UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE ESTUDIOS SUPERIORES ARAGÓN

“SISTEMA DE GESTIÓN DE BECAS DE LA UNAM

(SGB)”

TRABAJO ESCRITO

EN LA MODALIDAD DE DESARROLLO DE UN CASO PRÁCTICO, QUE PARA OBTENER EL TÍTULO DE:

INGENIERO EN COMPUTACIÓN

P R E S E N T A:

SANDOVAL TEXCAHUAC MIGUEL ANGEL

ASESOR:

M. EN C. FELIPE DE JESÚS GUTIÉRREZ LÓPEZ

MÉXICO 2019

ÍNDICE

INTRODUCCIÓN

La Universidad Nacional Autónoma de México es una de las universidades más importantes a nivel mundial y gran parte de su éxito se debe a la gestión y aprovechamiento de los recursos que le son asignados, para llevar a cabo las funciones sustantivas de docencia, investigación y difusión de la cultura.

La UNAM gestiona cerca de 180,000 becas para que una gran parte de los alumnos de bachillerato, licenciatura y posgrado puedan continuar sus estudios y evitar la deserción. Sin embargo, llevar el control de las convocatorias, selección y control de candidatos, elaboración y gestión de padrones, dispersión de apoyos y creación de reportes eran actividades que se realizaban de forma manual y en archivos de ofimática separados, lo cual además de ser muy complicado aumentaba la posibilidad de cometer errores que se traducían en problemas de efectividad y alcance de los objetivos.

Por lo anterior, fue imperante la creación de un sistema web que apoyara en la gestión de las cuentas bancarias de alumnos y becarios, y la dispersión de los pagos. Por lo que en el presente trabajo se mostrará la manera en la cual se abordó el problema y se diseñó, desarrolló e implementó la solución requerida para tener un buen control en toda la UNAM, haciendo uso de tecnologías de software libre y los servicios de cómputo en la nube que ofrece Google.

CAPÍTULO 1.- ANTECEDENTES

1.1 Cloud Computing

Anteriormente las organizaciones para resolver ciertos problemas o mejorar su funcionamiento mediante soluciones basadas en las Tecnologías de la Información (TI), estaban obligadas a destinar una gran parte de sus recursos humanos, temporales, espaciales, materiales y financieros. Sin embargo y gracias a los avances acelerados en los últimos años en materia de tecnología, ya pueden optar por contratar servicios de infraestructura, plataforma y software de manera rápida y confiable, para resolver sus problemas. A este nuevo paradigma de aprovisionamiento de recursos se le conoce con el nombre de *Cómputo en la Nube* o *Cloud Computing* en inglés.

El Cloud Computing permite ofrecer servicios de computación a través de Internet y como analogía basta con pensar en servicios cotidianos como la energía eléctrica, el agua potable o la telefonía; donde el proveedor tiene la responsabilidad de crear, organizar, mantener y administrar todo lo necesario para que el usuario final reciba lo que necesita y sólo pague por lo que consuma.

Lo anteriormente mencionado es la razón por la cual el concepto de la nube se está extendiendo a una velocidad considerable, ya que permite reducir los tiempos de respuesta y los esfuerzos, permitiendo que el personal que labora en las áreas de TI centre su atención en actividades de mayor impacto dentro de la organización.

Las empresas que brindan servicios en la nube, pueden lograrlo gracias a que cuentan con la infraestructura necesaria en forma de grandes Centros de Datos protegidos y distribuidos de forma estratégica en el mundo, con lo que garantizan el acceso a los diferentes recursos tales como: almacenamiento, memoria, procesamiento, software, base de datos, entre otros. Dichos centros cuentan con las medidas de seguridad, operación y mantenimiento necesarias para garantizar que el usuario pueda disponer de los servicios en todo momento, siempre y cuando cuente acceso a Internet. *(IBM,2018)*

1.1.1 Tipos de servicios en la nube

El cómputo en la nube puede dividirse en tres categorías en función de los servicios que ofrece, las cuales son: Infraestructura como servicio (*Infraestructure as a Service* - IaaS), Plataforma como servicio (*Platform as a Service* - PaaS) y Software como servicio (*Software as a Service* - SaaS).

Cada una de las categorías mencionadas aportan diferentes niveles de control, flexibilidad y administración, con el objetivo de satisfacer las necesidades de los diferentes usuarios.

1.1.1.1 Infraestructura como Servicio (Infraestructure as a Service o IaaS)

En la Infraestructura como Servicio que comúnmente se abrevia *IaaS* por sus siglas en inglés *Infraestructure as a Service*, se brinda a los usuarios acceso a recursos de computación, como servidores, almacenamiento y redes. Este nivel proporciona el mayor control y flexibilidad de la administración de los recursos y guarda el mayor parecido con los recursos de TI existentes dentro de las organizaciones, es decir permite disponer de servidores con diversos sistemas operativos, un centro de datos para almacenamiento y firewall para el control de la red, además de permitir reducir o escalar verticalmente los recursos con rapidez para ajustarlos a la demanda del usuario final.*(Amazon Web Services ,2017)*

El proveedor de servicios en la nube cuenta con la infraestructura mientras que la organización paga por el servicio para usarla, la instala, configura y administra (sistemas operativos y aplicaciones), evitando así la complejidad y el gasto que supone la compra y administración de los servidores físicos y la infraestructura de los centros de datos.

A continuación, se muestra un diagrama IaaS:

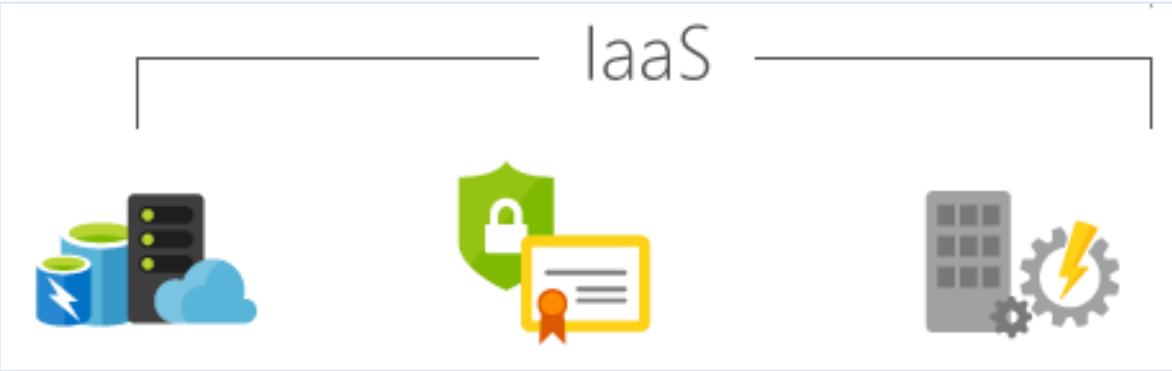

Fig. 1.1 Diagrama que representa los servicios ofrecidos por IaaS.

(Servidores y almacenamiento, redes/firewalls de red y planta física para centro de datos)

(Microsoft Azure, 2018)

Entre los escenarios principales que podemos encontrar con esta infraestructura son los siguientes:

- Desarrollo y pruebas.
- Hospedaje de sitios web.
- Almacenamiento, copias de seguridad y recuperación.
- Aplicaciones web.
- Análisis de Macrodatos.

1.1.1.2 Plataforma como Servicio (Platform as a Service o PaaS)

La Plataforma como Servicio conocida comúnmente como *PaaS* por sus siglas en inglés *Platform as a Service*, es la estructura de la nube en la cual se puede desarrollar, gestionar y entregar aplicaciones. IaaS está incluido dentro de PaaS, es decir, PaaS incluye infraestructura (servidores, almacenamiento y redes), pero también ofrece middleware, herramientas de desarrollo, servicios de inteligencia empresarial (BI), sistemas de administración de base de datos. Esta infraestructura está configurada para sustentar el ciclo de vida completo de las aplicaciones web: compilación, pruebas, implementación, administración y actualización.

Tradicionalmente la construcción de aplicaciones suele ser costosa, compleja y lenta, además de requerir de hardware específico con sistema operativo, base de datos, servidores y otros. Pero con *PaaS* es más eficiente el proceso puesto que provee toda la infraestructura que se necesita para desarrollar y correr aplicaciones sobre Internet, contribuyendo en mejorar la eficacia de la organización, puesto que ya no debe preocuparse por el abastecimiento de recursos, planificación de capacidad, mantenimiento de software, o parches para la ejecución de las aplicaciones. *(Microsoft Azure, 2018)*

A continuación, se muestra un diagrama PaaS:

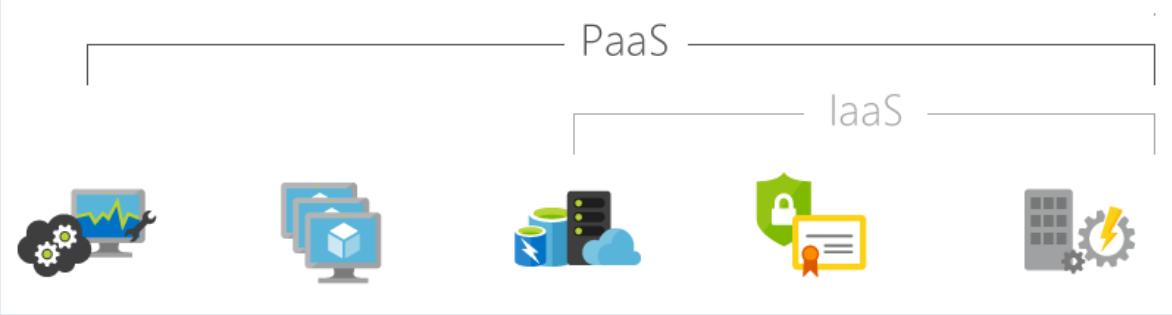

Fig. 1.2. Diagrama que representa los servicios ofrecidos por PaaS.

(Herramientas de desarrollo, administración de base de datos, análisis de negocios, Sistema Operativo, servidores y almacenamiento, seguridad/firewalls de red y planta física para centro de datos.)

(Microsoft Azure, 2018)

La implementación de la plataforma como servicio es utilizada comúnmente por las organizaciones en los siguientes casos:

- Marco de Desarrollo.
- Análisis o inteligencia empresarial.
- Servicios Adicionales.

1.1.1.3 Software como Servicio (Software as a Service o SaaS)

El Software como Servicio o *SaaS* por sus siglas en inglés *Software as a Service* permite la distribución de software a los usuarios a través de la red, abarca tanto *PaaS* como *IaaS*. De tal manera que los usuarios no tienen que preocuparse por la configuración, implementación o mantenimiento de las aplicaciones, ya que todas estas responsabilidades son asumidas por el proveedor. Lo único por lo cual el usuario debe encargarse es de cómo utilizar de forma correcta el software. Esta distribución puede llegar a cualquier organización sin importar la ubicación geográfica y un ejemplo común de la implementación de este servicio son las aplicaciones de email en la web, las cuales permiten enviar y recibir mensajes sin que el usuario tenga que administrar los recursos, servidores o sistemas operativos en los que se ejecuta el programa. El SaaS debe garantizar la disponibilidad y seguridad de las aplicaciones y los datos almacenados. *(Microsoft Azure, 2018)*

El SaaS es el cual es el servicio más común del Cloud Computing y a veces se le conoce con el nombre de Software bajo demanda. A continuación, se muestra un diagrama:

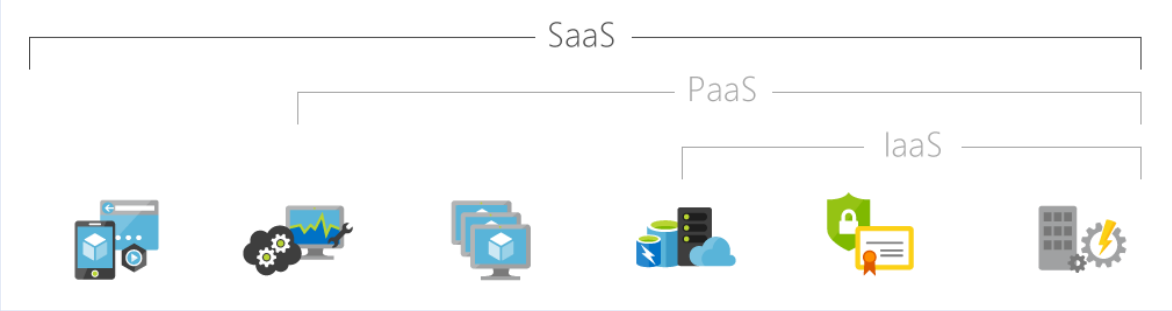

Fig. 1.3. Diagrama que representa los servicios ofrecidos por SaaS.

(Aplicaciones hospedadas, herramientas de desarrollo, administración de base de datos, análisis de negocios, Sistema Operativo, servidores y almacenamiento, seguridad/firewalls de red y planta física para centro de datos.) (Microsoft Azure, 2018)

1.1.1.4 Contendor como Servicio (CaaS)

Existe un nivel intermedio entre el *IaaS* y el *SaaS* denominado Contenedor como Servicio o *CaaS* por sus Siglas en Inglés *Container as a Service,* y es un concepto por el cual se dispone de contenedores[1] bajo demanda. Lo que lo diferencia de la estructura de IaaS y PaaS es la virtualización en contenedores y no la utilización de máquinas virtuales. Es decir, utiliza funciones nativas del núcleo de Linux con la cual se busca aislar cada uno de los procesos que se tienen dentro de un mismo sistema operativo. La tecnología de contenedores encapsula las aplicaciones con su sistema de archivos y los aísla del sistema subyacente,

[1] Los contenedores son una herramienta que permiten empaquetar e implementar aplicaciones para transportarse fácilmente.

permitiendo así gestionar de manera paralela varias aplicaciones con distintos requerimientos en un único sistema operativo. *(Redes Zone, 2017)*

1.1.2 Tipos de nubes

Existen en el mercado distintos tipos de servicios en la nube y se pueden clasificar según la ubicación donde se encuentre la infraestructura, tales como: *(IBM,, 2018)*

- **Nubes Públicas:** son la forma más común de implementar la informática en la nube, puesto que los recursos son propiedad de un proveedor de servicios quien los administra y ofrece a través de Internet. En este tipo de nube se comparte el mismo hardware, almacenamiento y dispositivos de red con otras organizaciones. El acceso a los servicios y la administración de las cuentas se hace a través de un explorador web.

- **Nubes Privadas:** se componen de recursos que son usados por exclusivamente una organización. Pueden ubicarse en el centro de datos de la misma organización o con un proveedor de servicios externo. Sin embargo, se encuentra aislada de las demás, ya que cuenta con una red privada y el hardware y software están dedicados a una sola organización de tal manera que ésta, puede personalizar de forma más sencilla los recursos para cumplir con requisitos específicos de TI.

- **Nubes Híbridas:** combinan la infraestructura local de las nubes privadas con las públicas, de tal manera que las organizaciones pueden beneficiarse de las ventajas de ambas. Los datos pueden moverse entre nubes, obteniendo de esta forma flexibilidad y opciones de implementación.

1.1.3 Ventajas del Cloud Computing

Algunos de los beneficios que podemos obtener al hacer uso del *Cloud Computing* son los siguientes:

- **Flexibilidad:** los usuarios pueden escalar sus servicios para ajustarlos a las necesidades del momento, personalizarlos y acceder a ellos desde cualquier punto que tenga conexión a Internet.

- **Escalabilidad:** la infraestructura de Cloud puede escalarse bajo demanda, es decir que si se necesita de más recursos en automático son asignados, para garantizar de esta manera el servicio.

- **Opciones de almacenamiento:** se puede elegir entre una solución de tipo pública, privada o hibrida, en función de las necesidades de seguridad de la información.

- **Opciones de Control:** la organización puede determinar el nivel de control con la elección de algunas de las estructuras de servicio, las cuales incluyen Software como Servicio, Plataforma como Servicio e Infraestructura como Servicio.
- **Selección de herramientas:** se pueden elegir diferentes herramientas y características predefinidas, para la creación de una solución que responda a las necesidades específicas de la organización.

- **Características de seguridad:** se cuenta con cifrado y claves que ayudan proteger los datos.

- **Eficiencia:** al no preocuparse por los costos de infraestructura o mantenimiento, las organizaciones pueden poner en producción en tiempos menores sus aplicaciones.

- **Accesibilidad:** las aplicaciones que se encuentran en Cloud así como los datos, están disponibles desde cualquier dispositivo que tenga una conexión a Internet.

- **Difusión:** permite el lanzamiento de aplicaciones desarrolladas al mercado de una forma más rápida.

- **Seguridad de datos:** los errores de hardware no suponen gran problema en la pérdida de datos, debido a la redundancia y copias de seguridad en red.

- **Ahorro en el equipo:** al utilizar equipos remotos, las organizaciones ahorran costos en la adquisición de servidores y otros equipos de infraestructura.

- **Estructura de pago:** es de tipo *utility,* esto quiere decir que los usuarios sólo pagan por los recursos que utilizan.

- **Valor estratégico:** los servicios Cloud brindan una ventaja competitiva, por el uso de la tecnología más innovadora.

- **Trabajo optimizado:** ya que los proveedores se encargan de gestionar la infraestructura, permite a las organizaciones centrarse en el desarrollo de aplicaciones y otras actividades de mayor prioridad.

- **Actualizaciones regulares:** las soluciones son actualizadas con regularidad, brindando así la tecnología más moderna.

- **Colaboración:** la colaboración se puede dar desde diversos lugares, debido al acceso mundial.

- **Ventaja competitiva:** al destinar más recurso a TI en vez de a la infraestructura, se tiene un mayor desarrollo y crecimiento organizacional.

1.2 Google Cloud Platform

La plataforma de servicios en la nube de Google en inglés se llama *Google Cloud Platform* o *GCP*, así como también existen los servicios de Amazon denominados *Amazon Web Services* o AWS, los de Microsoft (Azure), IBM (IBM Cloud), entre otros. Cada una ofrece características según el proveedor con precios variables según las necesidades a cubrir.

Google Cloud Platform cuenta con un conjunto de equipos activos físicos, como computadoras y unidades de disco duro, recursos virtuales como máquinas virtuales (MV) que se encuentran en los centros de datos de Google distribuidos en ciertas regiones del mundo tales como: Estados Unidos, Europa occidental y Asia Oriental. Cada una de estas regiones se divide en zonas, las cuales a su vez son identificadas con un nombre que combina un identificador de letra con el nombre de la región, es decir para nombrar una zona de Asia Oriental sería *Asia-east1-a.* Con esta distribución se busca proporcionar varios beneficios, entre ellos la redundancia en caso de falla y latencia reducida al ubicar los recursos más cerca de los usuarios. *(Google Developers,2018)*

Los servicios están compuestos por software y hardware y proporcionan acceso a los recursos subyacentes. Cuando se desarrolla una aplicación o un sistema en GCP, se mezclan y combinan estos, para crear la infraestructura que se necesita, después se agrega el código para los escenarios requeridos.

