RESUMEN

DESARROLLO ÁGIL DEL NUEVO SISTEMA INSTITUCIONAL BASADO EN UNA ARQUITECTURA ORIENTADA A MICROSERVICIOS

por

Abraham Hernández Rivadeneyra

Asesor principal: Alejandro Walterio García Mendoza

RESUMEN DE TESIS DE MAESTRÍA

Universidad de Montemorelos

Facultad de Ingeniería y Tecnología

Título: DESARROLLO ÁGIL DEL NUEVO SISTEMA INSTITUCIONAL BASADO EN UNA NUEVA ARQUITECTURA ORIENTADA EN MICROSERVICIOS.

Investigador: Abraham Hernández Rivadeneyra

Asesor principal: Alejandro Walterio García Mendoza, Maestría en Tecnologías de la Información

Fecha de culminación: Mayo de 2019

Problema

En esta investigacion se pretende dar solucion a la hipotesis planteada, en cuanto a demostrar que es más agil el desarrollo del nuevo sistema institucional basado en una nueva arquitectura de microservicios.

Metodología

En esta investigación, para llevar a cabo la comprobación de que el desarrollo de la nueva arquitectura es mucho más ágil que el anterior, se llevó a cabo la implementación de la metodología de puntos de casos de uso para la obtención del peso que tiene cada módulo. Después del peso obtenido, se necesita saber las horas que

tuvo cada desarrollo, el cual es obtenido por medio de bitácoras donde se describen las horas realizadas.

Resultados

Se llevó la implementación de esta arquitectura en las inscripciones del mes de enero, lo cual comprueba su funcionalidad. A partir de esa fecha, cualquier pago realizado de los servicios de la institución fue realizado por medio del nuevo sistema. En cuanto al desarrollo con una arquitectura en microservicios, se obtuvo que es más ágil que el anterior.

Conclusión

El resultado obtenido en esta investigación comprueba que por medio de la implementación de esta nueva arquitectura se puede agilizar el desarrollo de los módulos faltantes al sistema institucional o de cualquier sistema nuevo.

Universidad de Montemorelos

Facultad de Ingeniería y Tecnología

DESARROLLO ÁGIL DEL NUEVO SISTEMA INSTITUCIONAL BASADO EN UNA ARQUITECTURA ORIENTADA A MICROSERVICIOS

Tesis
presentada en cumplimiento parcial
de los requisitos para el grado de
Maestría en Ciencias Computacionales

por

Abraham Hernández Rivadeneyra

Mayo 2019

DESARROLLO ÁGIL DEL NUEVO SISTEMA INSTITUCIONAL
BASADO EN UNA ARQUITECTURA ORIENTADA
A MICROSERVICIOS

Proyecto

presentado en cumplimiento parcial de
los requisitos para el grado de
Maestría en Ciencias
Computacionales

por

Abraham Hernández Rivadeneyra

APROBADO POR LA COMISIÓN:

M.C. Alejandro Walterio García Mendoza
Asesor principal

Mtro. Carlos Alberto Guzmán Valenzuela
Asesor externo

M.C. Saulo Hernández Osoria
Miembro

Dr. Ramón Andrés Díaz Valladares
Director de Posgrado e Investigación

M.C. Daniel Arturo Gutiérrez Colorado
Miembro

3 de mayo de 2019
Fecha de aprobación

DEDICATORIA

Primero que nada, agradezco a mi Dios por esta oportunidad de aprender cada vez más y llegar hasta donde me encuentro en este momento.

A mis padres Abrahan Hernández Casados y María Rivadeneyra Antele, por su apoyo incondicional en todo tiempo.

A mi hermana que siempre estuvo dándome ánimos para concluir esta nueva etapa.

A cada persona que me ayudó y aconsejó en el transcurso de esta investigación.

TABLA DE CONTENIDO

LISTA DE FIGURAS

LISTA DE TABLAS

RECONOCIMIENTOS

Agradezco primero a Dios, porque Él es el proveedor de todo el conocimiento, y porque me ha guiado en cada etapa de la vida.

Al mi asesor principal, el maestro Alejandro Walterio García Mendoza, por el tiempo dedicado en la revisión y la dirección de mi tesis.

Al maestro Saulo Hernández Longoria, como asesor secundario en esta investigación.

