómo esta define la interfaz de usuario, así también, implementa su lógica de negocio, y permite el acceso a los datos. Sin embargo, mantiene un nivel de modularidad que permite reutilizar componentes para crear nuevos servicios.

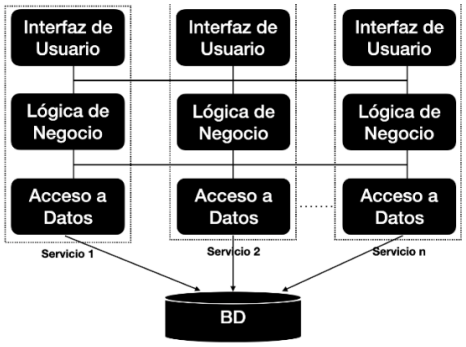

Figura 1 Arquitectura Orientada a Servicios (SOA).

Por otro lado, la Arquitectura de Microservicios se utiliza para desarrollar aplicaciones como un conjunto de servicios pequeños [De la Torre, 2022]. Una característica fundamental de estos servicios es que deben estar totalmente desacoplados, es decir, deben de utilizar su propio modelo de datos y cada servicio ejecutar su propio proceso de lógica de negocios. Por tanto, permite modularizar un sistema, de tal manera que se puedan implementar mecanismos de alta disponibilidad en aquellos componentes prioritarios.

Generalmente, una arquitectura de microservicio se utiliza para implementar el back-end de un servicio [Wang,2018], es decir, desarrolla la lógica de negocio y el acceso a los datos. Puede observarse también, que cada microservicio cuenta con su propio modelo de datos, figura 2.

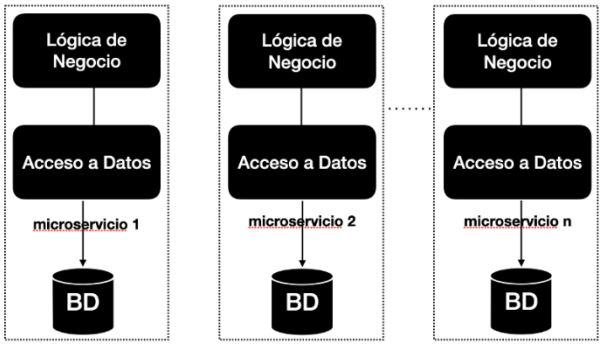

Figura 2 Arquitectura de Microservicios.

El Tecnológico Nacional de México, campus Instituto Tecnológico de Morelia, ha implementado un programa institucional denominado Fábrica Académica de Software (FAS), la cual, tiene el fin de proporcionar un ambiente de desarrollo de software real vinculando a los estudiantes con diferentes empresas o proyectos, que les permitan obtener una primera experiencia laboral dentro del recinto universitario [Sánchez Hernández, 2018].

En el presente documento se presentan dos casos de estudio de proyectos de desarrollo de software implementados por equipos de la FAS. El primer proyecto es un software de gestión administrativa para la División de Estudios Profesionales del Instituto Tecnológico de Morelia y el segundo, un sistema de gestión de proyectos de edición gráfica para la empresa “Tecnicámara es todo en fotografía, SA de CV”.

2. Métodos

1-A. Descripción del primer Caso de Estudio "DEP"

La División de Estudios Profesionales (DEP) del Instituto Tecnológico de Morelia requería de un software que les permitiera gestionar los trámites de alumnos que solicitaban traslados, equivalencias, solicitudes a comité académico, etc. Esta gestión abarcaba otros departamentos que participaban en el proceso de atención. Los alumnos, sin necesidad de registrarse en el sistema, podían realizar diversas solicitudes y ver en tiempo real el estatus de estas.

1-B. Análisis y selección de la Arquitectura de Software DEP

Primero, se analizaron los requerimientos técnicos, entre estos, la cantidad de solicitudes de alumnos, que, en promedio, eran de 475 solicitudes por semestre y no es considerado un servicio que requiera alta disponibilidad. La interfaz de usuario debía ser amigable, tanto para estudiantes, como para el personal del Instituto que iba a operar el proceso a través del sistema. Por lo anterior, se determinó utilizar una base de datos centralizada, implementándose en Postgresql. También se observó que varios componentes requerían implementar interfaces comunes, como el caso de las interfaces de búsqueda y filtrado.

Finalmente, se determinó que la arquitectura mas adecuada para este sistema era implementar la Arquitectura Orientada a Servicios, utilizando como framework de desarrollo a Django, que se ajusta perfectamente a dicha arquitectura. En la figura 3, tenemos un fragmento de la arquitectura que se utilizó para el sistema DEP donde podemos observar que esta arquitectura nos permite utilizar, por ejemplo, la lógica de negocios del controlador send_mail con las interfaces Progreso y Documentos.