Todos los recursos de GCP deben estar asignados a un proyecto con un nombre, identificador único a nivel mundial y un número asignado, manteniendo una cierta configuración, permisos y metadatos. Tal y como se puede observar en la siguiente figura:

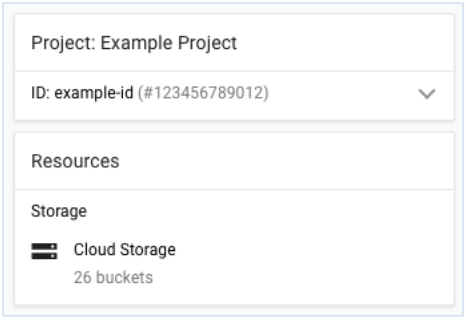

Fig. 1.4 Ejemplo de estructura de un proyecto (Google Developers,2018)

GCP ofrece tres formas de interacción con el usuario que son: consola de la plataforma, interfaz en línea de comandos y bibliotecas de clientes.

- **Consola de la plataforma:** es una interfaz gráfica basada en web y representa forma más básica de interacción entre el usuario y los servicios de Google. En la imagen continua se puede ver una captura de pantalla:

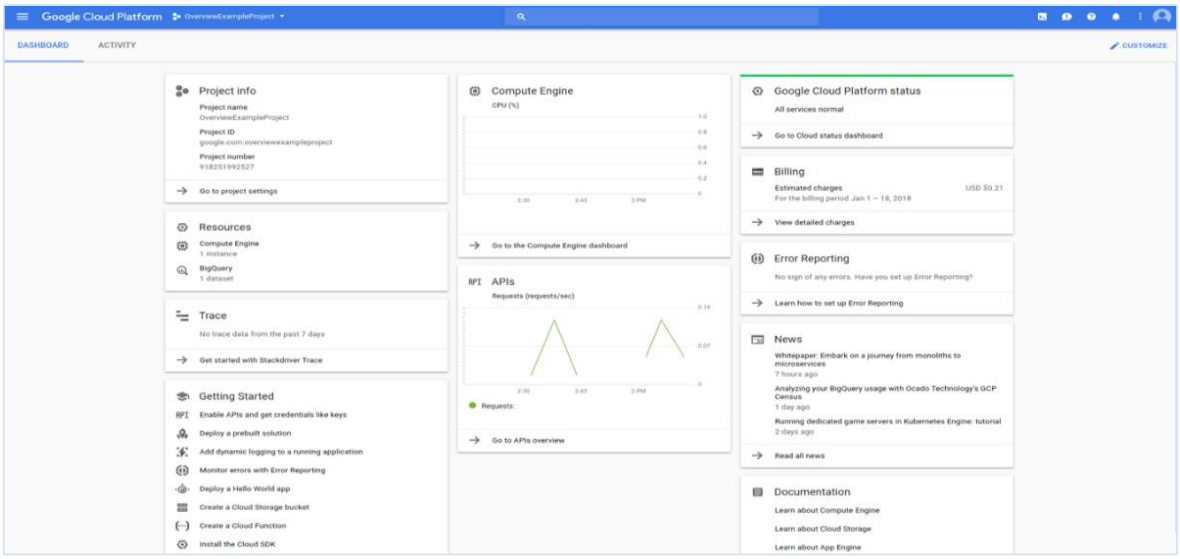

Fig. 1.5. Interfaz web de la plataforma de Google (Google Developers,2018)

- **Línea de Comandos:** Google Cloud SDK[2], proporciona gcloud una herramienta de línea de comandos que puede ser usada para administrar tanto el flujo de trabajo de desarrollo como los recursos de GCP. Adicionalmente se ofrece Cloud Shell, un entorno de shell basado en navegador. A continuación, una imagen de muestra:

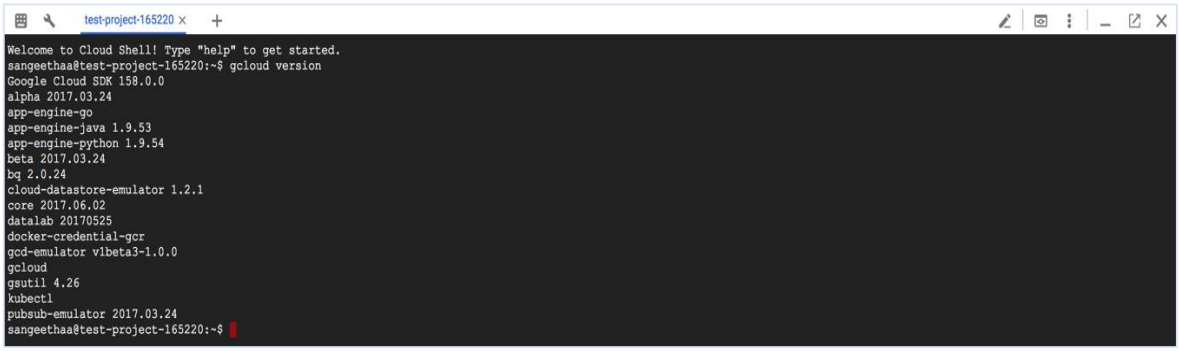

Fig. 1.6. Cloud Shell de Google (Google Developers,2018)

- **Bibliotecas de clientes:** el SDK incluye bibliotecas de clientes que permiten crear y administrar recursos de una manera sencilla mediante las API[3], las cuales cumplen con dos propósitos principales:

[2] Es un conjunto de herramientas para *Cloud Platform*, donde se incluyen *gcloud*, *gsutill* y bq. Estas herramientas ofrecen acceso desde la línea de comandos a *Google Compute Engine, Google Cloud Storage, Google BigQuery* (productos y servicios de GCP).

[3] Una API es un conjunto de funciones y procedimientos que pueden ser aprovechados por otras aplicaciones. Las siglas provienen del inglés Application Programming Interface o Interfaz de Programación de Aplicaciones en español.

- Las API de aplicaciones brindan accesos a los servicios.
- Las API de administración ofrecen funcionalidad para la administración de recursos.

Las API de Google también puede ser utilizadas para acceder a productos tales como: Google Maps, Google Drive y YouTube.

1.2.1 Servicios de GCP

Los servicios ofrecidos por la nube de Google pueden ser clasificados de la siguiente manera:

- Computación y alojamiento.
- Almacenamiento.
- Redes.
- Big Data.
- Aprendizaje Automático.

1.2.1.1 Servicios de Computación y alojamiento

Dentro de estos servicios podemos elegir entre trabajar en un entorno sin servidor, usar una plataforma de aplicaciones administradas, aprovechar las tecnologías de los contenedores o crear infraestructuras propias basada en la nube mediante Máquinas Virtuales. Tal y como se muestra en el esquema siguiente:

Fig. 1.7. A la izquierda la responsabilidad es mayor para Google mientras hacia el lado opuesto es del usuario (Google ,2018)

1.2.1.2 Servicios de Almacenamiento

Cualquiera que sea la aplicación del cliente, necesita almacenar datos, por lo que GCP proporciona diversos servicios de almacenamiento, que incluyen lo siguiente:

- **Cloud SQL:** es un manejador de bases datos en MySQL o PostgreSQL.
- **Cloud Spanner:** es un servicio de datos relacional de misión crítica.
- **Cloud Datastore y Cloud Bigtable:** representan dos opciones para almacenamiento de datos NoSQL.

- **Cloud Storage:** es almacenamiento de datos escalable, consistente y de gran capacidad.
- **Discos persistentes:** tal y como lo señala el sitio web, es un servicio de almacenamiento en bloques fiable y de alto rendimiento para instancias de máquinas virtuales.

1.2.1.3 Servicios de Redes

Estos servicios ayudan a mantener equilibrada la carga de tráfico generada en los recursos, además de permitir crear registros DNS y conectar una red existente con la red de Google. Sus funciones principales son:

- **Redes, firewalls y rutas**: permite controlar el tráfico que entran en las instancias de una red. El firewall tiene un conjunto de reglas predeterminadas que pueden personalizarse
- **Balanceo de carga:** permite distribuir el tráfico entre las instancias del servidor en función de los datos del protocolo IP entrante y la dirección el puerto.
- **DNS en la nube:** permite publicar y mantener registros del Sistema de Nombre de Dominio.
- **Cloud VPN:** ofrece la capacidad de conectar la red existente a través de una conexión IPSEC[4].

1.2.1.4 Servicios Big Data

Este tipo de servicios permiten procesar y consultar grandes datos en la nube para dar respuestas a preguntas complicadas y sus principales componentes son:

- **BigQuery:** ofrece la capacidad de analizar datos, de los cuales se pueden crear esquemas personalizados para organizarlos en conjuntos de datos y tablas. Permite ejecutar comandos similares a SQL y cuenta a su vez con una interfaz web.
- **Cloud Dataflow:** es un servicio administrado en conjunto con el SDK para procesar altos volúmenes de datos por lotes y transmisión.
- **Cloud Pub/Sub:** es un servicio de mensajería asíncrona para el envío de mensajes estructurados en formato JSON.

1.2.1.5 Servicios de Aprendizaje Automático

Cloud AI tiene una variedad de servicios de aprendizaje automático o Machine Learning (ML) por sus siglas en inglés, en los que se pueden utilizar API's con modelos previamente entrenados para aplicaciones específicas o bien permite construir y entrenar modelos propios.

[4] Conjunto de protocolos cuya función es asegurar las comunicaciones sobre el Protocolo de Internet (IP).

1.2.2 Google App Engine

Google App Engine es el servicio PaaS de GCP, en el cual Google tiene la administración de la mayor parte de los recursos. Supongamos que una aplicación que funciona en la nube exige más capacidades de cómputo debido a que el tráfico al sitio ha aumentado, Google se encargará de escalar el sistema de manera automática para proporcionar los recursos necesarios, al igual que si se requiere de una actualización de seguridad.

Los lenguajes de programación admitidos hasta el momento por el entorno estándar son: Phyton, Java, PHP y Go. Con la utilización del SDK de App Engine, se puede desarrollar y probar en una máquina local un entorno que simule App Engine.

Google también cuenta con tecnologías de almacenamiento como Google Cloud SQL, el cual es compatible con MySQL y PostgreSQL. Además de proporcionar espacio para archivos de gran tamaño con Google Storage.

Cloud Security Scanner es la herramienta que analiza y detecta vulnerabilidades de seguridad en las aplicaciones web y en contenido mixto.

Google permite la configuración con eclipse a través del plugin de Google Cloud Platform, para hacer el *Deployment* o despliegue de nuestra aplicación en App Engine y poder visualizarlo en la web. *(Google, 2018)*

Con la creación de una cuenta en la nube, Google te regala $300 pesos para probar sus diferentes servicios o establecer un entorno de desarrollo antes de lanzarlo a producción. Una vez concluido el crédito, será necesario pagar para usar la plataforma.

1.3 Web Services

La necesidad de compartir datos e información entre computadoras con distintos sistemas operativos, husos horarios, ubicaciones geográficas y capacidades de procesamiento, condujo a la creación de tecnologías y estándares universales como los *Web Services* o *Servicios Web* en español.

La importancia de los Web Services radica en que permiten a las aplicaciones intercambiar información sin importar el lenguaje de programación o el sistema operativo. IBM los define como: “Los servicios web son aplicaciones autónomas modulares que se pueden describir, publicar, localizar e invocar a través de una red.” *(IBM, 2013)*

1.3.1Tipos de Web Services

Dos de las tecnologías más utilizadas hoy en día en el mundo de la informática para compartir información mediante Web Services son: SOAP (Simple *Object Access Protocol o Protocolo Simple de Acceso a Objetos*) y REST (*Representational State Transfer o Transferencia de Estado Representacional*).

1.3.1.1 SOAP

En este protocolo se utilizan archivos XML (eXtensible Markup Language o Lenguaje de Marcado Extensible) para encapsular los datos de manera análoga a un sobre de correo (*envelope*); se define una cabecera (*head*) y un cuerpo de mensaje (*body*) para así poder enviarlos mediante algún protocolo de transporte como http, smtp o ftp. Maneja el puerto 80 el cual en muy pocas ocasiones está bloqueado por los firewalls, brindando una fiabilidad muy alta.

Los documentos en SOAP deben definirse siguiendo el formato WSDL (*Web Services Description Language*), el cual describe las funciones y servicios disponibles para que otras aplicaciones hagan uso de ellas.

SOAP se enfoca en los RPC (*Remote Procedure Call* o Llamada a Método Remoto), lo que significa que aquellas aplicaciones que quieran interactuar con este servicio sólo podrán hacerlo a través de los mismos servicios. *(Benjamin,2007)*

Algunas ventajas y desventajas de SOAP son:

Ventajas	Desventajas
• Independencia de la plataforma donde se ejecuta. • Simplifica la comunicación entre proxies y firewalls. • Soporte a diferentes protocolos de transporte como http, ftp y smtp.	• Es más lento que otros middlewares, ya que hace uso de un XML muy cargado de definiciones. • Sólo permite una conexión a la vez, cuando se está haciendo uso del protocolo http. • La latencia suele ser alta cuando se hace uso del protocolo http, ya que los firewalls por lo general analizan las conexiones usando este protocolo. • El soporte es distinto entre diferentes lenguajes de programación.

Tabla 1.1. Ventajas y desventajas de SOAP (Sandoval M,2018)

A continuación, se muestra un diagrama en el que se comparte información mediante el protocolo SOAP:

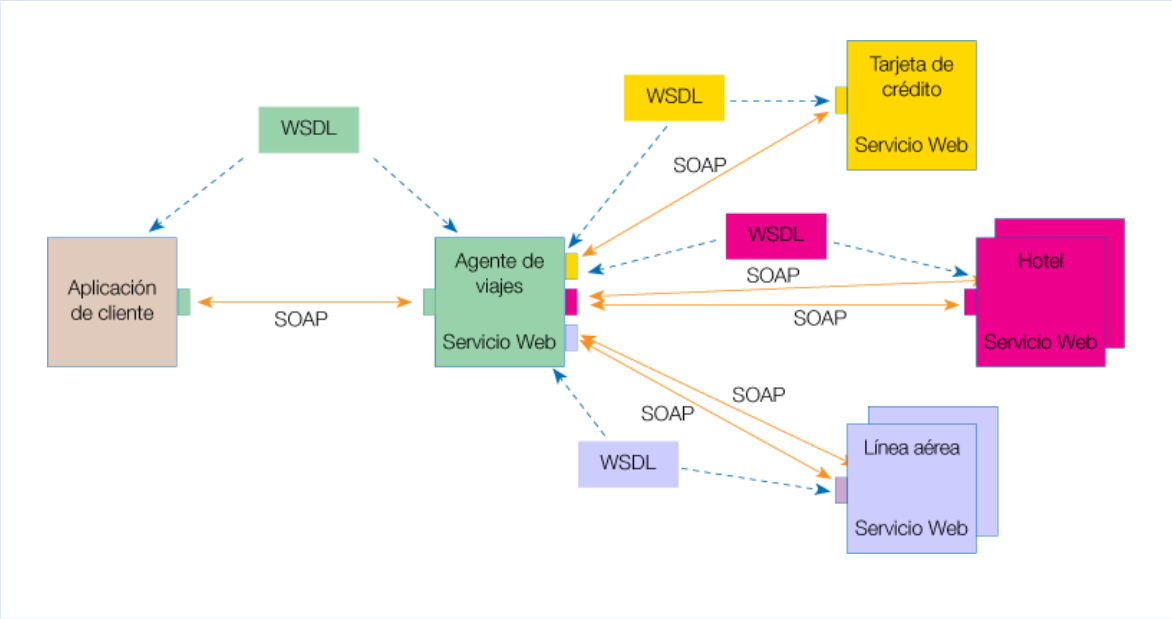

Fig. 1.8. Diagrama que muestra la dinámica de trabajo de una petición SOAP en donde se puede ver la codificación WSDL (http://www.w3c.es/Divulgacion/GuiasBreves/ServiciosWeb,2003)

1.3.1.2 REST

Este servicio es el más utilizado a nivel mundial y no maneja estados (*stateless*), lo que significa que el servidor nunca guardará los datos de la petición, liberando memoria al no ser necesario mantener las sesiones de los solicitantes de recursos.

REST a diferencia de SOAP no interactúa a través de servicios si no de recursos y lo hace mediante rutas URI y los verbos http siguientes:

- *POST* para crear.
- *GET* para leer y consultar.
- *PUT* para editar.
- *DELETE* para eliminar.

REST permite manipular datos tanto en formato JSON (JavaScript Object Notation o Notación de Objetos de JavaScript) como en XML y tiene la capacidad de atender grandes volúmenes de peticiones.

1.4 Java

Es un lenguaje de programación orientado a objetos de Oracle Corporation pero que originalmente pertenecía a Sun Microsystems. Fue creado con la idea de ser multiplataforma, es decir que un mismo código tuviera la capacidad de ejecutarse en diferentes computadoras, sistemas operativos y dispositivos; de ahí la frase "*Write once, run anywhere*", es decir "*escríbelo una vez y córrelo donde sea*". *(Ictea,2018)*

Para que las aplicaciones desarrolladas en Java puedan ejecutarse en diferentes entornos requieren de una *Máquina Virtual de Java* o *JVM* por sus siglas del inglés *Java Virtual Machine*. Esta máquina funciona a manera de traductor de instrucciones específicas del código Java para cada sistema y dispositivo.

El entorno de Java dispone de un Kit de Desarrollo o JDK (Java Development Kit) y un Ambiente de Ejecución o JRE (Java Runtime Environment) para que los usuarios puedan desarrollar y ejecutar sus aplicaciones en diferentes plataformas.

1.5 Patrón MVC

El patrón MVC (Model View Controller o Modelo Vista Controlador) es un esquema de desarrollo que separa la aplicación en tres capas con roles específicos, así pues el "Modelo" se encarga de los datos que alimentan al software, el "Controlador" ejecuta toda la lógica de negocio y sirve de intermediario entre el "Modelo" y la "Vista", y esta última se encarga de interactuar con el usuario.