Al maestro Daniel Gutiérrez, como coordinador del postgrado en cuanto a la orientación presentada durante el curso de la maestría.

Al ingeniero Omar Soto, que me apoyó incondicionalmente durante el desarrollo del proyecto.

A cada persona que tomó parte de su tiempo en ayudarme con la revisión del documento.

A cada uno de los maestros que me impartieron clases, una parte fundamental de mi desarrollo en esta nueva etapa.

A cada uno de mis compañeros de la maestría, ya que me acompañaron en esta etapa de mi vida.

A mis padres y mi hermana, que me apoyaron y me motivaron a trabajar arduamente y a concluir este trabajo de la mejor manera.

CAPÍTULO I

ANTECEDENTES DEL PROBLEMA

Planteamiento del problema

Actualmente, el proceso de desarrollo que se lleva a cabo en la Dirección de Tecnología Informática de la Universidad de Montemorelos corresponde a una arquitectura tradicional o también conocida como monolítica, la cual está compuesta por módulos que son adaptados de acuerdo con los requerimientos que van siendo necesarios día con día. Esta arquitectura no facilita la rápida construcción de una aplicación y vuelve cada vez más difícil el mantenimiento de dicho desarrollo.

Se pretende dar solución en cuanto a demostrar que es más ágil el desarrollo del nuevo sistema institucional basado en una nueva arquitectura orientada a los microservicios.

Declaración de problema

El problema a investigar en este estudio fue el siguiente:

La Universidad de Montemorelos cuenta con una arquitectura monolítica dentro de los sistemas actuales. En lo que se ha podido observar, esta proporciona una fácil implementación y una escalabilidad limitada dentro de los márgenes del desarrollo ya existente.

Entre las desventajas de este tipo de implementación se encuentran las siguientes:

1. Arquitecturas que generan aplicaciones grandes y eso las hace complejas al momento de querer realizar algunos cambios o mejoras.

2. La necesidad de implementar toda la aplicación para cada una de las actualizaciones.

3. Al momento de escalar, se pueden presentar algunos conflictos por tratarse de módulos distintos.

4. Es una barrera querer adoptar nuevas tecnologías dentro de las implementaciones actuales.

Por medio del presente estudio se quiere demostrar lo ágil de un nuevo tipo de desarrollo, así como el proceso que se lleva para la creación de una arquitectura orientada a microservicios, mostrando cómo se desarrolla la implementación en un módulo del sistema institucional de la Universidad de Montemorelos, tomando en cuenta las siguientes características para una buena implementación y un mejor funcionamiento:

1. La construcción de los servicios debe ser pequeña.

2. Debe permitir la escalabilidad del sistema de manera independiente, sin afectar el funcionamiento de todo el sistema.

3. Debe hacer que la funcionalidad de cada servicio sea única.

4. Debe poder implementar tecnologías nuevas o de cualquier tipo.

Justificación

La Universidad de Montemorelos requiere de una arquitectura robusta, moderna, que cumpla con estándares internacionales actuales y que agilice la implementación del sistema institucional. El montar el nuevo desarrollo sobre esta arquitectura le permitirá a la dirección de tecnología informática generar software de una manera

más ágil y con una mayor capacidad de adaptación, en comparación con desarrollos actuales. Además, se pretende que la dirección de tecnología informática de la Universidad de Montemorelos llegue a ser un proveedor de soluciones de la Iglesia Adventista del Séptimo Dia (IASD) en México, orientadas bajo este tipo de esquema de desarrollo.

Objetivos

Para llevar a cabo el desarrollo y la implementación de una arquitectura basada en microservicios se plantearon ciertos objetivos a cumplir, estos están divididos en objetivo general y objetivos específicos.

Objetivo general

Para la presente investigación se planteó el siguiente objetivo general:

Dejar establecida de manera global la arquitectura del nuevo sistema institucional en la Universidad de Montemorelos e implementarla en el módulo de caja, demostrando que el desarrollo de la arquitectura de microservicios es más ágil que la arquitectura tradicional con la que se cuenta actualmente en la Universidad de Montemorelos.

Objetivos específicos

A continuación se plantean los siguientes objetivos específicos:

1. Definir las capas con las que cuenta la arquitectura de microservicios del sistema institucional.

2. Determinar las tecnologías con las que se van a desarrollar cada una de las capas.

3. Definir la administración de los microservicios.

4. Diseñar el esquema para la distribución de los microservicios.

5. Demostrar que el nuevo desarrollo basado en una arquitectura de microservicios es más ágil que el desarrollo anterior.