2-A. Descripción del segundo Caso de Estudio "Tecnicámara"

La empresa Tecnicámara requería de un sistema web que implemente un kiosko digital que permita ofrece diferentes servicios de edición gráfica, serigrafía, impresión de fotos, etc. El sistema debe ofrecer los diferentes servicios citados arriba e invertir esfuerzos en los módulos mas utilizados aplicando alta disponibilidad para garantizar que el usuario pueda accederlos. Los clientes

deberán registrarse en el sistema para poder comenzar un proyecto gráfico y poder retomarlo en cualquier momento.

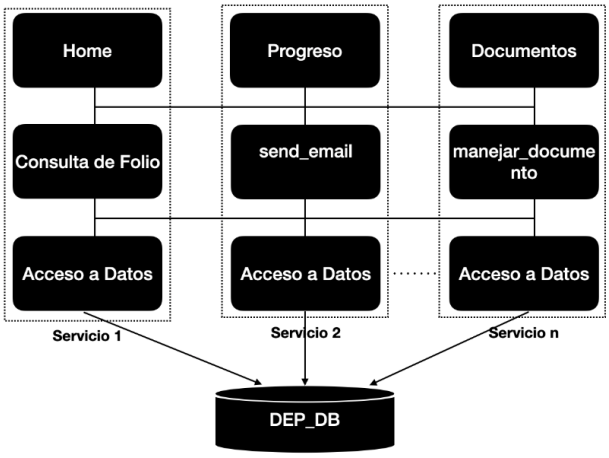

Figura 3 Arquitectura Orientada a Servicios aplicada al software DEP.

2-B. Análisis y selección de la Arquitectura de Software para el Sistema Tecnicámara

Al igual que en el caso anterior, se analizaron los requerimientos funcionales y técnicos, entre los que se destacan; número de peticiones esperadas por día de 50, los módulos más utilizados son, serigrafía con un 60% de las peticiones, impresión de fotografía con un 30% de las peticiones. La interfaz de usuario debería ser amigable y con alto grado de fiabilidad y disponibilidad para no poner en riesgo la imagen de la empresa.

Dadas las características mencionadas anteriormente se decidió utilizar tanto la arquitectura orientada a servicios como la arquitectura de microservicios. La primera se utilizó en todos los módulos que no requerían alta disponibilidad y, la segunda, únicamente con los servicios de serigrafía e impresión de fotos. Por tanto, la decisión también se tomó en base a la determinación de que no todo el sistema requería microservicios lo que implicaría acortar el tiempo de desarrollo sin perjudicar la fiabilidad y disponibilidad del sistema. En la figura 4 se presenta parte de la arquitectura del sistema de kiosko digital Tecnicámara en donde podemos observar la manera en que pueden convivir ambas arquitecturas, con una base de datos distribuida de manera vertical. Otra ventaja del uso de la arquitectura

planteada es la posibilidad de escalado horizontal de los microservicios, en los momentos en que se detecte un alza significativa de peticiones en esos servicios. Finalmente, el sistema Tecnicámara se implementó utilizando el framework de desarrollo Node en conjunto con el framework express, y para el manejo de las bases de datos se utilizó Postgresql.

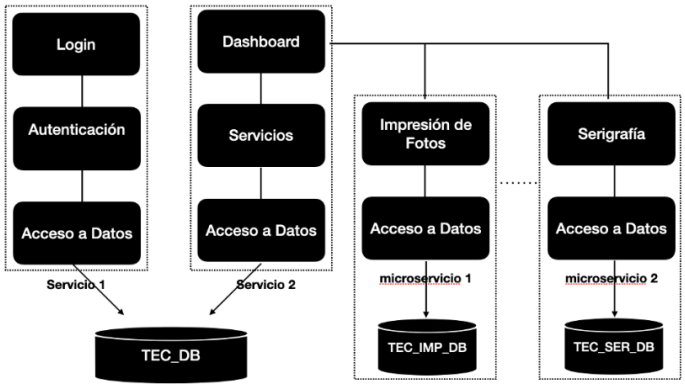

Figura 4 Arquitectura del sistema web Tecnicámara.