A continuación se incluye un diagrama con el patrón MVC:

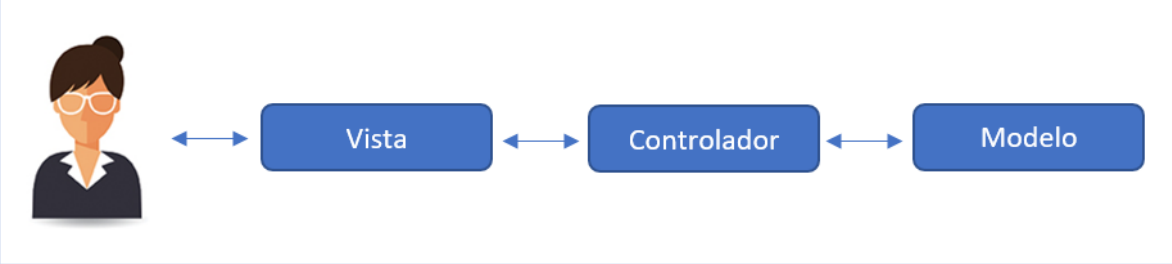

Fig. 1.9. El Modelo Vista-Modelo-Controlador separa las aplicaciones en capas (Sandoval M,2018)

CAPÍTULO 2.- PLANTEAMIENTO DE LA PROBLEMÁTICA Y SU SOLUCIÓN

2.1 Análisis del problema

La Dirección General de Orientación y Atención Educativa de la UNAM (DGOAE) es la entidad responsable de otorgar y gestionar alrededor de 180,000 becas a nivel bachillerato, licenciatura y posgrado para alumnos y egresados. Actualmente, la publicación de convocatorias, selección y control de candidatos, elaboración y gestión de padrones, dispersión de apoyos y creación de reportes, son actividades que se realizan de forma manual en papeles y tablas de MS Excel. La falta de automatización consume muchos recursos temporales, humanos y materiales, además de que también es importante tomar en cuenta los errores que puedan cometerse y la necesidad de capacitar al nuevo personal propio de la rotación.

Algunos de los errores que se comenten en el proceso de control de becas son:

- Falta de datos.
- Datos erróneos.
- Errores por homonimias.
- Falta de verificación de inscripción.
- Errores de distribución de becas:
    - Diferencias en la información del becario vs la cuenta bancaria.
    - Becarios sin asignación de recurso.
    - Asignación de recursos rechazados.

2.2 Descripción del proceso

El proceso de selección de alumnos para beneficiaros de beca sigue el siguiente flujo:

a) La DGOAE envía a la SEP un catálogo de los alumnos inscritos en la UNAM en formato CSV.
b) La SEP brinda acceso a los alumnos recibidos al sistema SUBES (Registro al Sistema Único de Beneficiarios de Educación Superior).
c) La SEP notifica a la DGOAE que el sistema se encuentra abierto.
d) La DGOAE verifica que en efecto el sistema se encuentre disponible y libera la publicación de la convocatoria de Becas.
e) Los alumnos ingresan al sistema de SUBES con las credenciales adecuadas y se aseguran que su información sea correcta:
    a. Si su información posee detalles, recurren con el jefe del departamento de becas para informar de su situación y realizar las actividades necesarias para solucionar el problema.
    b. Si es correcta, el alumno selecciona la beca de acuerdo a la convocatoria, con la cual obtiene un número de folio y su registro en el Sistema SUBES queda concluido.

f) El alumno ingresa al sitio *www.dgoae.unam.mx* para registrar el número de folio obtenido en SUBES, junto con la información personal necesaria indicada en el mismo sistema.
g) Pasado el tiempo reglamentario de la convocatoria, el sistema se cierra.
h) La DGOAE pide a la SEP el archivo con la información de registros realizados en el sistema SUBES. Y es el jefe del Departamento de Enlace con la comunidad quien recibe la solicitud y proporciona la información a la UNAM en un archivo plano.
i) La DGOAE concilia la información enviada por la SEP con la propia, seleccionando a los alumnos que concluyeron ambos registros, cumplen con los requisitos de la convocatoria y que pueden ser beneficiarios de beca.
j) Posteriormente, la DGOAE envía una propuesta de oficio para la DGAE (Dirección General de Asuntos Escolares) y la Dirección General de Planeación, y en caso de ser aceptado por ambos, estos preparan un archivo con la lista de los alumnos, su estatus actual y la situación económica, el cual es depositado en un servidor sftp para su descarga; acompañado de un oficio de disponibilidad.
k) La DGOAE realiza una conciliación de la información[5] y considerando criterios de priorización selecciona a los beneficiarios[6].
l) Una vez definidos los alumnos beneficiarios, se define un padrón, el cual es enviado a través de correo electrónico a la SEP.
    a. La SEP recibe el correo electrónico con el padrón, lo valida en el Sistema SUBES y en caso de ser correcto publica los resultados del folio en su portal con los alumnos seleccionados.
m) Para que los alumnos reciban su pago, la DGOAE genera en Excel los archivos en el formato que requiere INBURSA y BANAMEX, y los envía a Fundación UNAM quien a su vez los hace llegar a la banca para que procese los pagos.
n) Finalmente, y una vez que la banca realiza sus procesos, envía otro archivo describiendo los estados de los pagos.

2.3 Objetivos

El objetivo que se desea alcanzar la DGOAE es: Desarrollar un sistema institucional de becas para apoyar a la gestión de cuentas bancarias, alumnos y becarios, dispersión de pagos.

El sistema deberá cubrir con los requerimientos siguientes:

a) Back End:
- Servicios en la nube en la modalidad de Software como Servicio.
- Escalable bajo demanda.
- Seguridad y disponibilidad.

[5] Los archivos de intercambio entre dependencias tienen formato .csv o .xls y son manejándolos en Excel.

[6] La aplicación de los criterios de selección y priorización se realiza de forma manual, mediante el ordenamiento, filtrado y marcado de los registros de los beneficiarios.

b) Front End:

- Carga de los diferentes catálogos que se tienen.
  - Matrícula Escolar de la UNAM.
  - Carreras.
  - Planteles.
  - Carreras por plantel.
  - Planes de estudios.
  - Créditos acumulados por año por plan de Estudios.
  - Información bancaria.
  - Relación de Programa Beca – Alumno.
  - Registro realizado en SUBES.
- Carga de instrucciones de pago de los bancos INBURSA y BANAMEX.
- Asignación de Roles.
- Permisos de navegación según el rol.

2.4 Solución propuesta

Para alcanzar el objetivo planteado y satisfacer los requerimientos, se desarrollará un Sistema de Información que se soportará bajo Google Cloud, el cual ofrece la estructura de Software como Servicio bajo el nombre de App Engine; no requiere de administradores, cuenta con altos niveles de seguridad, se apoya en servicios disponibles para la comunidad, desarrollo rápido y crecimiento bajo demanda.

El Back End se desarrollará en JAVA con el uso del IDE Eclipse, para hacer el deployment al App Engine a través de un plugin y el Front End se trabajará con HTML, CSS, Java Script.

Este sistema ayudará a gestionar la carga de los archivos que son utilizados y las instrucciones de pago de cada beneficiario. Se reducirán los tiempos de carga y consulta, mejorando la efectividad de la DGOAE al tener los datos de cada alumno relacionado entre sí, como lo pueden ser el Plantel en el que se encuentra inscrito, la carrera a la que pertenece, su información bancaria, los pagos que se le han realizado y el estatus de los mismos.

Dentro de las consultas que se podrán realizar en el sistema se encuentran las siguientes:

- Escuelas y Facultadas.
- Carreras.
- Programas Académicos.
- Planes de Estudios.
- Alumnos.
- Consulta SUBES.
- Búsquedas filtradas (Nombre, Número de Cuenta UNAM, CURP).

2.5 Metodología ágil SCRUM

Para la elaboración y desarrollo del sistema se trabajará bajo la metodología ágil Scrum, garantizando de esta manera un trabajo colaborativo y en equipo. En la cual se definirán las entregas parciales del producto final.

Scrum fue elegido de entre las metodologías ágiles debido a que fue desarrollada especialmente para proyectos con entornos complejos, requisitos cambiantes y que no están bien definidos y para la obtención de resultados prontos.

En la imagen siguiente se puede ver un modelo de desarrollo con Scrum:

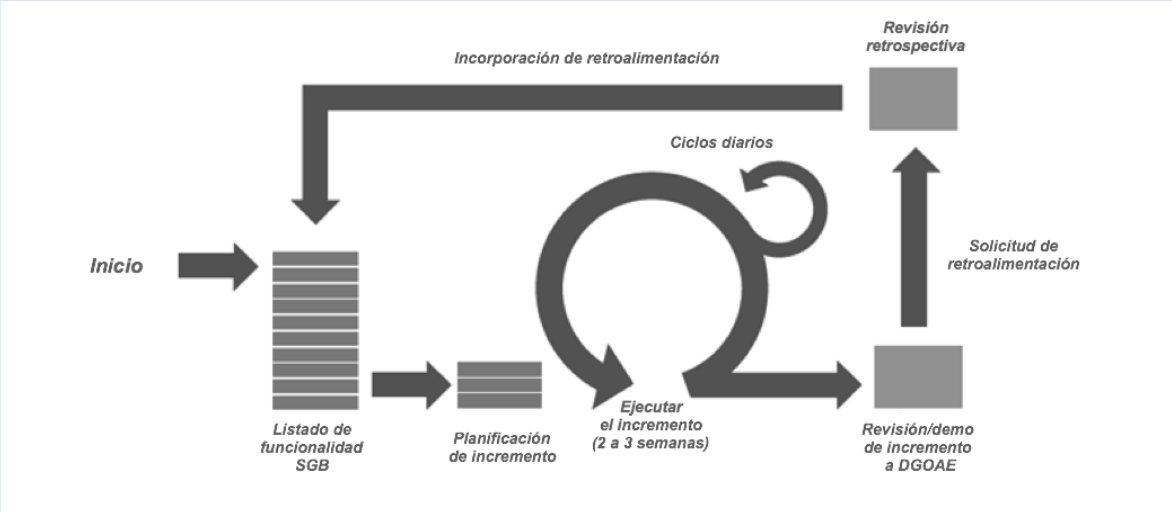

Fig. 2.1 Modelo de desarrollo con Metodología Scrum

(https://blog.ida.cl/estrategia-digital/metodologia-scrum-en-proyectos-digitales/, 2018)

A través de la metodología se define lo siguiente:

- Reuniones quincenales con la Dirección de Becas.
- Reuniones diarias entre el equipo de Desarrollo para revisión de Avances.
- Roles:
  - Dueño del Sistema.
  - Administradores del Proyecto.
  - Equipo de Desarrollo.

CAPÍTULO 3.- DISEÑO Y DESARROLLO DEL SISTEMA

El Lenguaje de Modelado Universal o UML (Unified Modeling Language) se utilizó para diseñar el sistema, de ahí que en los apartados siguientes se incluirá la Arquitectura del sistema, Diagramas de casos de uso general, secuencia, estado, clases y base de datos.

3.1 Arquitectura del sistema

3.1.1 Arquitectura física

Por la forma en que esta desarrollado el proyecto, no se cuenta con una arquitectura física propia, debido a que se basa en un servicio en la nube de Google, por lo cual sólo se necesita un equipo con acceso a Internet para acceder al sistema.

En la imagen siguiente se muestre un esquema básico de la arquitectura física:

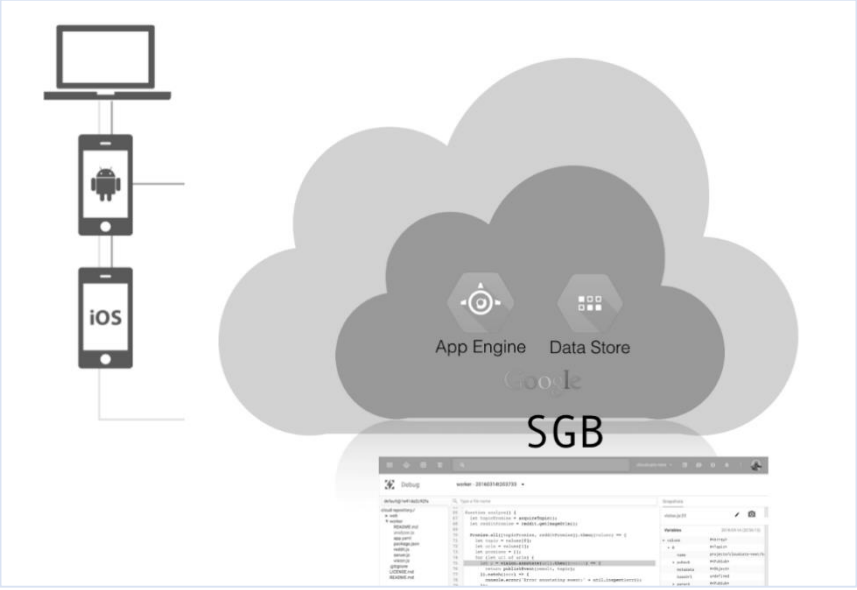

Fig. 3.1. Diagrama de Arquitectura Física del Proyecto. (Sandoval M,2018)

3.1.2 Arquitectura lógica

La arquitectura lógica se dividió en cuatro capas, las cuales se dedicarán a las siguientes funciones: Persistencia, Servicios, Comunicación y Aplicación. A continuación, una breve explicación de cada una de ellas:

- **Persistencia:** encontraremos todo lo referente a la implementación de la lógica para el almacenamiento de la información, en este caso la comunicación con SQL y la conexión con la base de datos.

- **Servicios:** incluye aquellos servicios que cumplen con el esquema de negocios de la dependencia y a su vez mantiene comunicación con la capa inferior para ejecutar las reglas de negocio.
- **Comunicación**: en esta capa se declara la manera en que se tiene acceso a los servicios ofrecidos por la aplicación, así como la manera en que se da la comunicación entre la aplicación y los servicios. Para este proyecto la comunicación se estableció través de archivos con estructura JSON.
- **Aplicación:** es la parte visual del proyecto donde el usuario tiene interacción con el sistema.

A continuación, un esquema visual de las capas del proyecto, que como se puede observar cumplen con el modelo MVC:

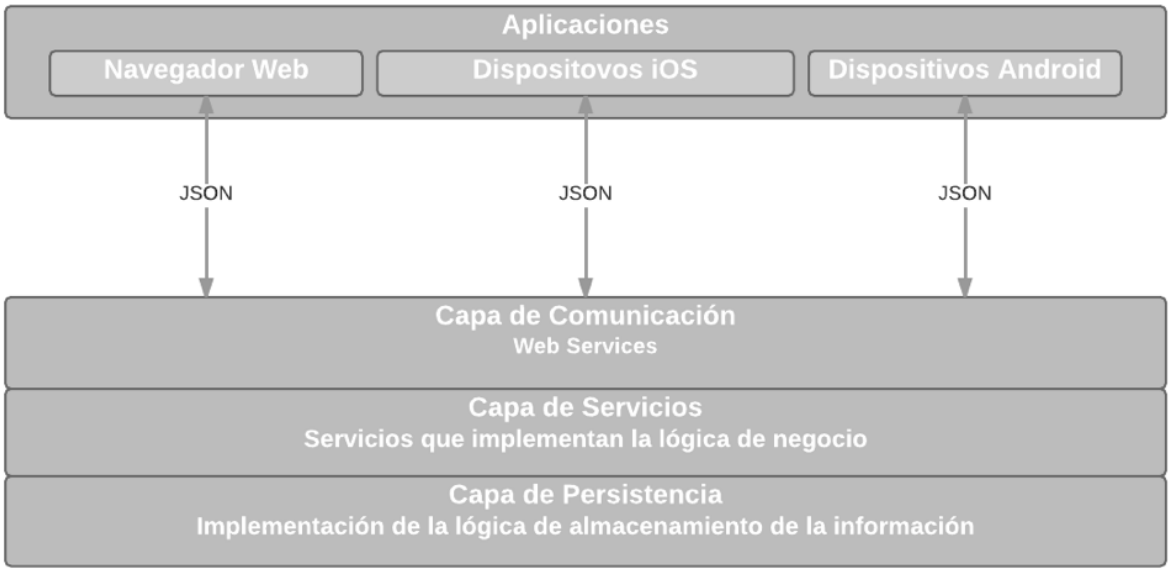

Fig. 3.2. Representación Arquitectura Lógica (Sandoval M,2018)

3.2 Diagrama de casos de uso

El siguiente diagrama muestra los casos de uso generales del sistema:

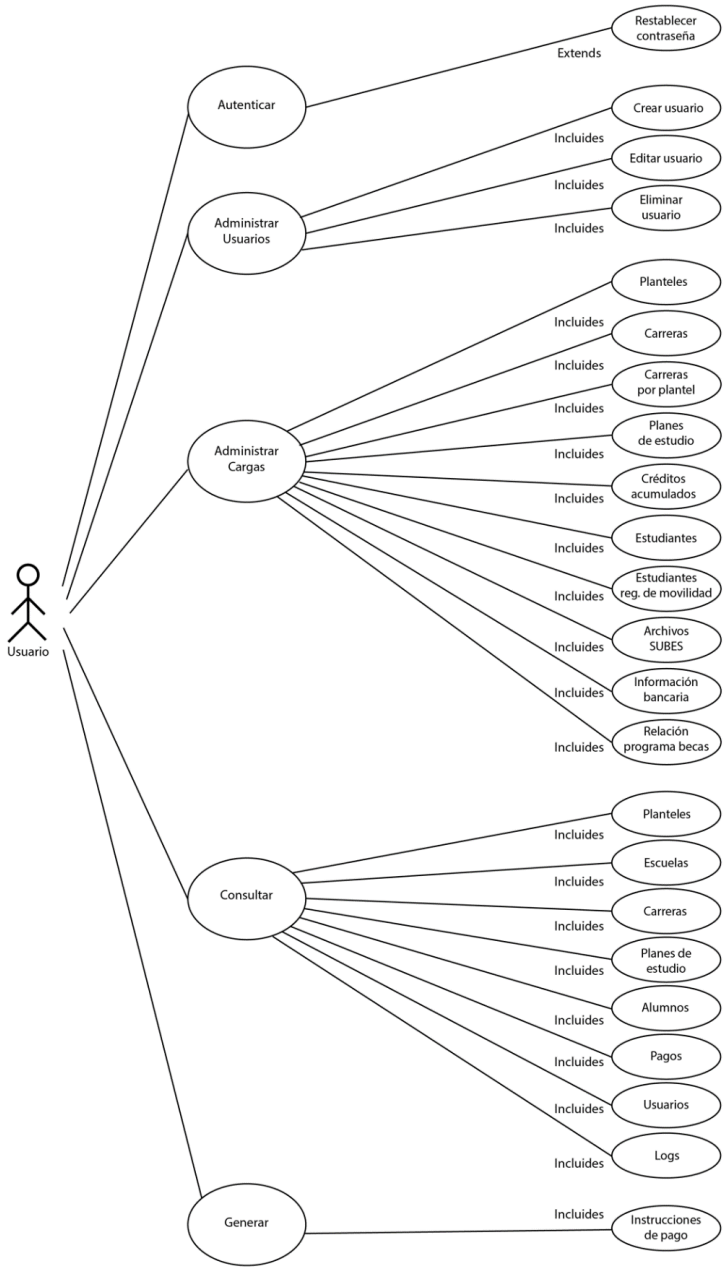

Fig. 3.3 Diagrama General de Casos de Uso (Sandoval M,2018)

3.3 Diagramas de secuencia

En este apartado se muestran los diagramas de secuencia para el acceso, instrucciones de pago, manejo de usuarios, consultas, logs y cargas.