6. Realizar la implementación de la arquitectura de microservicios en el módulo de caja del sistema institucional.

Hipótesis

En esta investigación se plantea la siguiente hipótesis:

Es posible crear una arquitectura orientada a microservicios que implemente estándares internacionales y que agilice el desarrollo del sistema institucional de la Universidad de Montemorelos.

Limitaciones

Algunas limitaciones de esta investigación fueron las siguientes:

1. Las tecnologías utilizadas para el desarrollo corresponden a versiones community.

2. La planeación del tiempo para cumplir con la elaboración de los módulos en las fechas que indica la dirección de sistemas.

3. El control de la decisión que debe tomar la dirección de sistemas para permitir la implementación de la arquitectura como tal.

4. La habilidad que deben tener las personas en el uso de las herramientas que se utilicen.

Delimitaciones

Se presentan algunas delimitaciones en esta investigación:

1. Se trabajo dentro del contexto de la Universidad de Montemorelos.

2. La arquitectura de microservicios fue pensada de acuerdo con los requisitos específicos del módulo de caja de la Universidad de Montemorelos.

3. Es una arquitectura basada en las tecnologías de Mulesoft y Express.

Definición de términos

De acuerdo con la literatura revisada, se definen los siguientes términos:

Arquitectura orientada a servicios (SOA): es la utilización de servicios para dar soporte a los requisitos del negocio. Permite la creación de sistemas altamente escalables que reflejan el negocio de la organización.

Escalabilidad: es la capacidad de adaptación que tiene un software ante el cambio de circunstancias o tecnologías nuevas (Khazaei et al., 2017).

Backend: es la parte de datos del software que no es accesible para el usuario; encargado de contener toda la lógica, el manejo de datos y el acceso a los distintos tipos de servidores (Aerts, Cailliau, de Groote y Noterman, 2016).

Middleware: es una aplicación encargada de proporcionar infraestructura necesaria para respaldar la construcción de aplicaciones sofisticadas e inteligentes, permitiendo el intercambio de información entre aplicaciones (Del Esposte, Kon, Costa y Lago, 2017).

API: siglas correspondientes a Application Programming Interface.

REST: siglas correspondientes a Representational State Transfer.

HTTP: siglas correspondientes a HyperText Transfer Protocol.

JSON: siglas correspondientes a JavaScript Object Notation.

Ejabberd: es un servidor de mensajería en tiempo real (Balalaie, Heydarnoori y Jamshidi, 2016).

IoT: siglas correspondientes a Internet of Things

Seneca: conjunto de tecnología para escribir microservicios y organizar la lógica de negocios de su aplicación (Lv y Wang, 2016).

BPM: siglas correspondientes a Business Process Management.

Contenedor: es una instancia de una imagen Docker. Un contenedor representa la ejecución de una sola aplicación, proceso o servicio (De la Torre, Wagner y Rousos, 2019).

Docker: es una plataforma de código abierto para el despliegue de aplicaciones dentro de contenedores de software, que envuelve una pieza de software en un completo sistema de archivos que contienen todo lo necesario para ejecutar (Vigneshwaran Sudalaikkan, 2016).

CAPÍTULO II

MARCO TEÓRICO

Introducción

La utilización de un sistema puede ser objetivo crítico dentro de una empresa, ya que puede ser de gran beneficio o perder la ventaja competitiva o, incluso, no ser capaz de sobrevivir si no funciona correctamente (Mens, Magee y Rumpe, 2010).

Se puede describir un sistema como un conjunto que está compuesto por una gran variedad de componentes conectados entre sí, que son de vital importancia para el correcto funcionamiento del sistema (Sangwan, 2015).

En el desarrollo de software, los lenguajes de programación son una parte fundamental, ya que están en un cambio constante con el propósito de mejorar la robustez y mejorar la reutilización de código, buscando un enfoque hacia la distribución, modularización y un acoplamiento más flexible (Dragoni et al., 2017).

Arquitectura de software

Bass, Clements y Kazman (2013) definen la arquitectura de software de un programa o sistema informático como la estructura o estructuras del sistema, que comprende elementos de software, las propiedades visibles externamente de esos elementos y las relaciones entre ellos.