3. Resultados

El visualizar alternativas de arquitecturas para el desarrollo de Software e implementarla en proyectos de la Fábrica Académica de Softwawre, ha generado diversos resultados que han sido muy valiosos e impresindibles para los siguientes proyectos de desarrollo de software. Algunos de estos hallazgos son:

- La implementación de SOA y arquitecturas de Microservicios han proporcionado una mayor flexibilidad para la gestión de cambios y han permitido que tanto docentes como estudiantes que desarrollan proyectos dentro de la FAS, tengan una visión más amplia al determinar el tipo de arquitectura apropiada para cada proyecto de software, afirmando ahora que la arquitectura de microservicios es un derivado de la arquitectura orientada a servicios, que implementa unidades pequeñas pero independientes que corren de manera concurrente, lo que mejora la mantenibilidad y el rendimiento del sistema.
- Al implementar desarrollo de software utilizando una arquitectura con Microservicios se ha confirmado la flexibilidad de tener cada microservicio

independiente, incluso del lenguaje en que es implementado, por lo que se pueden realizar desarrollos a la par sin que los equipos de programación tengan algún problema entre ellos o se colisionen actividades. Esto ha garantizado que puedan implementarse varios servicios en un mismo tiempo, acortando los tiempos y la complejidad del desarrollo.

- Por otro lado, uno de los resultados relevantes en esta implementación es que no siempre se deben realizar implementaciones solo basadas en SOA o solo basadas en microservicios, como ser observó en la figura 4, es posible desarrollar un sistema de información utilizando como base la arquitectura SOA, y para aquellos componentes que requieren alta disponibilidad con un bajo nivel de acoplamiento, se pueden desarrollar bajo la arquitectura de microservicios.

4. Discusión

¿Qué arquitectura escoger, servicios o microservicios? ¿Qué diferencia hay entre las dos arquitecturas? Estas son algunas de las preguntas que los desarrolladores siempre se hacen cuando deben diseñar la arquitectura de un software de gestión de información, y para entender su diferencia, se debe comprender que la arquitectura de Microservicios es un derivado de la arquitectura orientada a servicios (SOA), que implementa unidades pequeñas pero independientes que corren de manera concurrente, lo que mejora la mantenibilidad y el rendimiento del sistema.
Ahora bien, con una arquitectura SOA, se pueden desarrollar sistemas de gestión de información que contengan componentes que puedan interrelacionarse y acceder a una base de datos centralizada, esto en esencia (y según nuestra experiencia) es un sistema monolítico, al que se le agregan interfaces de comunicación estándar.

5. Conclusiones

La implementación de una arquitectura SOA o de Microservicios ha permitido iniciar un cambio importante en el desarrollo de aplicaciones que pueden ser optimizadas, escaladas y modificadas sin tener que inhabilitar todos los servicios.

Dicho en otras palabras, ahora no es necesario tener varias aplicaciones web que presten los diferentes servicios que se necesitan, sino que todos estarán concentrados en la misma plataforma; lo que también coadyuva a minimizar costos en los servicios, pues esta herramienta puede reusar compontes comunes en sus diferentes servicios o aplicaciones que componen la plataforma de aplicaciones.

Por otro lado, se puede establecer que las arquitecturas de microservicios hacen que las aplicaciones sean más fáciles de escalar y más rápidas de desarrollar, por lo que consideramos es un elemento fundamental de la optimización en el desarrollo de aplicaciones hacia un modelo nativo de la nube. La flexibilidad de tener cada microservicio independiente, incluso del lenguaje en que es implementado, hace que se puedan realizar desarrollos a la par sin que los equipos de programación tengan algún problema entre ellos o se colisionen actividades. Esto ha garantizado que puedan implementarse varios servicios en un mismo tiempo, acortando los tiempos y la complejidad del desarrollo.

Finalmente, se puede concluir que la Arquitectura Orientada a Servicios y la Arquitectura de Microservicios pueden ser complementarias, y su uso dependerá del nivel de desacoplamiento que necesitemos para cada servicio.

6. Bibliografía y Referencias

[1] IBM Cloud Education. (2019, July 17). ¿Qué es la SOA (arquitectura orientada a servicios)? Https://Www.Ibm.Com/Mx-Es/Cloud/Learn/ Soa.

[2] De La Torre, C., Wagner, B., & Rousos, M. (2022). .NET Microservices: Architecture for Containerized .NET Applications (M. Pop & S. Hoag, Eds.; 6a. Ed.). Microsoft Developer Division, .NET and Visual Studio product teams.

[3] Sánchez Hernández, M. Z., Vega Flores, M. Y., & Pintor Estrada, A. A. (2018). Experiencia laboral universitaria a través de la Implementación de una fábrica académica de software. Compendio Investigativo de Academia Journals Celaya 2018, 5039–5044.

[4] Wang, F.-J., & Fahmi, F. (2018). Constructing a Service Software with Microservices. 2018 IEEE World Congress on Services (SERVICES), 43–44. https://doi.org/10.1109/SERVICES.2018.00035.