A continuación, el diagrama de secuencia para el acceso:

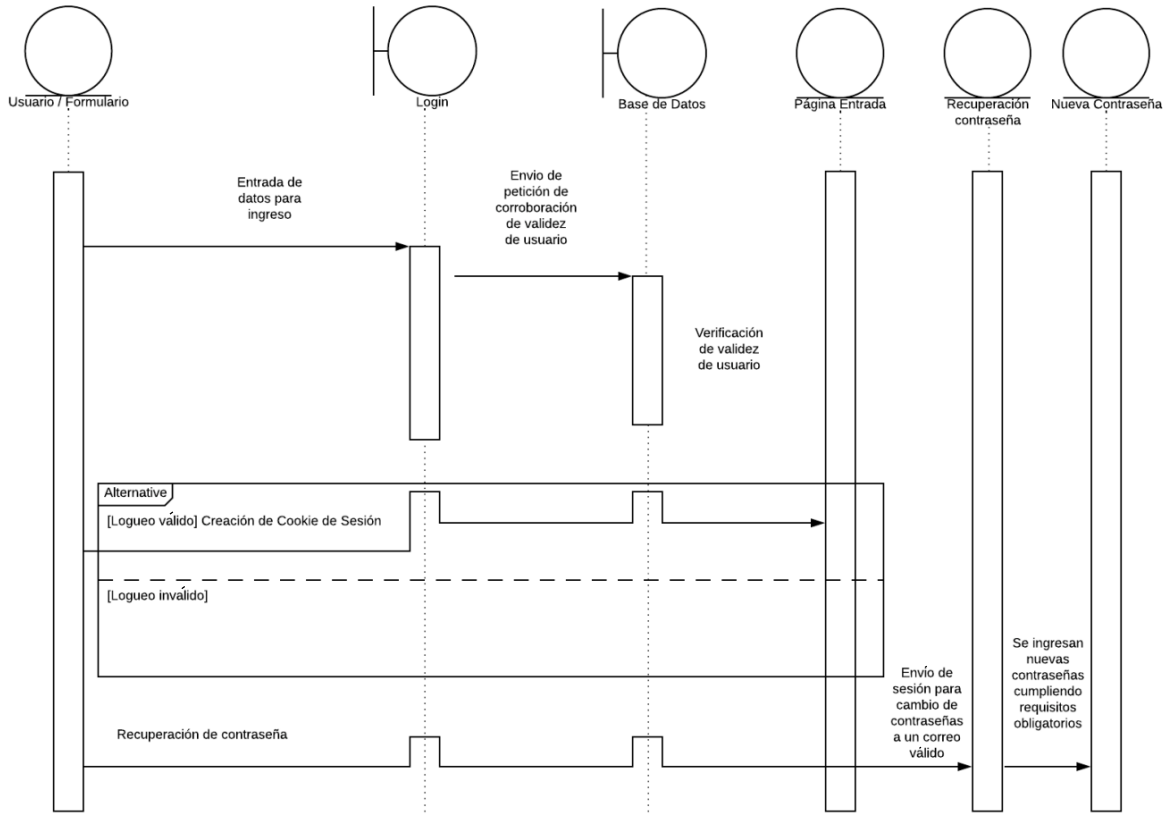

Fig. 3.4. Diagrama de Secuencias describiendo el acceso al sistema (Sandoval M,2018)

Ahora se verán los diagramas de secuencia para las instrucciones de pago:

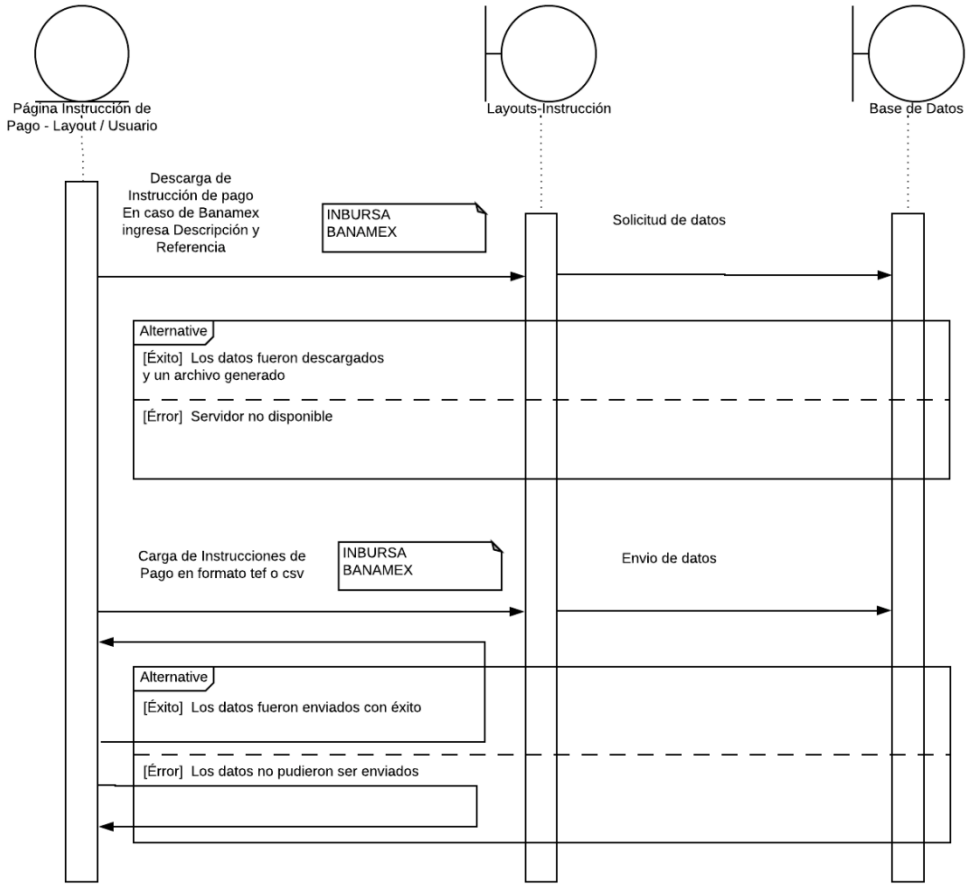

Fig. 3.5. Diagrama de secuencias describiendo las instrucciones de Pago (Sandoval M,2018)

En el diagrama siguiente se muestra la secuencia para los usuarios:

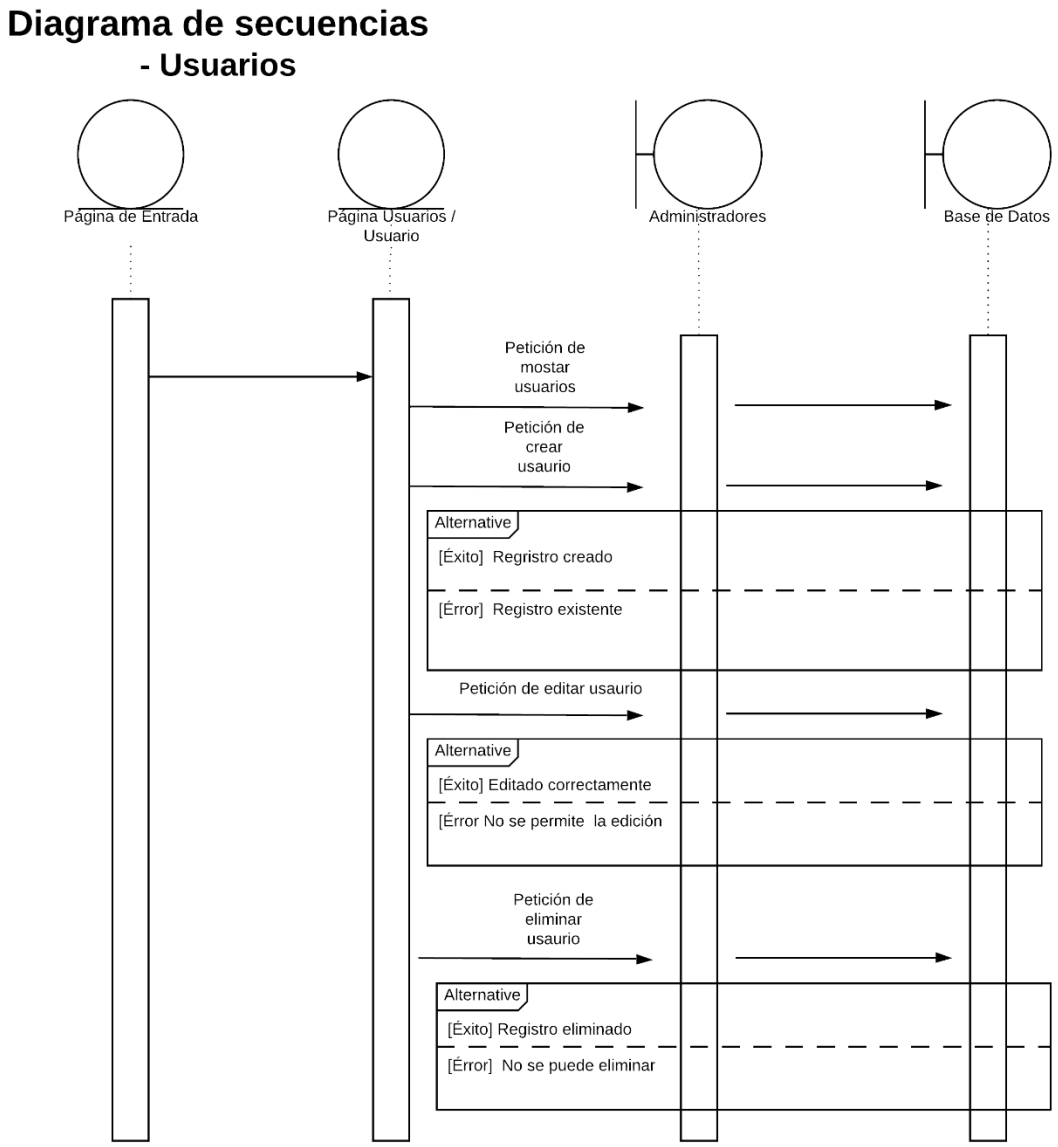

Fig. 3.6. Diagrama de secuencias describiendo el apartado de Usuarios. (Sandoval M,2018)

La secuencia de las consultas se puede observar en la imagen de abajo:

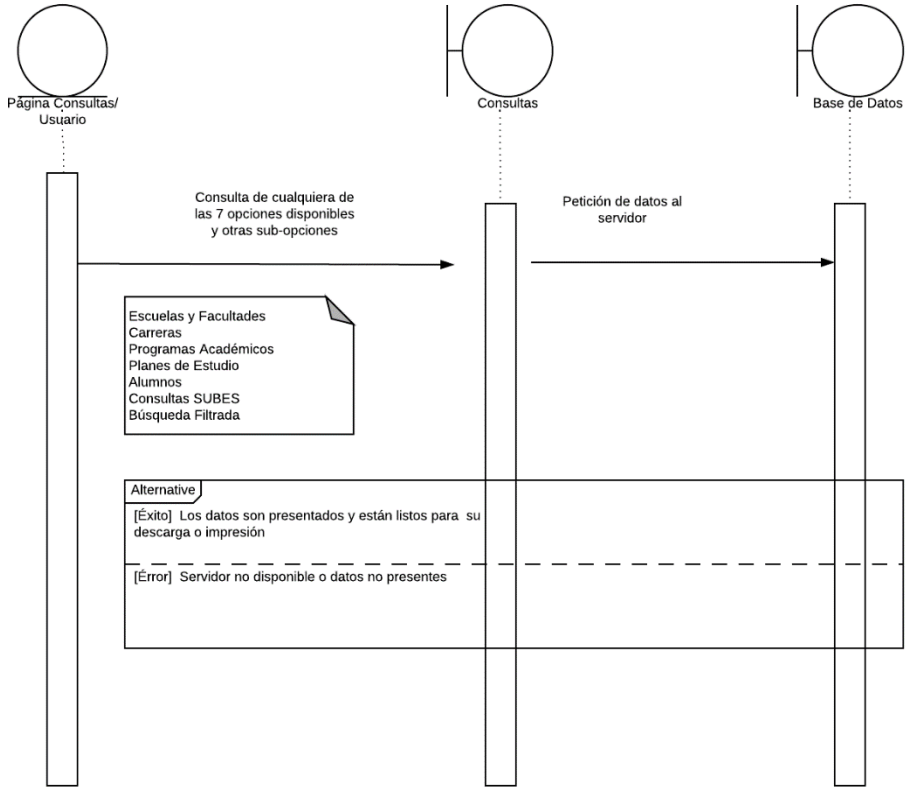

Fig. 3.7. Diagrama de secuencias describiendo las Consultas (Sandoval M,2018)

En el próximo diagrama se muestra el proceso de la consulta de Logs:

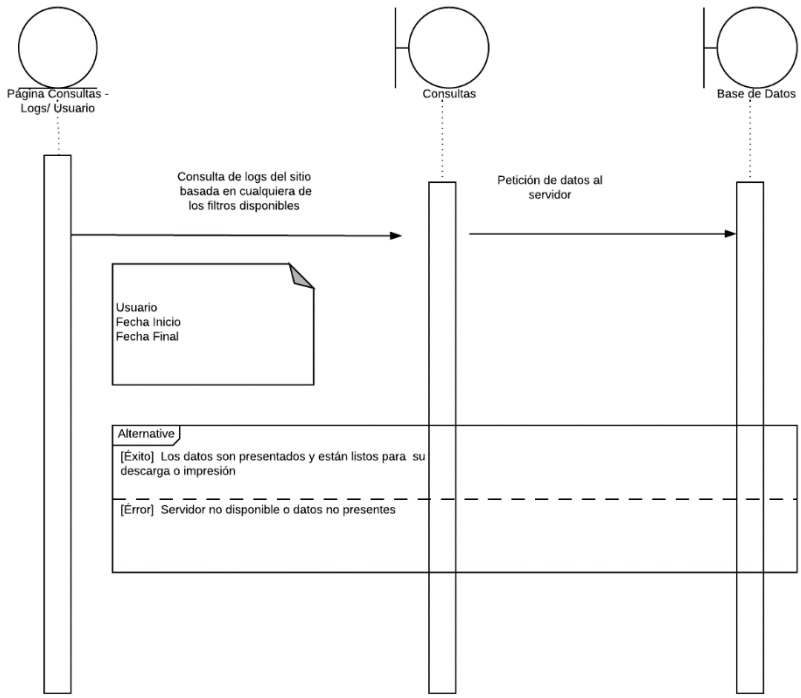

Fig. 3.8. Diagrama de secuencias describiendo el apartado de Logs (Sandoval M,2018)

Y en el último de los diagramas de secuencia se encuentra el de las cargas:

Diagrama de secuencias
- Cargas

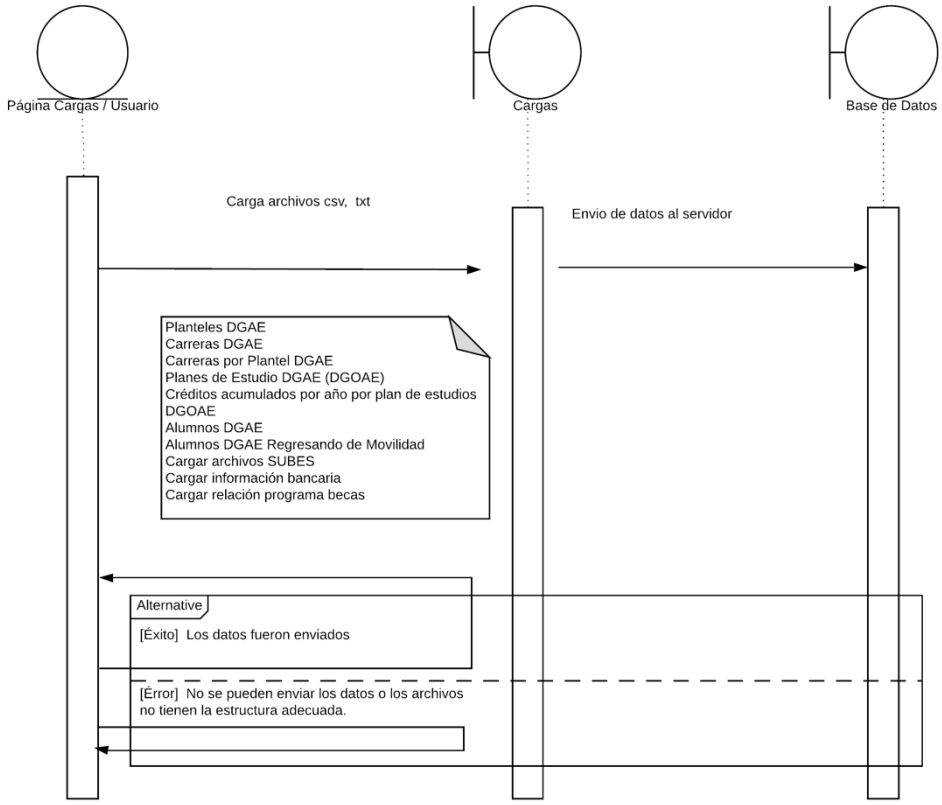

Fig. 3.9. Diagrama de secuencias describiendo las Cargas (Sandoval M,2018)

3.4 Diagramas de estado

A continuación se muestran los diagramas de estados para el Root, el Jefe de Departamento y el administrador:

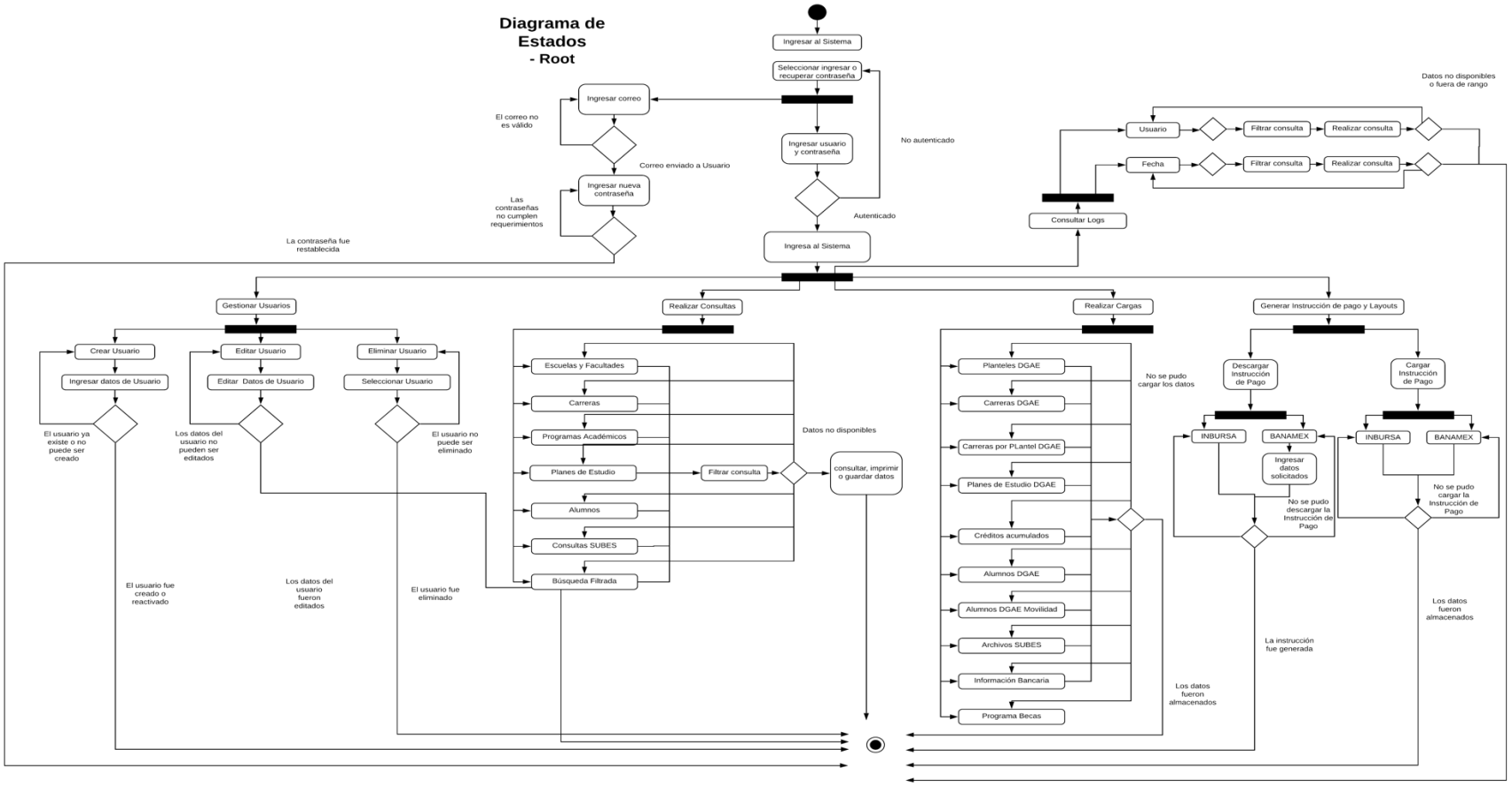

Fig. 3.10. Diagrama de Estados para el usuario Root

(Sandoval M,2018)

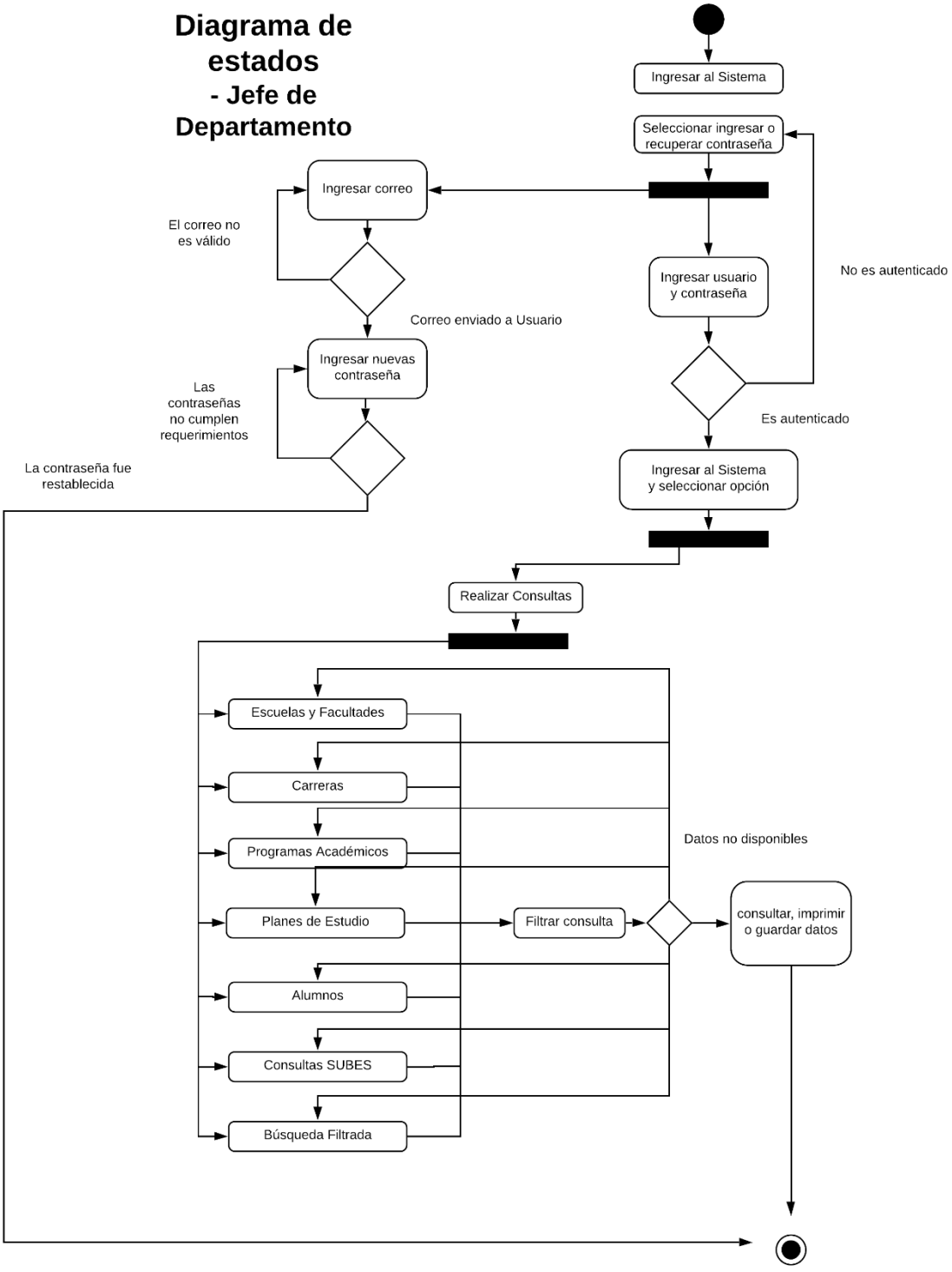

Fig. 3.11 Diagrama de Estados para el usuario Jefe de Departamento (Sandoval M,2018)

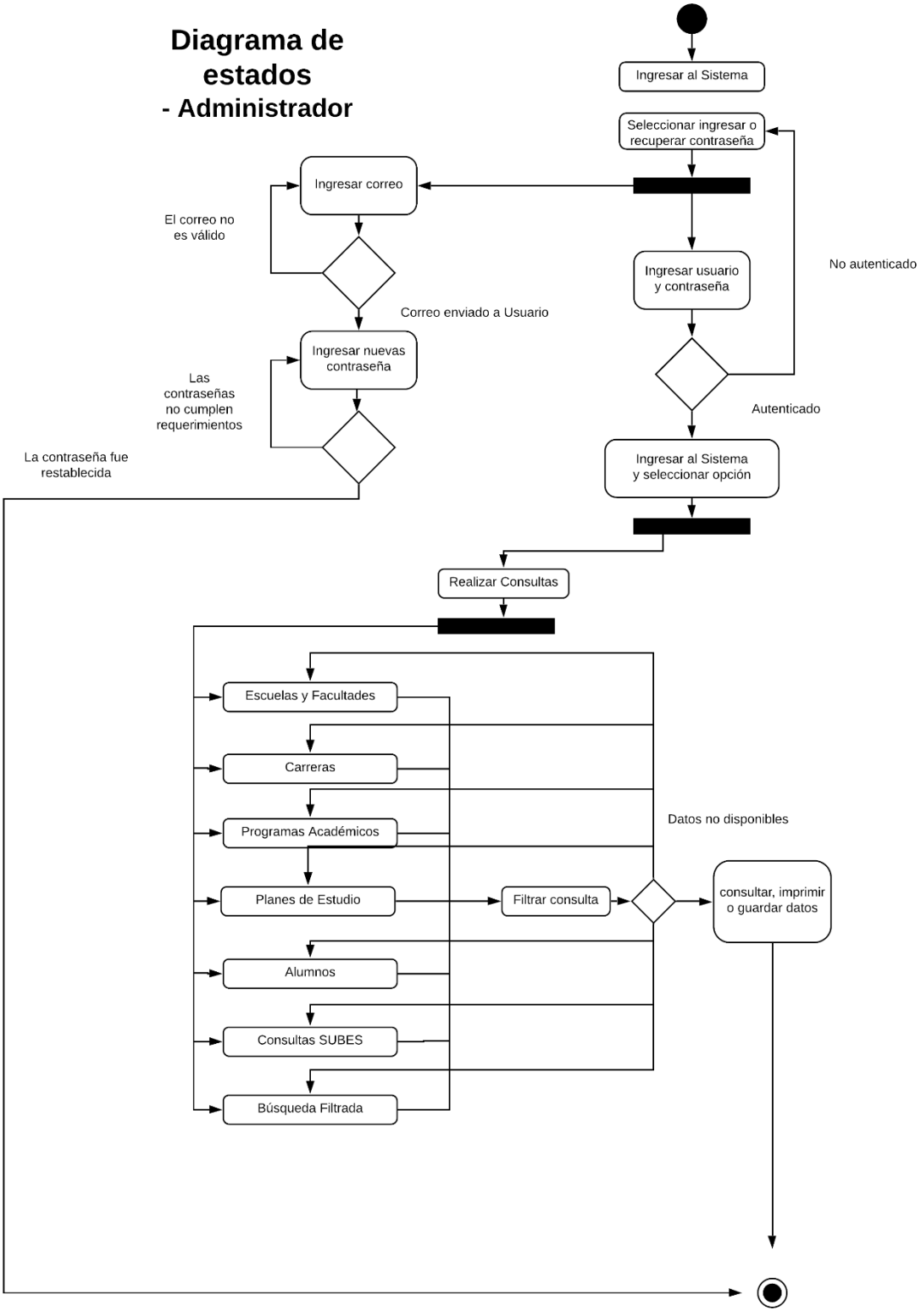

Fig. 3.12. Diagrama de Estados para el usuario Administrador (Sandoval M,2018)

3.5 Diagramas de clases

En este apartado se mostrarán los diagramas de clases para representar el sistema y sus interacciones.

A continuación los paquetes que conforman el sistema:

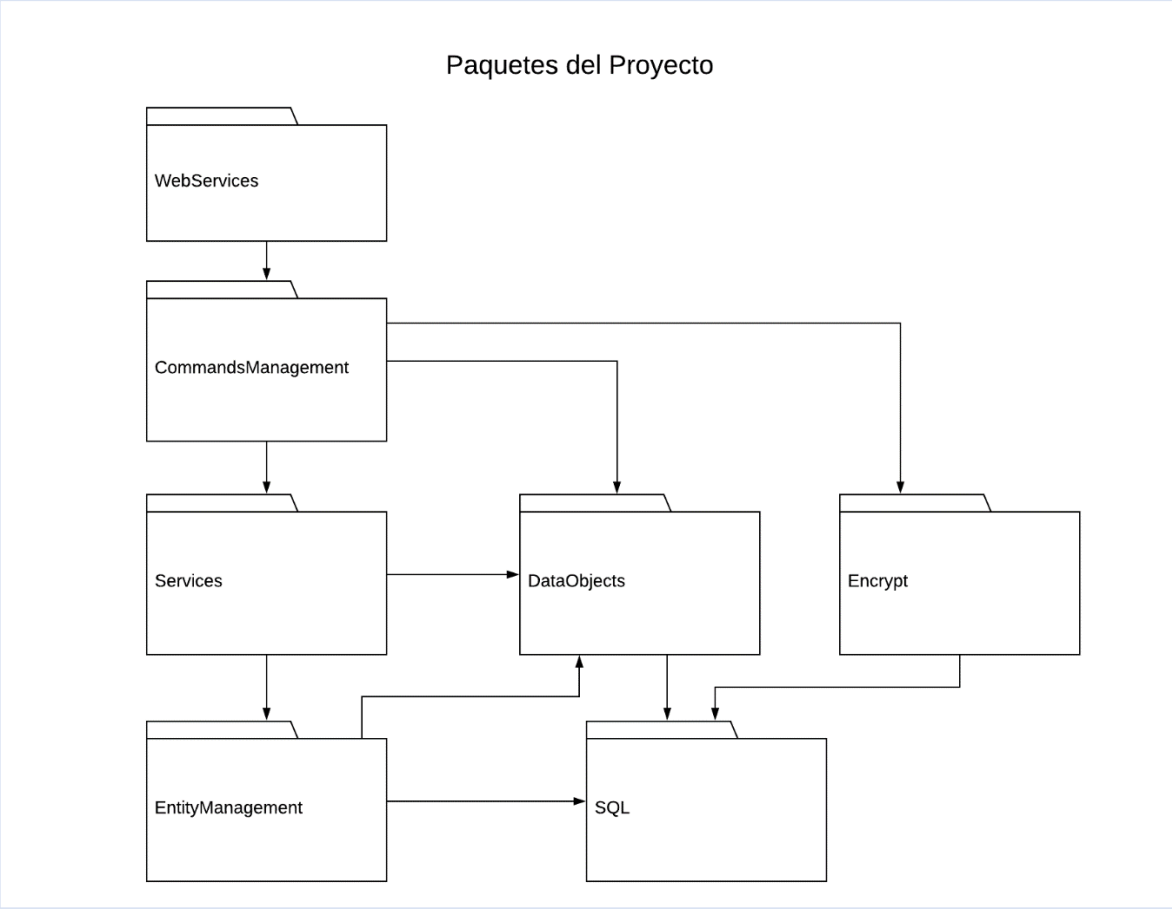

Fig. 3.13. Diagrama UML de los paquetes de Clases en Java (Sandoval M,2018)

El diagrama de la página siguiente muestra con más detalle el contenido de los paquetes:

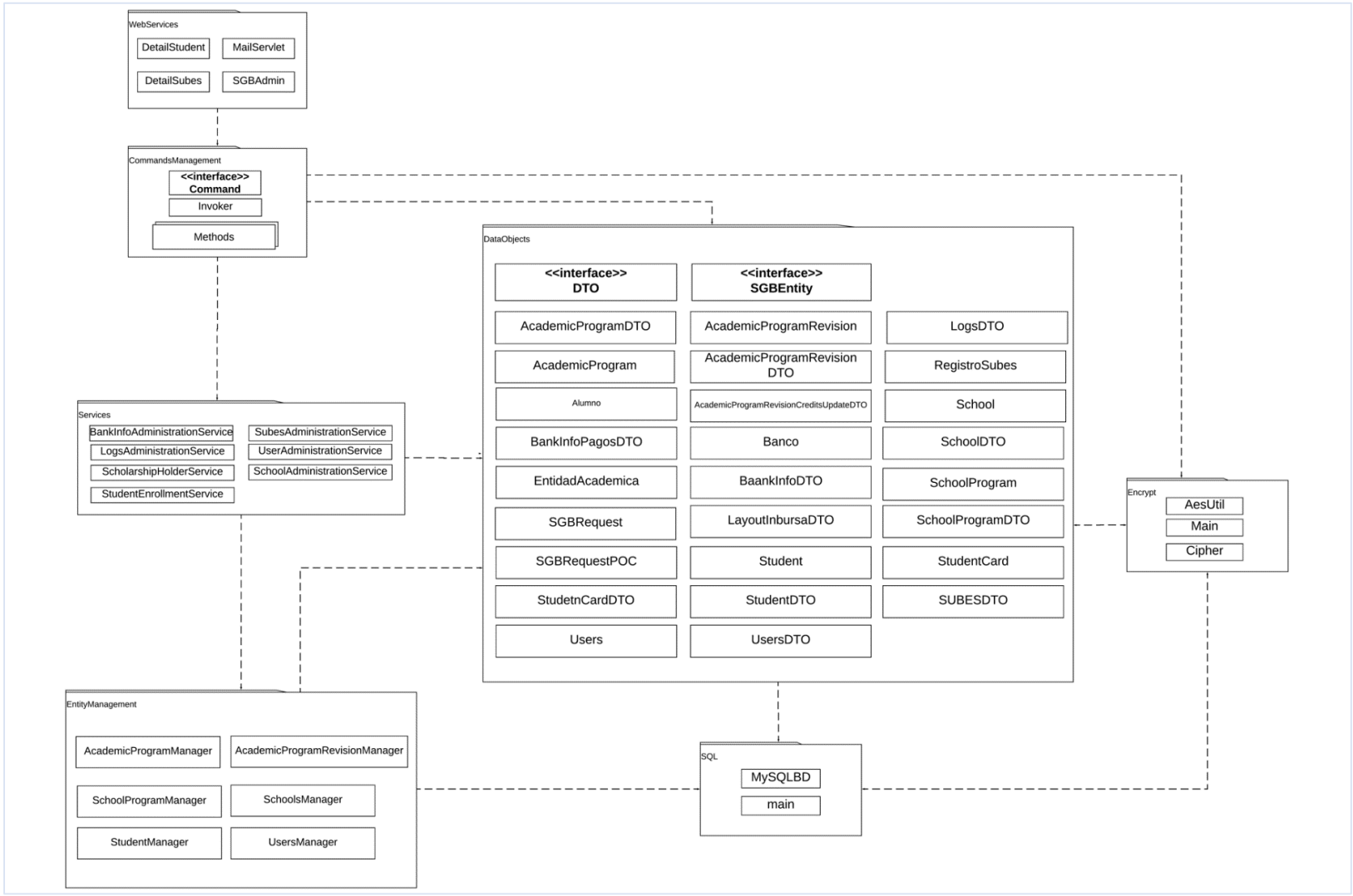

Fig. 3.14. Diagrama UML de paquetes y clases de Proyecto en Java

(Sandoval M,2018)