Gorton (2011) comenta que una arquitectura debe diseñarse para cumplir con los requisitos y limitaciones específicos de la aplicación para la que está destinada.

La arquitectura de software debe tener definido cada uno de los componentes que existen dentro de su propia estructura, que permita apreciar la interacción con el resto del sistema, así como tener una definición clara del diseño utilizado para mejorar su funcionamiento y así cumplir con el propósito determinado (López Hinojosa, 2017).

En la actualidad se está rodeado por una gran variedad de sistemas, cada uno diseñado con una arquitectura que va de acuerdo con su funcionamiento. Seleccionar la correcta es lo que permite cumplir con el propósito para el cual ha sido creado el sistema (Sangwan, 2015).

Esto ha ocasionado que surja la necesidad de un mejoramiento día con día, no tanto en el ámbito de seguridad o estándares comunes, sino en la reutilización de componentes y el mejoramiento de la robustez dentro del código (Dragoni et al., 2017).

Uno de los ámbitos importantes es la creación rápida de componentes, que permitan y faciliten la interacción de ellos para el cumplimiento de los requisitos funcionales y no funcionales del sistema. Dividir la arquitectura en componentes y separarlos por niveles permite una mejor toma de decisiones de manera jerárquica, ya que cada nivel está compuesto por normas que rigen su nivel (López Hinojosa, 2017).

Es importante tener en cuenta que la existencia de dependencias entre componentes obliga a que, al realizar un cambio, se generen otras modificaciones en los demás. Al eliminar esas dependencias innecesarias, los cambios se localizan y no se propagan a través de los demás componentes de la arquitectura (Gorton, 2011).

El Lenguaje descriptivo arquitectónico es una herramienta que ayuda para tener una buena documentación y facilitar la comunicación que debe existir entre el arquitecto y las personas interesadas del proyecto (Mens et al., 2010).

Dentro de la arquitectura de software, otro elemento fundamental es que el arquitecto cuente con ciertas habilidades. Gorton (2011) señala las siguientes:

1. La comunicación, donde el arquitecto debe tener la facilidad de comunicarse con los clientes para permitirse entender bien qué es lo que el cliente requiere y espera de un sistema, ya que él será el intermediario entre el cliente y su equipo de desarrollo.

2. El conocimiento de la ingeniería de software, porque el tenerlo le dará al arquitecto una mayor credibilidad de su trabajo.

3. Un conocimiento amplio en el uso de tecnologías, en especial de las que él maneja. Así como la utilización de herramientas que sean necesarias para llevar a cabo este tipo de desarrollo y el estar actualizado dentro de los estándares de tecnologías, diseños y arquitecturas de software.

4. La gestión de riesgos es un factor que el arquitecto debe de tomar en cuenta y hacerlo saber a los clientes o al personal con el que trabaja para estar atentos a cualquiera de esas circunstancias.

Es necesario tener en cuenta que dentro de esta área se presentarán desafíos en cuanto a una arquitectura con alta complejidad, al cambio de tecnologías en el lapso de desarrollo y a la necesidad de funcionar con otro tipo de sistemas (Strimbei, Dospinescu, Strainu y Nistor, 2015).

Arquitectura de microservicios

¿Que son los microservicios? Es el desarrollo de una aplicación compuesta por una gran cantidad de componentes pequeños, independientes, ligeros y con una funcionalidad única (López Hinojosa, 2017). Para la invocación de estos microservicios es necesario una API HTTP REST (MuleSoft, 2019).

En la Figura 1 se puede observar el claro crecimiento en los últimos años que ha tenido el interés de arquitectura basada en microservicios. Todo eso de acuerdo con el aumento de la tecnología, ya que entre más rápido sea, los consumidores quieren sistemas más rápidos (Avidan y Otharsson, 2018).