3.6 Diagrama de la base de datos

La base de datos fue diseñada de manera relacional y a continuación se muestra el correspondiente Diagrama E-R o Entidad Relación, en el que se puede observar la tabla Student como centro de todo el sistema:

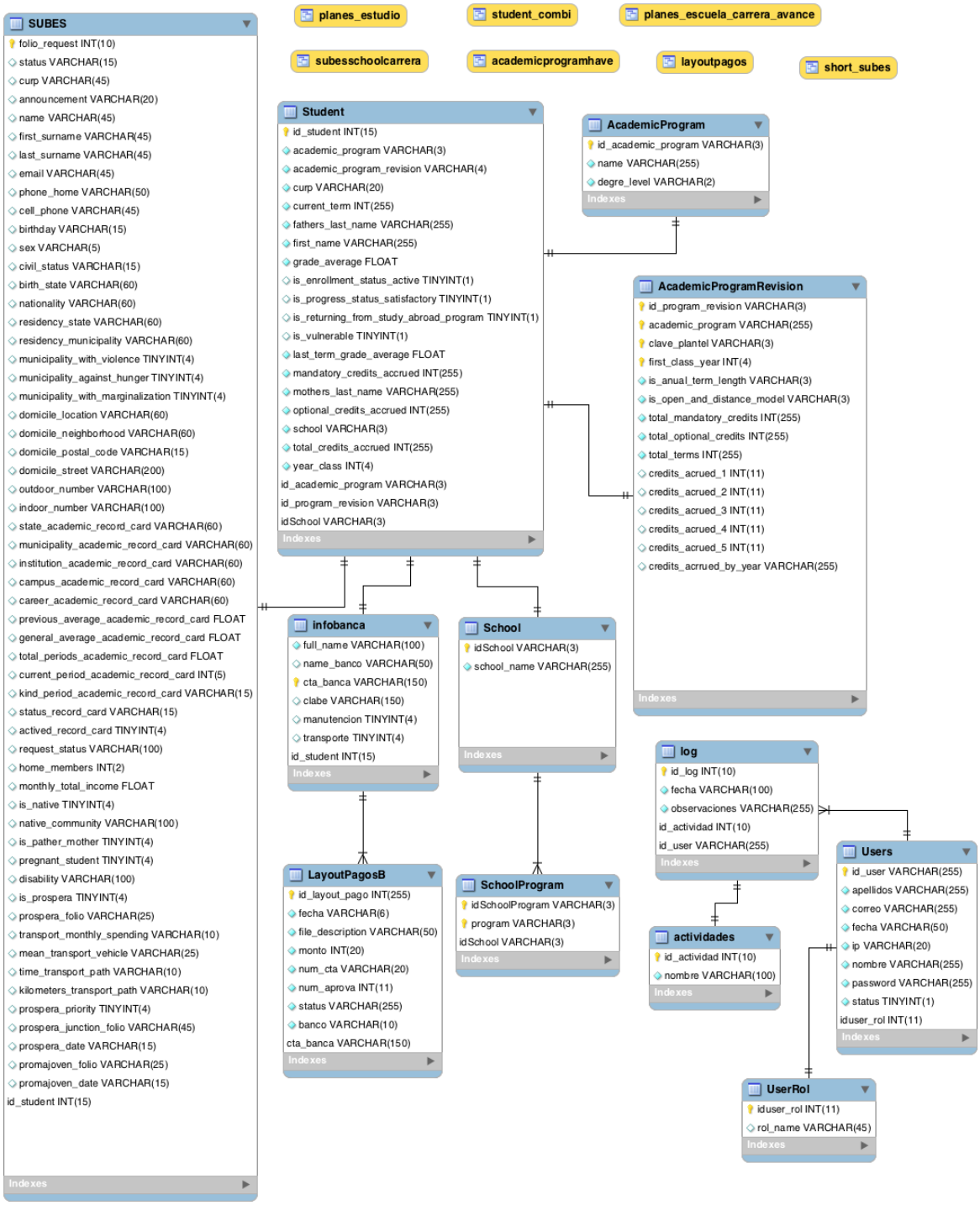

Fig. 3.15. Diagrama de la Base de Datos (Sandoval M,2018)

3.7 Estructura del Código

Para explicar el desarrollo del sistema, se mostrarán algunos segmentos del código realizado en java. Por ejemplo, la clase AesUtil dentro del paquete de encrypt, permite codificar y decodificar cadenas para brindarle seguridad a la información; tal como la contraseña de los usuarios que se codifica a través de una llave. A continuación, un segmento de código de dicha clase:

```
public class AesUtil {

    private static int keySize;
    private static int iterationCount;
    private static Cipher cipher;
    private static Properties encriptionProperties = new Properties();

    static {

        try {
        encriptionProperties.load(Thread.currentThread().getContextClassLoader().getResourceAsStream("mx/unam/becas/encript/Cipher.properties"));

        } catch (IOException ex) {
            throw new RuntimeException(ex);
        }

        keySize = Integer.parseInt(encriptionProperties.getProperty("keysize"));
        iterationCount = Integer.parseInt(encriptionProperties.getProperty("iterationCount"));
        System.out.println("key " + keySize + " iter " + iterationCount);

        try {
            cipher = Cipher.getInstance("AES/CBC/PKCS5Padding");
        } catch (NoSuchAlgorithmException e) {
            throw fail(e);
        } catch (NoSuchPaddingException e) {
            throw fail(e);
        }
    }

    public static String encrypt(String plaintext) {
        try {
            SecretKey key = generateKey(encriptionProperties.getProperty("salt"), encriptionProperties.getProperty("passphrase"));
            byte[] encrypted = doFinal(Cipher.ENCRYPT_MODE, key, encriptionProperties.getProperty("iv"), plaintext.getBytes("UTF-8"));
            return base64(encrypted);
        }
        catch (UnsupportedEncodingException e) {
            throw fail(e);
        }
    }

    public static String decrypt(String ciphertext) {
        try {
            SecretKey key = generateKey(encriptionProperties.getProperty("salt"), encriptionProperties.getProperty("passphrase"));
            byte[] decrypted = doFinal(Cipher.DECRYPT_MODE, key, encriptionProperties.getProperty("iv"), base64(ciphertext));
            return new String(decrypted, "UTF-8");
        }
        catch (UnsupportedEncodingException e) {
            throw fail(e);
        }
    }
```

Fig. 3.16 Código de la clase AesUtil (Sandoval M,2018)

Otro ejemplo es la clase SGBAdminServlet, la cual es encargada de recibir y atender las peticiones con los métodos correspondientes:

```java
import java.io.IOException;

/**
 * Servlet con el webservice para la prueba de concepto del Sistema de Gestión de Becas;
 * toda la cominicación es mediante JSON
 * @author franciscoadam
 * */
@SuppressWarnings("serial")
public class SGBAdminServlet extends HttpServlet {

    @Override
      public void doPost(HttpServletRequest request, HttpServletResponse response)
          throws IOException, ServletException {

        String requestJSON = IOUtils.toString(request.getReader());
        Gson gson = new Gson();
        SGBRequest sgbRequest = gson.fromJson(requestJSON, SGBRequest.class);

        String responseJSON = "Servicio y/o método desconocido";

        response.setContentType("application/json; charset=UTF-8");


        /**Convocatoria =  Call for Applications
         * Candidatos = applicants
         * Becarios = scholarship holder
         * Matrícula = School Enrollment
         * */

        responseJSON = Invoker.getInstance().executeCommand(sgbRequest);
        response.getWriter().println(responseJSON);


    }

}
```

Fig. 3.17 Clase SGBAdminServlet para levantar servicios y atender peticiones (Sandoval M,2018)

Las peticiones de la clase anterior se realizan a través de un archivo de tipo JSON, el cual recibe como parámetros el nombre del servicio, método y otros argumentos:

```
{
Service:"Nombre_del_Servicio",
        "Method":"nombre_del_metodo",
                arguments:[argumentos]
}
```

Fig. 3.18 JSON para atender peticiones (Sandoval M,2018)

Los argumentos del archivo JSON pueden variar dependiendo del servicio, pueden ser un arreglo sencillo o también bidimensionales como se esquematiza a continuación:

```
{
Service:”Nombre_del_Servicio”,
        ”Method”: ”nombre_del_metodo”,
                arguments:[[a,b,c,d],[1,2,3,4]]
}
```

Fig. 3.19 JSON para atender peticiones (Sandoval M,2018)

CAPÍTULO 4.- IMPLEMENTACIÓN Y MEJORAS

4.1 Entorno de desarrollo

El entorno de desarrollo fue establecido en equipos Macintosh con sistema Operativo MacOs Sierrra 10.12.6, aunque también se pudo realizar en equipos con Sistema Operativo Windows o Linux.

4.1.1 Instalación de Java

Para instalar Java, bastó con descargar Java SE Development Kit 7u80 de la dirección URL siguiente:

http://www.oracle.com/technetwork/java/javase/downloads/java-archive-downloads-javase7-521261.html

4.1.2 Instalación de Eclipse y Maven

- **Eclipse:** se instaló el IDE de Eclipse en su versión Oxygen para desarrolladores Java desde el sitio oficial: https://www.eclipse.org/downloads/
- **Maven:** para la gestión del proyecto en java se instaló Maven a través del repositorio oficial de Eclipse accediendo al menú Help -> Eclipse Marketplace…

- **Librerías auxiliares:** dadas las características del proyecto en donde se requiere solicitar datos a una base de datos y cifrar algunos, fue necesario incluir dos bibliotecas; para la conexión a la base de datos se utilizó el conector “mysql-connector-java-5.0.8-bin.jar” y para cifrar los datos “UPSRJCipher.jar”. Dichas bibliotecas fueron añadidas al proyecto por medio de la ruta Eclipse -> Properties -> Java Build Path -> Libraries en donde se señalaron las rutas en donde están guardadas, tal y como se puede ver a continuación:

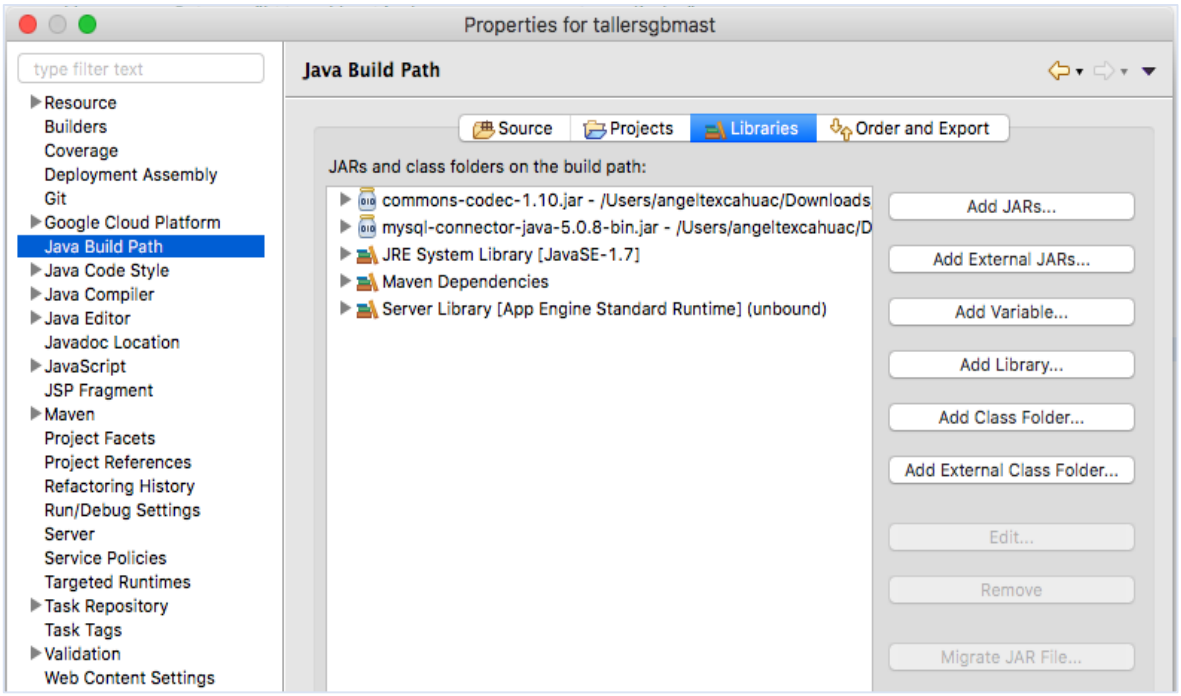

Fig. 4.1 Configuración de bibliotecas (Sandoval M,2018)

4.1.3 Instalación del Plugin App Engine para Eclipse

Para instalar el plug in de App Engine para Eclipse, bastó con descargarlo e instalarlo a través del siguiente enlace de Google Cloud:

https://cloud.google.com/appengine/docs/standard/java/download#java_mac

4.1.4 Generación del proyecto en App Engine

Dentro del panel de preferencias que se accede a través de Eclipse -> Preferences, se indicó la ruta donde está instalado el SDK de Google Cloud:

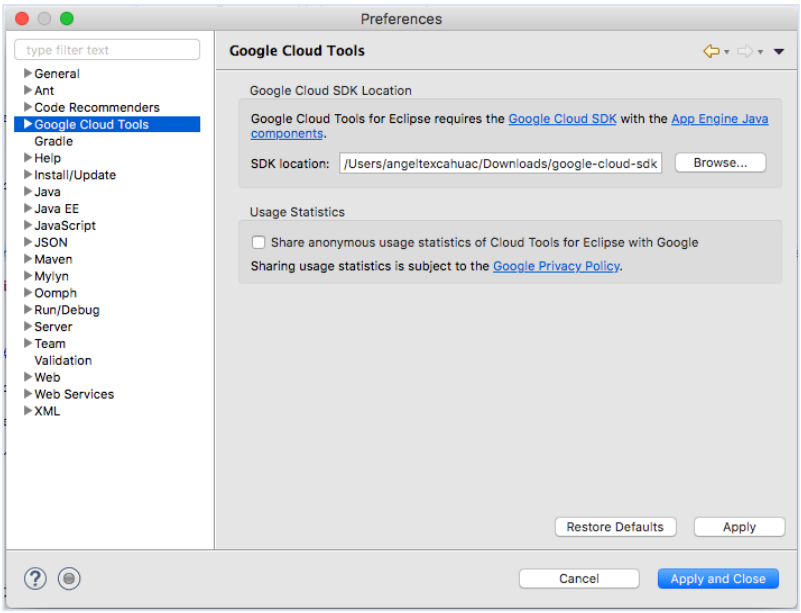

Fig. 4.2 Ruta de acceso al SDK de Google Cloud (Sandoval M,2018)

Una vez habilitado el SDK de Google Cloud fue posible crear el proyecto en Google App Engine, de la siguiente manera:

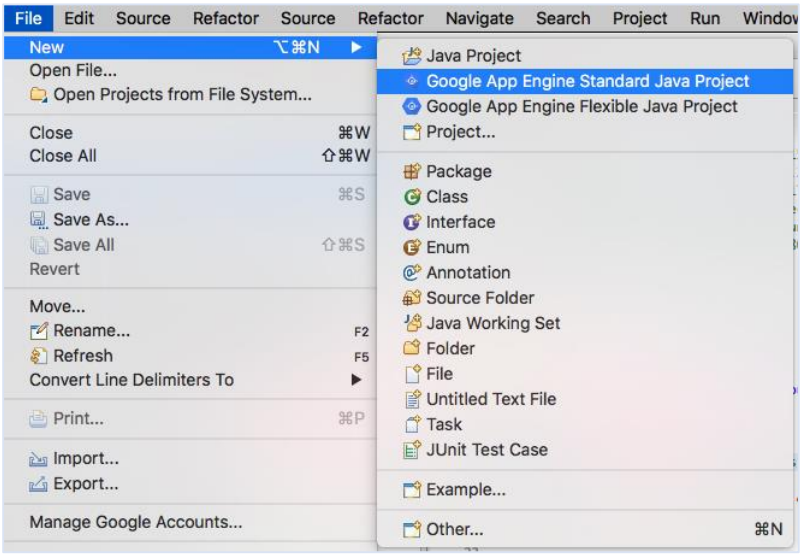

Fig. 4.3 Creación del proyecto Java para App Engine (Sandoval M,2018)

Posteriormente se proporcionaron los siguientes datos a manera de ejemplo para crear el proyecto “sgbadmin”:

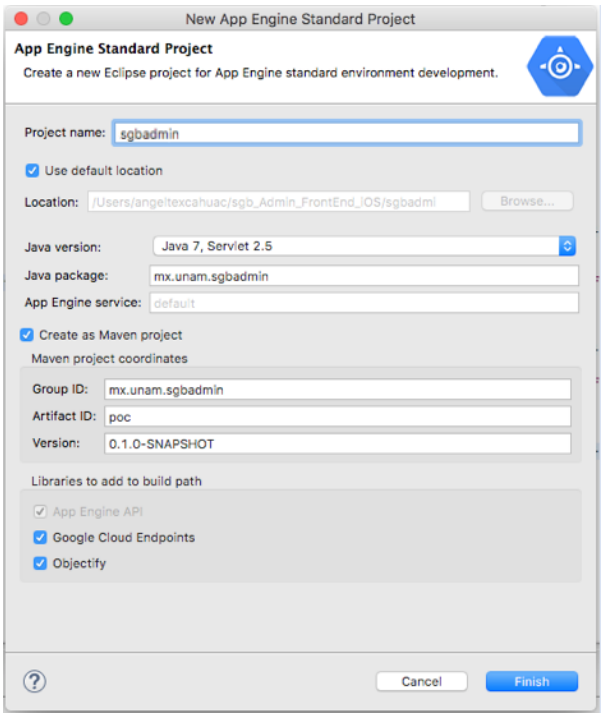

Fig. 4.4 Datos de configuración para la creación del proyecto (Sandoval M,2018)

Para ver el funcionamiento del proyecto se creó un servidor accediendo a la ruta File -> New -> Other -> Server, se seleccionó la opción server, el tipo de servidor “Google” y “App Engine Standard”. En la consola se proporcionó el nombre del servidor y del host y los puertos. Tal y como se muestra en la imagen siguiente:

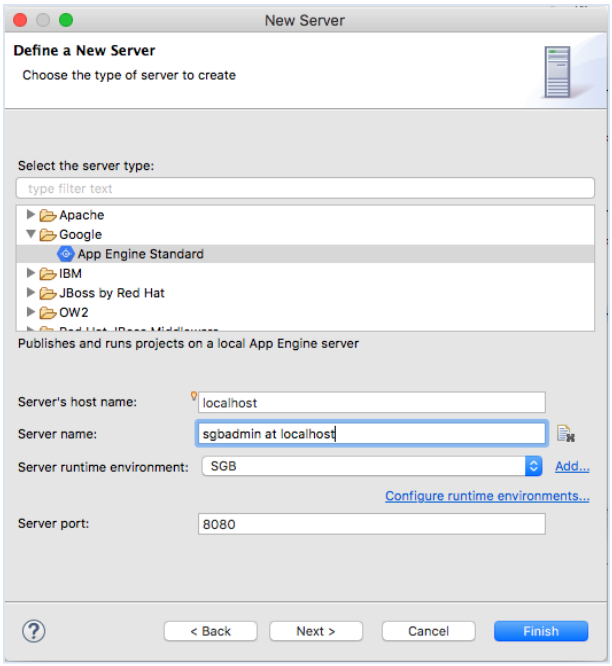

Fig. 4.5 Creación del Servidor Google (Sandoval M,2018)

Una vez creado el servidor se actualizó el proyecto en el panel "Project Explorer", se seleccionó la carpeta raíz y dando clic con el botón derecho se eligió la opción Update Project dentro de Maven.

Posteriormente, se arrancó y verificó el funcionamiento del servidor mediante la opción "Start Server" localizada en el panel "Server". En la figura próxima se puede verificar el mensaje "Hello App Engine", indicando que todo opera de manera satisfactoria:

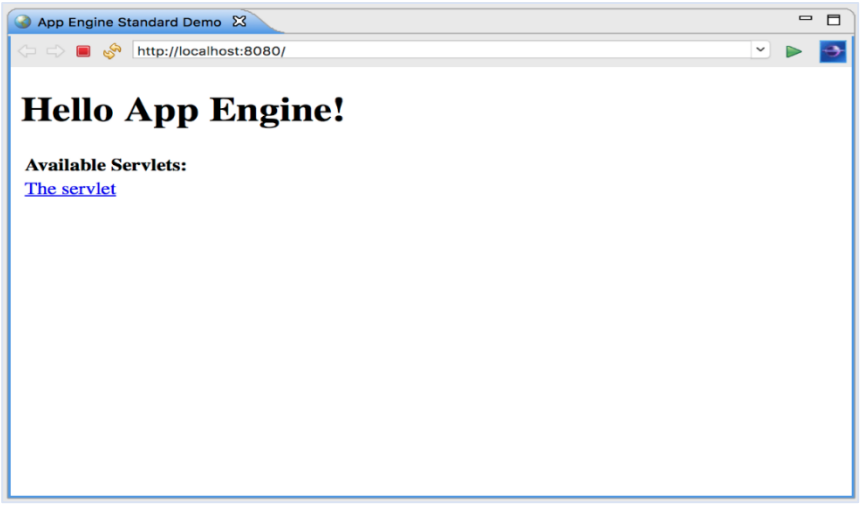

Fig. 4.6 Servidor en funcionamiento (Sandoval M,2018)

4.1.5 Configuración de App Engine

La primera vez que se utiliza App Engine, Google te regala $300 USD de crédito para hacer uso de la plataforma por un año, pero hay que tener en cuenta que esto se puede aprovechar una sola vez por cuenta.

Para hacer uso de los servicios de Google App Engine fue necesario contar con una cuenta de correo electrónico de Gmail, ingresar a través del navegador a la consola de Google Cloud Platform (https://console.cloud.google.com/?hl=es) y llenar un formulario. Al finalizar aparece la página principal de App Engine con el Dashboard:

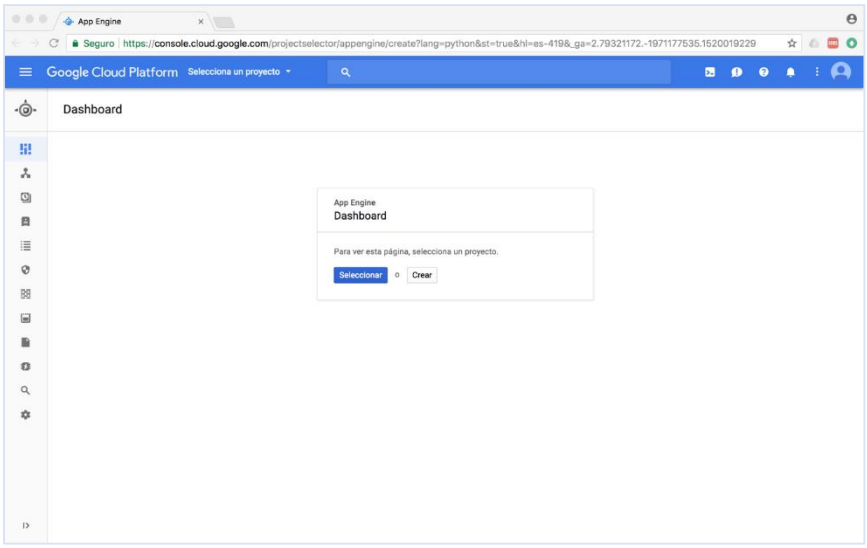

Fig. 4.7 Dashboard del App Engine de Google (Sandoval M,2018)

Al presionar el botón de Crear se muestra otra ventana para agregar el nombre y el id; es importante mencionar que el id servirá de enlace para hacer el *deployment* del proyecto desde eclipse:

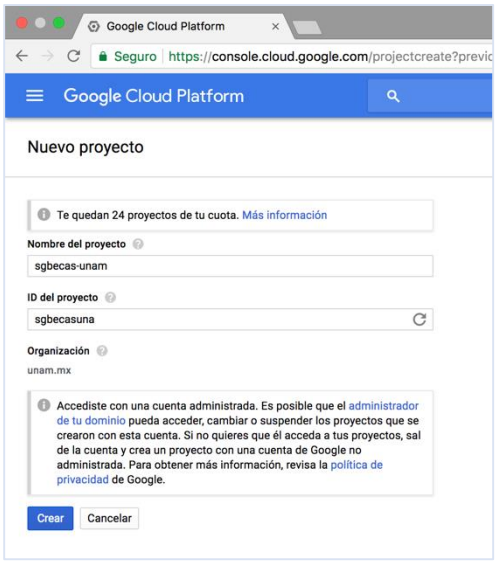

Fig. 4.8 Definición del nombre del proyecto y su id (Sandoval M,2018)

Posteriormente aparece el Panel de Control de Google Cloud Platform, con la información del proyecto y los servicios que se pueden utilizar:

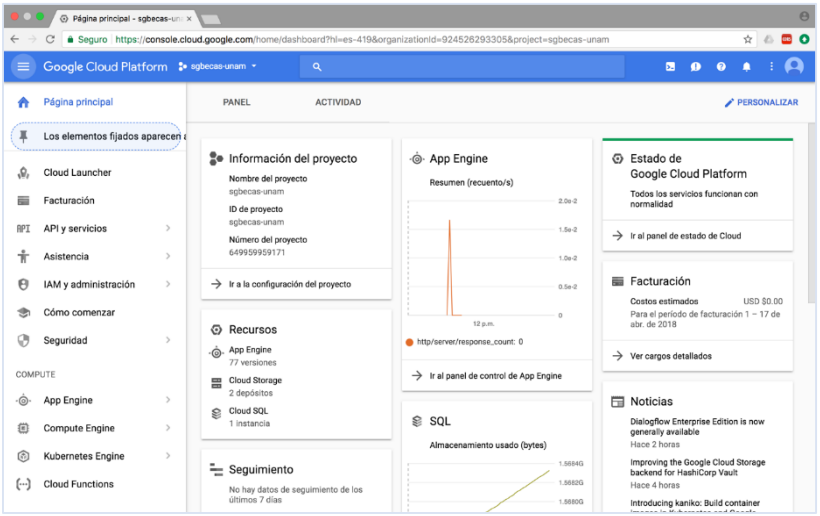

Fig. 4.9 Panel de Control de GCP (Sandoval M,2018)

Si nos desplazamos sobre el menú derecho es posible ver servicios de almacenamiento y si seleccionamos la opción de almacenamiento tipo SQL, nos permitirá crear instancias de base de datos:

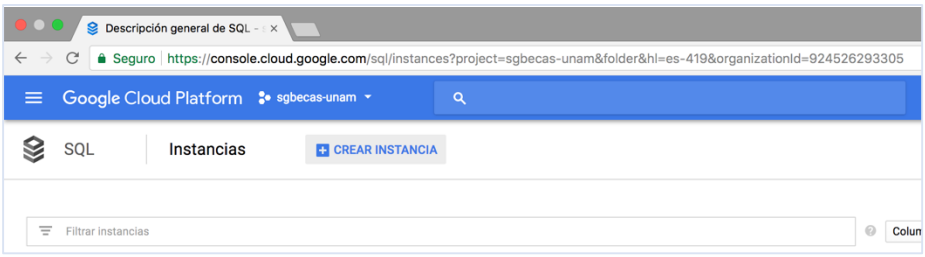

Fig. 4.10 Ventana de creación de instancias SQL (Sandoval M,2018)

Cuando creamos una instancia SQL, GCP nos ofrece dos alternativas de motores de Base de Datos: MySQL y PostgreSQL. En este proyecto de becas se utilizó MySQL.

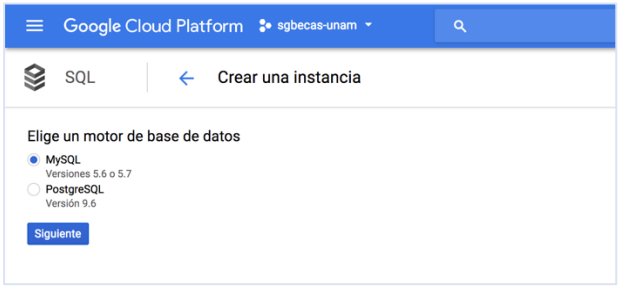

Fig. 4.11 Selección del motor de Base de Datos (Sandoval M,2018)

Una vez seleccionado el motor de Base de Datos se indicó la versión de la instancia, que en este caso fue la segunda:

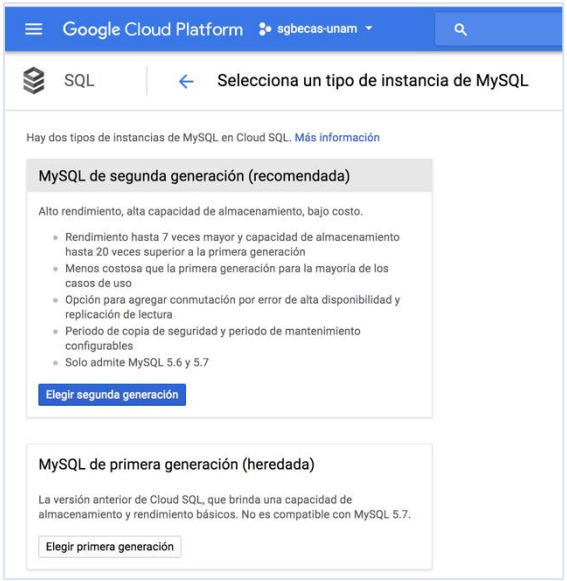

Fig. 4.12 Selección de la versión de la instancia (Sandoval M,2018)

Y finalmente se indica el nombre de la instancia, contraseña, región y ubicación:

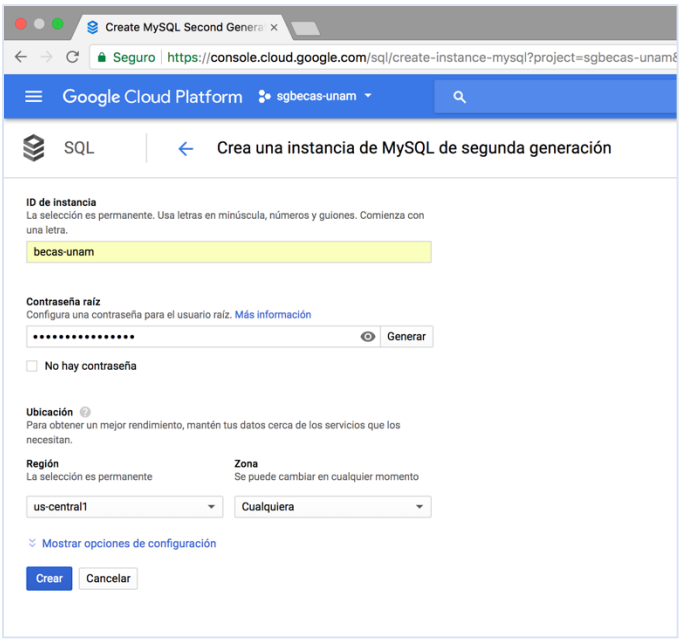

Fig. 4.13 Definición del nombre, contraseña y ubicación de la instancia (Sandoval M,2018)

Al dar clic sobre el botón de Crear, GCP comienza a levantar la instancia con la configuración indicada y una vez terminada, muestra una ventana con información que incluye la IP pública para acceder:

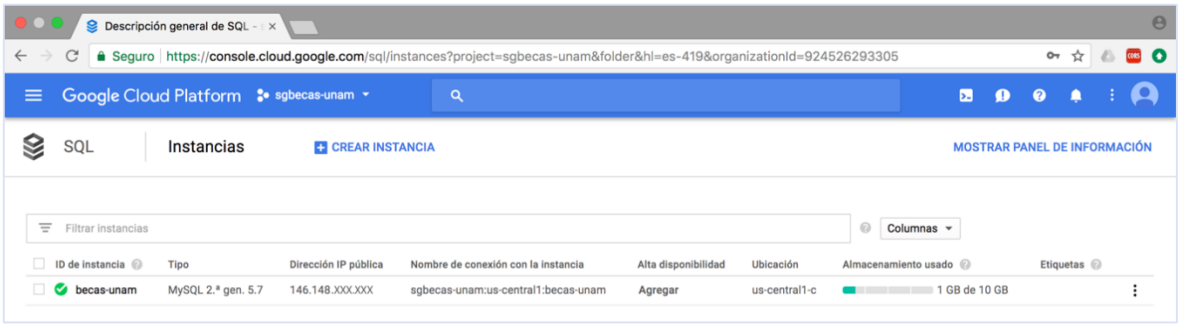

Fig. 4.14 Información de la instancia con la IP pública (Sandoval M,2018)

Para la creación de la base de datos es necesario presionar sobre el *id de instancia*, seleccionar la pestaña *BASES DE DATOS* y dar clic en el botón *Crear base de datos*:

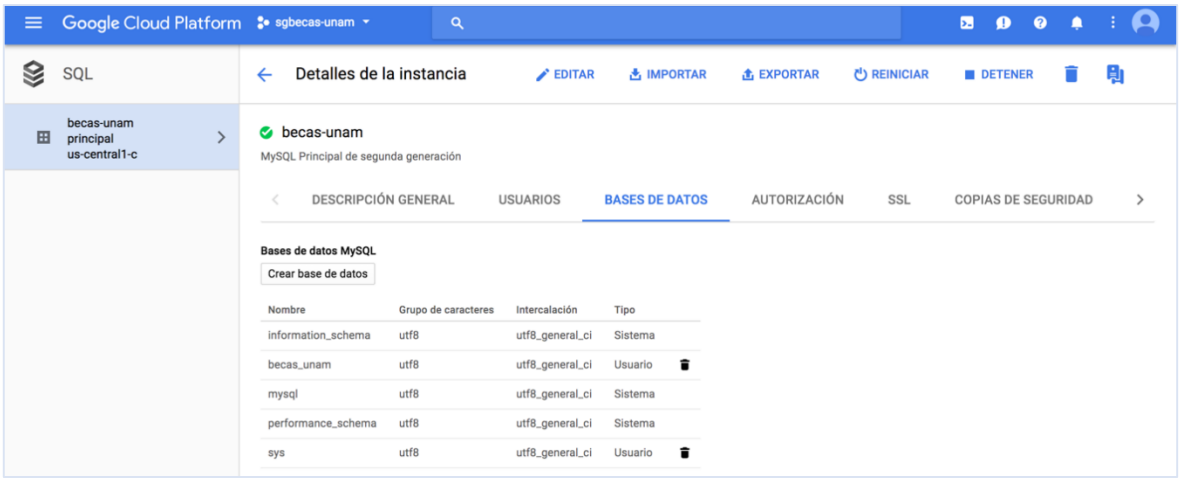

Fig. 4.15 Creación de la base de datos (Sandoval M,2018)

El usuario asignado por defecto a nuestra base de datos es *root*, pero se puede definir uno particular desde el apartado de *USUARIOS*.

También es importante mencionar que para mejorar el rendimiento y procesamiento de los datos, es necesario añadir marcas en la base de datos en la sección de Detalles de la instancia > EDITAR > Añadir marcas de base de datos:

Fig. 4.16 Ajuste de marcas (Sandoval M,2018)

Para poblar nuestra base de datos, es necesario contar con el usuario, IP pública y contraseña ingresada al momento de crear la instancia de SQL. Y para conectarnos es posible hacerlo desde la línea de comandos con la sentencia siguiente:

mysql -u [usuario] -h [dirección IP] -p

Si los datos del usuario e ip son correctos, el sistema solicitará la contraseña para ejecutar la conexión:

```
[Angel@localhost ~]$ mysql -u admin -p -h 146.148.XXX.XXX
Enter password:
```

Fig. 4.17 Acceso a la base de datos desde consola (Sandoval M,2018)

4.2 Entorno de producción

Para realizar la migración de nuestro entorno de desarrollo al de producción fue necesario haber creado previamente nuestro proyecto en App Engine, al igual que la instancia de SQL.