En el desarrollo actual se ve involucrado el uso de procesos orientados a un desarrollo ágil, haciendo que las aplicaciones deban ser más rápidas (Rahman y Gao, 2015). La utilización de microservicios cada vez va ganando gran popularidad dentro del área de desarrollo de software, realizando un cambio desde la forma en que se percibe y se diseña la aplicación (Dragoni et al., 2017)

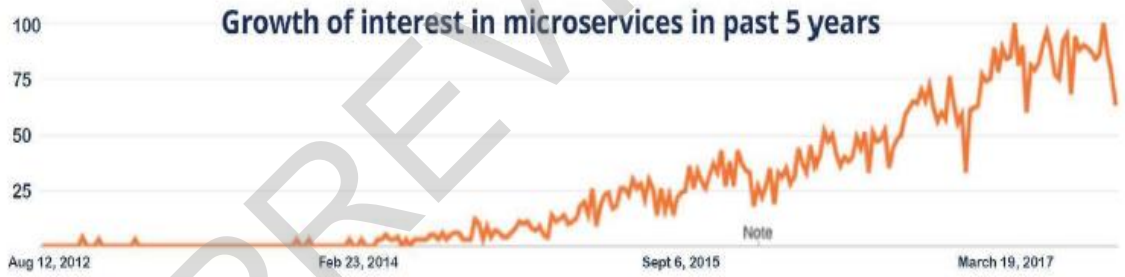

Figura 1. Interest in microservices architecture has grown rapidly since 2012 as a way to solve common problems with application monoliths (Avidan y Otharsson, 2018).

En este tipo de arquitectura es necesario dividir la lógica comercial en pequeños componentes o servicios que contengan estructuras ligeras, permitiendo una integración de los procesos existentes (Oberhauser, 2016). Se busca con esto una creación de microservicios que funcionen de manera independiente a los otros y sean organizados de acuerdo con las funciones que realiza cada uno

(Khazaei et al., 2017). Esta arquitectura es una herramienta que permite simplificar el desarrollo de las aplicaciones, facilitando el desarrollo de una implementación rápida y un mejor mantenimiento de ella. De esta forma, pasa a ser una mejor opción para la gente que desarrolla dichos sistemas (Lv y Wang, 2016). Con la descomposición de una aplicación grande en un conjunto de componentes más pequeños, se permite desarrollar, desplegar, escalar, manejar y visualizar cada uno de ellos de forma independiente (López Hinojosa, 2017). Los microservicios apuntan a la mejora de un óptimo desarrollo, una implementación como tal y a tener la mejor estructura interna dentro del software (Heinrich et al., 2017).

De acuerdo con Mazzara et al. (2018), los principios básicos que rigen el uso de los microservicios son los siguientes:

1. Un contexto delimitado, que consiste en la combinación de funcionalidades que sean iguales.

2. El tamaño, ya que es una característica que distingue a los microservicios donde se pretende crearlos lo más pequeño posible, permitiendo su modificación en cualquier momento y que no afecte la funcionalidad de otros.

3. La independencia, encargada de que el acoplamiento que exista entre cada uno de ellos sea más flexible, donde ellos puedan operar de manera individual.

López Hinojosa (2017) propone cuatro criterios, que son utilizados para realizar la descomposición de microservicios en pequeños componentes; entre ellos se encuentran los siguientes: descomposición por funcionalidades, de acuerdo con su identificación para que encaje y se puedan juntar; descomposición por madurez, encargado de agruparlos dependiendo de sus requerimientos

funcionales y no funcionales; descomposición por el patrón de acceso a datos, basado en el acceso de la recuperación de los datos y descomposición por contexto, realizada a través de servicios que van a la misma entidad.

Nielsen (2015) engloba las características más comunes que deben tener los microservicios, que son las siguientes:

1. Un componente a través de servicios, el cual consiste en la integración de una funcionalidad dentro de los servicios que funciona de forma autónoma.

2. Organización de equipos en torno a las capacidades empresariales, que permite la creación de servicios por equipos diferentes.

3. Productos no proyectos. Este microservicio no se entrega como producto, ya que es utilizado por el equipo mientras sea útil para el desarrollo.

4. Gestión descentralizada de datos, en la que los datos contenidos en cada microservicio son únicos, ya que no permiten el acceso a su información por algún otro medio que no sea el mismo.

5. Capacidad multilenguaje, en que el uso de los microservicios permite la implementación de cualquier tipo de lenguaje de programación, aceptando la integración de distintas tecnologías. En la Figura 2 se observa cómo puede estar compuesto.

6. Despliegues continuos, que pueden ser elaborados y probados por separado.

7. Autocontenido, pueden ser de creación de microservicios autónomos.

8. Comunicación fuera del servicio, que es por medio de protocolos de solicitud y respuestas.

9. Responsabilidad única. En esta parte el microservicio debe contener una única función.