El *deployment* se realizó directamente hacia App Engine con la ayuda de Eclipse y el *plug in* que se instaló. Y el procedimiento fue el siguiente: en la barra de herramientas de Eclipse presionar el icono de Google Cloud Platform para administrar cuentas, crear un nuevo proyecto para App Engine y hacer el *deployment* del proyecto, tal y como se puede observar a continuación:

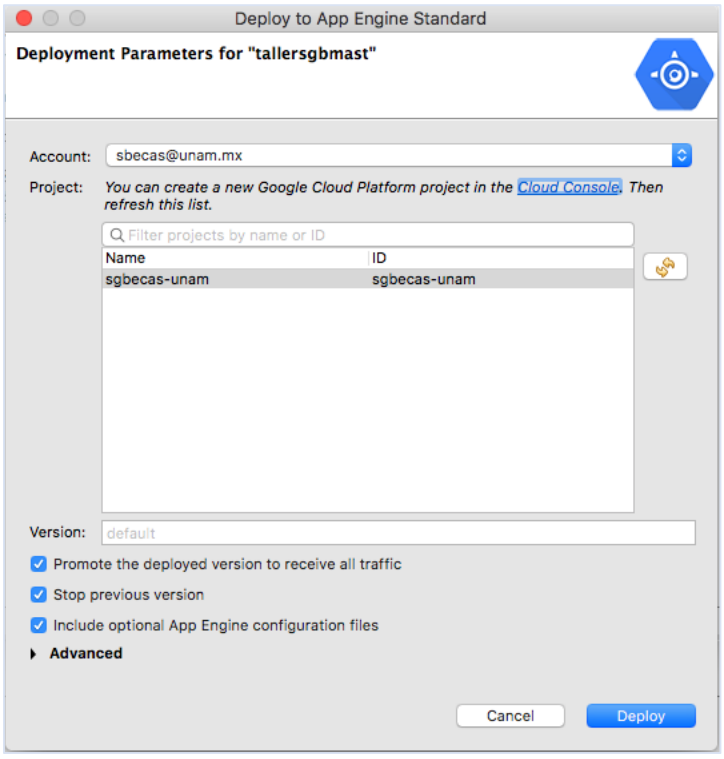

Fig. 4.18 Deployment del proyecto (Sandoval M,2018)

Si los datos de la cuenta son correctos, eclipse mostrará una ventana de consola con los procedimientos en ejecución:

```
Temporary staging for module default directory left in /Users/angeltexcahuac/sgb_Admin_FrontEnd_iOS/.metadata/.plugins/com.google.cloud.tools.eclipse.appengine.deploy/tmp/1526584099249/staging
Services to deploy:

descriptor:      [/Users/angeltexcahuac/sgb_Admin_FrontEnd_iOS/.metadata/.plugins/com.google.cloud.tools.eclipse.appengine.deploy/tmp/1526584099249/staging/app.yaml]
source:          [/Users/angeltexcahuac/sgb_Admin_FrontEnd_iOS/.metadata/.plugins/com.google.cloud.tools.eclipse.appengine.deploy/tmp/1526584099249/staging]
target project:  [sgbecas-unam]
target service:  [default]
target version:  [20180517t140830]
target url:      [https://sgbecas-unam.appspot.com]


Beginning deployment of service [default]...
Some files were skipped. Pass `--verbosity=info` to see which ones.
You may also view the gcloud log file, found at
[/Users/angeltexcahuac/.config/gcloud/logs/2018.05.17/14.08.29.207215.log].
```

Fig. 4.19 Consola de ejecución de procesos de Eclipse (Sandoval M,2018)

Una vez finalizado el deployment de manera satisfactoria, podremos acceder al sistema vía web en cualquier navegador con el *target url* mostrado en la consola anterior. En este caso https://sgbecas-unam.appspot.com:

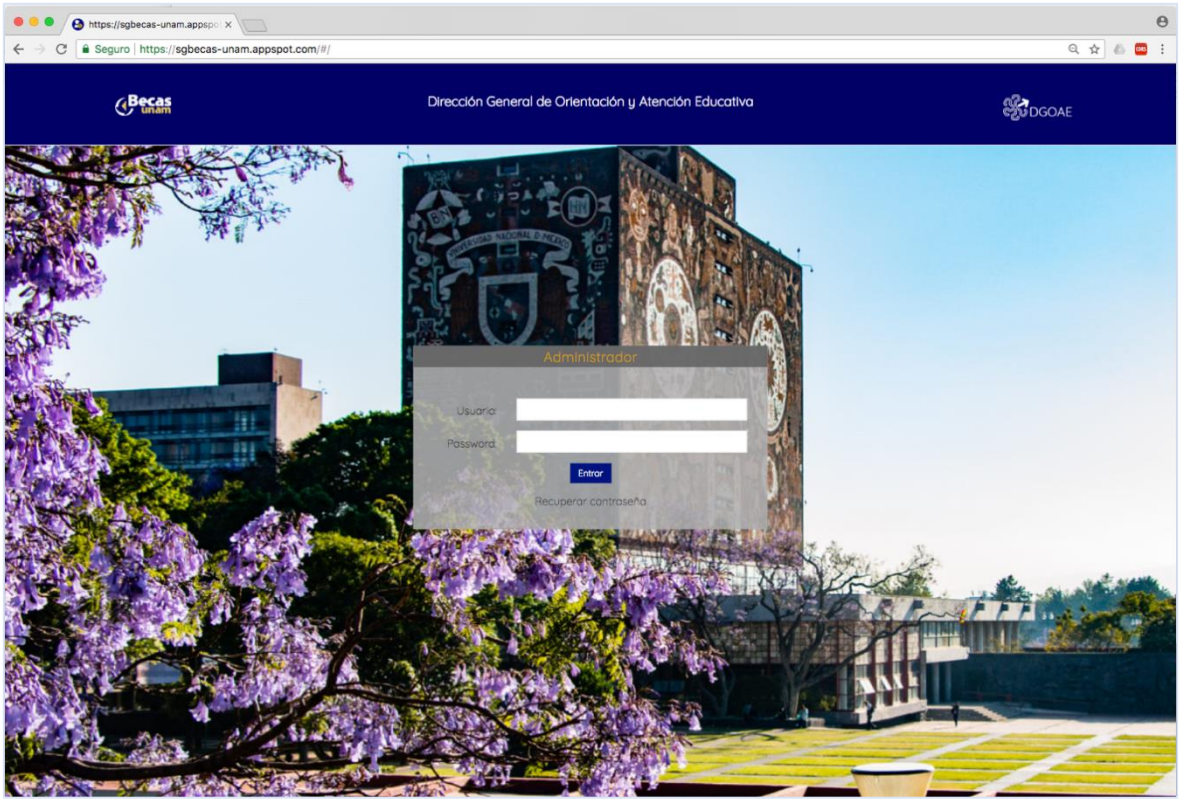

Fig. 4.20 Acceso al sistema de becas en producción (Sandoval M,2018)

4.3 Diseño de la APP

Adicional al sistema web desarrollado, también se realizó la propuesta de una APP o aplicación para dispositivos móviles dirigida para alumnos o aspirantes, la cual deberá funcionar tanto en sistema operativo iOS como Android.

Con esta APP, los alumnos podrán consultar su información académica, bancaria, revisar el estado de los pagos y si llegara a existir algún error con la información, poder dirigirse con el jefe de becas de la institución correspondiente a la brevedad.

Para el desarrollo se hizo la propuesta de llevarlo a cabo con Apache Cordova, el cual permitirá utilizar la misma arquitectura de los servicios ya creados y hacerlo multiplataforma.

En la figura siguiente se muestra la propuesta de interfaz de usuario:

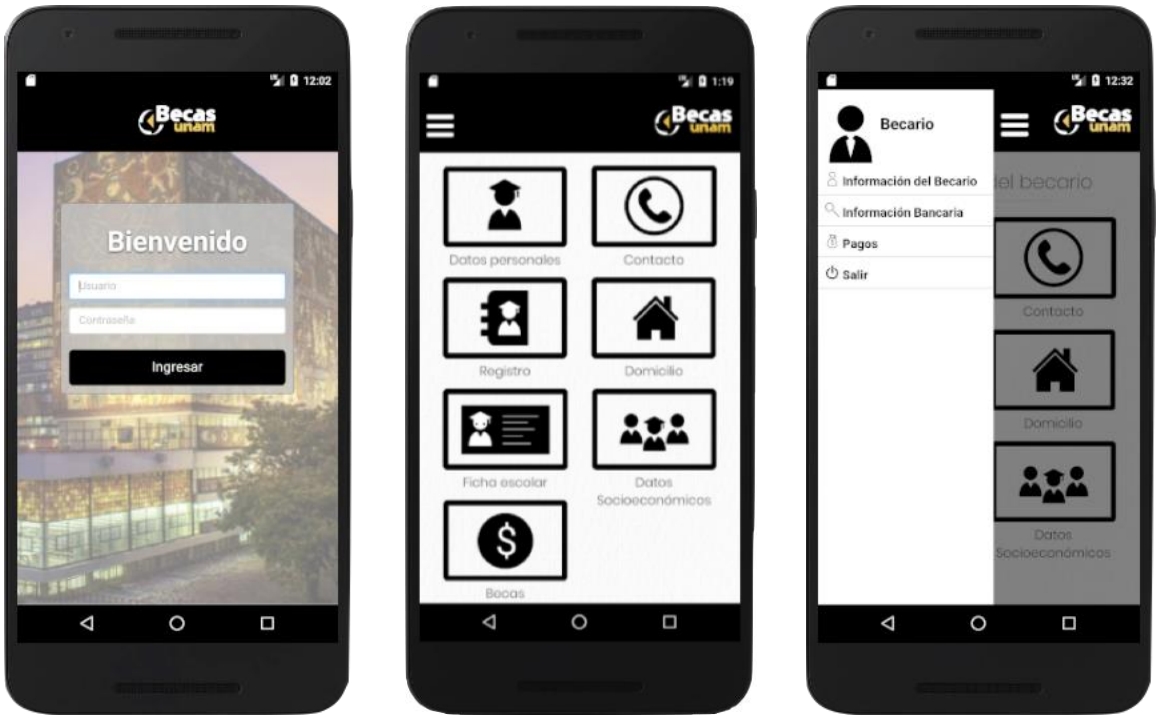

Fig. 4.21 Interfaz de usuario de la APP para dispositivos Android (Sandoval M,2018)

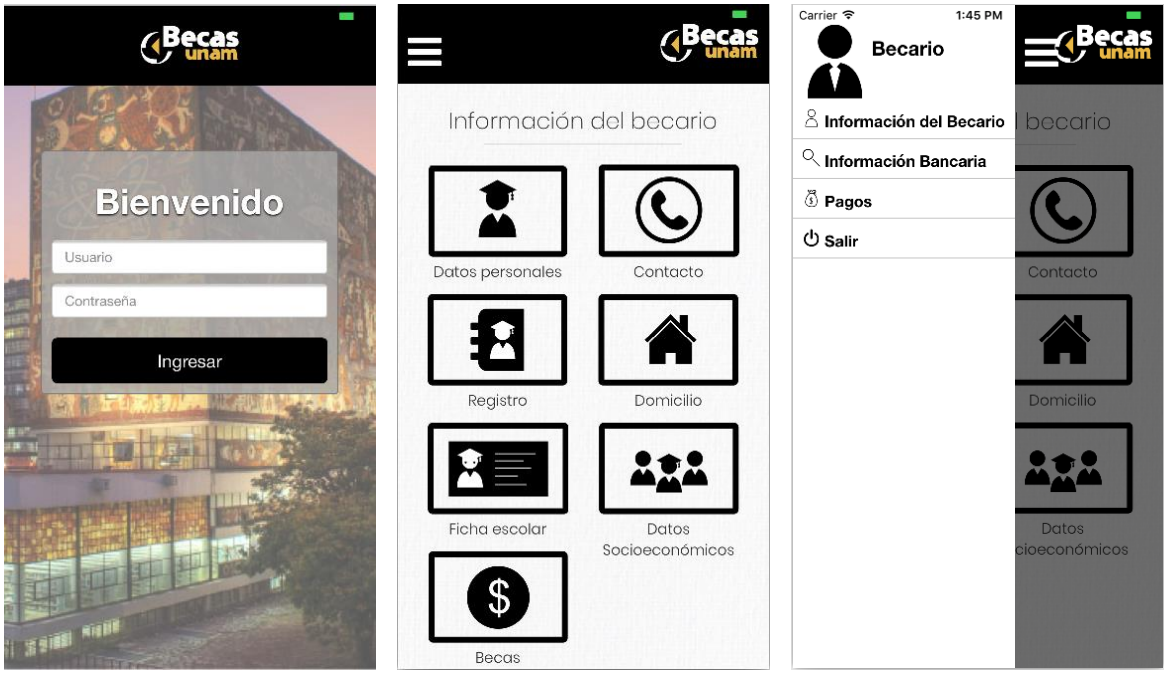

Fig. 4.22 Interfaz de usuario de la APP para dispositivos iOS (Sandoval M,2018)

CONCLUSIONES

Cada año la UNAM gestiona cerca de doscientas mil becas para alumnos de los niveles de bachillerato, licenciatura y posgrado con el objetivo de evitar la deserción, mejorar la permanencia y elevar las condiciones de estudio de su comunidad. Por lo que controlar esta actividad resulta ser muy complicada puesto que intervienen diferentes actores, procedimientos, tiempos y tecnologías.

Gracias a la implementación del *SISTEMA DE GESTIÓN DE BECAS DE LA UNAM (SGB),* el proceso de control de las convocatorias, selección de candidatos, elaboración y gestión de padrones, dispersión de apoyos y creación de reportes, ya no se realiza de forma manual y se han reducido enormemente los errores humanos, propios de la complejidad que lo rodea.

Por todo lo anterior, es posible concluir que resultó del todo benéfico automatizar los procesos de control de becas, puesto que el trabajo es más sencillo, centralizado, con menos errores y sobre todo contribuye a mejorar el alcance de los objetivos, es decir que se asignen los recursos a las personas que más lo necesitan.

BIBLIOGRAFÍA

- Amazon Web Services (2017). ¿Qué es la informática en la nube?, *AWS Amazon*, Recuperado de https://aws.amazon.com/es/what-is-cloud-computing/

- IBM (2018). ¿Qué es el cloud?, *IBM Cloud,* Recuperado de https://www.ibm.com/cloud-computing/mx/es/what-is-cloud-computing.html

- Amazon Web Services. (2017). Tipos de cloud computing, *AWS Amazon*, Recuperado de https://aws.amazon.com/es/types-of-cloud-computing/

- Salesforce. (2000-2017). Cloud Computing-Aplicaciones en un solo tacto, Cloud Computing, Recuperado de https://www.salesforce.com/mx/cloud-computing/

- Iruela J (2015,24 de noviembre). Tipos de nubes en Cloud Computing, INESEM, Recuperado de https://revistadigital.inesem.es/informática-y-tics/cloud-computing-tipos-de-nubes/

- Microsoft Azure (2018). ¿Qué es PasS?, Microsoft Azure, Recuperado de https://azure.microsoft.com/es-es/overview/what-is-paas/

- Microsoft Azure (2018). ¿Qué es SasS?, Microsoft Azure, Recuperado de https://azure.microsoft.com/es-es/overview/what-is-saas/

- IBM (2018). ¿Qué es el cloud?, *IBM Cloud,* Recuperado de https://www.ibm.com/cloud-computing/mx/es/what-is-cloud-computing.html

- Redes Zone (2016,17 de julio). IaaS, PaaS, CaaS, SaaS, ¿Qué significan estos conceptos de Cloud Computing?, *Redes@Zone,* Recuperado de https://www.redeszone.net/2016/07/17/iaas-paas-caas-saas-significan-estos-conceptos-cloud-computing/

- Digital Guide (2018). CaaS: los mejores proveedores de Container as a Service, 1&1Digital Guide, Recuperado de https://www.1and1.mx/digitalguide/servidores/know-how/caas-virtualización-a-demanda/

- IBM (2018). ¿Qué es la nube pública, privada e híbrida?, *IBM Cloud,* Recuperado de https://azure.microsoft.com/es-es/overview/what-are-private-public-hybrid-clouds/

- Martínez F. C. (2010, 11 de noviembre). Cómputo en la nube, *Seguridad Cultura de prevención para TI, Volumen 8, 6-9* Recuperado de https://revista.seguridad.unam.mx/content/c%C3%B3mputo-en-la-nube-0

- Google Developers (2018). Documentation, Google Cloud Platform, Recuperado de https://cloud.google.com/docs/overview/

- W3School (2018). Maps Home, *Google Maps API*, Recuperado de http://www-db.deis.unibo.it/courses/TW/DOCS/w3schools/googleapi/default.asp.html

- Google (2018) Google App Engine, Google Cloud Platform, Recuperado de https://cloud.google.com/appengine/

- W3C (España. 2018). Guía Breve de Servicios Web, *w3c.es*, Recuperado de https://www.w3c.es/Divulgacion/GuiasBreves/ServiciosWeb

- Nicolas Tedeschi (2018). Web Services, un ejemplo práctico, Microsoft, Recuperado de https://msdn.microsoft.com/es-es/library/bb972248.aspx

- IBM. (2013,28 de diciembre). Servicios Web, IBM, Recuperado de https://www.ibm.com/support/knowledgecenter/es/SS7K4U_9.0.0/com.ibm.websphere.zseries.doc/ae/cwbs_wbs2.html

- QODE. (2018) Web Services – REST vs SOAP, QODE, Recuperado de http://qode.pro/blog/web-services-rest-vs-soap/

- Benjamín González C (2007, 7 de abril). SOAP (Simple Object Access Protocol), QODE, Recuperado de https://desarrolloweb.com/articulos/1557.php

- Margaret Rouse .03-12-14. SOAP (Simple Object Access Protocol). Atlanta USA. http://searchmicroservices.techtarget.com/definition/SOAP-Simple-Object-Access-Protocol

- Eduard Tomás (2014, 25 de abril). Qué es REST, Recuperado de https://desarrolloweb.com/articulos/que-es-rest-características-sistemas.html

- BBVAOPEN4U (2016, 23 de marzo). API REST: qué es y cuáles son sus ventajas en el desarrollo de proyectos. Recuperado de https://bbvaopen4u.com/es/actualidad/api-rest-que-es-y-cuales-son-sus-ventajas-en-el-desarrollo-de-proyectos

- Leonardo De Seta. (2018, 13 de noviembre) Introducción a los servicios web RESTful, Recuperado de https://dosideas.com/noticias/java/314-introduccion-a-los-servicios-web-restful.

- IBM (2018). Introducción a la programación Java, parte 1. Developers Works, Recuperado de https://www.ibm.com/developerworks/ssa/java/tutorials/j-introtojava1/index.html

- Oracle (2018). ¿Qué es la tecnología Java y para que la necesito? JAVA. Recuperado de https://www.java.com/es/download/faq/whatis_java.xml

- Arias Ángel (2018). Aprende a programar con Java. Recuperado de https://books.google.com.mx/books?id=OvDBCwAAQBAJ&printsec=frontcover&dq=java&hl=es&sa=X&ved=0ahUKEwis2dOW2d_ZAhXB44MKHcv0BjgQ6AEILjAB#v=onepage&q=java&f=false

- Barrios J. M (2001, 30 de noviembre). Java Servlets, Recuperado de https://users.dcc.uchile.cl/~jbarrios/servlets/general.html

- Oracle, (2013) Tecnología Java Servlet, El tutorial de Java EE6, Recuperado de https://docs.oracle.com/javaee/6/tutorial/doc/bnafe.html