Universidad Continental

FACULTAD DE INGENIERÍA

Escuela Académico Profesional de Ingeniería de Sistemas e Informática

Tesis

Sistema de evaluación crediticia estudiantil en la nube para una ONG de la ciudad de Lima

Adolfo Francisco Ibarra Landeo

Para optar el Título Profesional de Ingeniero de Sistemas e Informática

Huancayo, 2019

ASESOR

Ing. Yuri Márquez Solis

AGRADECIMIENTO

Mediante estas cortas líneas, expreso mi más profundo y sincero agradecimiento al Ing. Yuri Marquez Solis por su total apoyo y su invaluable contribución como mentor y asesor durante el desarrollo de este trabajo de tesis, indispensable para lograr sacar con éxito este proyecto.

A todos y cada uno de mis profesores de la escuela académico-profesional de Ingeniería de Sistemas e Informática y a mi alma máter, la Universidad Continental, porque impregnó en mí el amplio conocimiento de carácter científico durante los seis años que duró mi formación, convirtiéndome en un mejor individuo y un profesional preparado para enfrentar la vida.

DEDICATORIA

A mis padres, Adolfo y Ananí, por darme el apoyo necesario y por la confianza depositada en mí persona. Con la realización de este trabajo logro demostrarles mi compromiso y dedicación con cada proyecto trazado a lo largo de mi vida.

A Alejandro y Astrid, mis hermanos, por su paciencia, compañía y cariño, por animarme a continuar y cuidar de mi equilibrio en todo momento.

A Angela, por motivarme a conseguir mis metas y preocuparse por mí siempre que fue necesario, por su apoyo incondicional y la comprensión de mis necesidades, objetivos y metas. Estaré eternamente agradecido por ser una parte muy importante en todo esto.

A todo aquel que halle en el conocimiento científico, tecnológico y en la investigación los instrumentos idóneos para contribuir al desarrollo de la sociedad y el incremento sustancial del progreso humano.

Adolfo F. Ibarra Landeo.

ÍNDICE

LISTA DE TABLAS

LISTA DE FIGURAS

LISTA DE ANEXOS

RESUMEN

El objetivo de esta tesis es mejorar la evaluación crediticia estudiantil de una Organización No Gubernamental (ONG) de la ciudad de Lima, mediante la Implementación de un sistema de evaluación crediticia estudiantil en la nube con características orientadas a la aceptación y variabilidad de demanda mediante la automatización del proceso de evaluación de una solicitud de crédito académico. Por lo que la solución planteó como objetivos específicos: Diseñar los componentes de software del sistema de evaluación crediticia orientados a lograr tasas aceptables de navegación, accesibilidad, funcionalidad y prevención de errores e identificar, diseñar y desplegar los recursos Cloud que cubran las necesidades de escalabilidad y rendimiento del sistema de evaluación crediticia. Obteniéndose los siguientes resultados luego de aplicarse una encuesta relativa a usabilidad a 19 usuarios: En desacuerdo 3.6 %, no tan de acuerdo 5,3%, parcialmente de acuerdo 14,9%, muy de acuerdo 35,4% y totalmente de acuerdo 40,7%.

Con respecto a escalabilidad, al analizar los resúmenes de uso del servicio proporcionados por Microsoft Azure, se comprobó que el sistema soporta 800 usuarios concurrentes, y en rendimiento un tiempo de respuesta promedio de 470 milisegundos (ms) con esa cantidad de usuarios.

Por lo tanto, al haber logrado los objetivos específicos, concluimos que el objetivo general, de implementar un sistema de evaluación crediticia estudiantil en la nube para una ONG, se ha logrado.

Palabras clave: Sistema de evaluación crediticia, servicios, SOA, Microsoft Azure, nube, WCF, MVC, C#.

ABSTRACT

The objective of this thesis is to improve the student credit assessment of an Non-Governmental Organization (NGO) in the city of Lima, through the implementation of a student credit assessment system in the cloud with characteristics oriented to the acceptance and variability of demand through the automation of the evaluation process of an application for academic credit. So, the solution raised as specific objectives: Design the software components of the credit assessment system aimed at achieving acceptable rates of navigation, accessibility, functionality and prevention of errors and identify, design and deploy the Cloud resources that cover the scalability and performance needs of the credit evaluation system. Obtaining the following results after applying a usability survey to 19 users: Disagree 3.6%, not so agree 5.3%, partially agree 14.9%, strongly agree 35.4% and totally agree 40.7%.

Regarding scalability, analyzing the summaries of service use provided by Microsoft Azure, it was found that the system supports 800 concurrent users, and in performance an average response time of 470 milliseconds (ms) with that number of users.

Therefore, having achieved the specific objectives, we conclude that the general objective of implementing a student credit assessment system in the cloud for an NGO has been achieved.

Keywords: Credit evaluation system, services, SOA, Microsoft Azure, cloud, WCF, MVC, C#.

INTRODUCCIÓN

Este trabajo de tesis tiene por finalidad mejorar la evaluación crediticia estudiantil de una ONG de la ciudad de Lima, considerando que este proceso se llevaba a cabo manualmente y era un trámite necesariamente presencial de aproximadamente 250 postulantes a las becas mensualmente, número que estaba restringido por la forma en que se desarrollaba el proceso antes mencionado.

Por ello, mediante la Implementación de un sistema de evaluación crediticia estudiantil en la nube, con características orientadas a la aceptación y variabilidad de demanda, se buscó la automatización del proceso de evaluación de una solicitud de crédito académico.

La solución planteó como objetivos específicos: Diseñar los componentes de software del sistema de evaluación crediticia orientados a lograr tasas aceptables de navegación, accesibilidad, funcionalidad y prevención de errores e Identificar, diseñar y desplegar los recursos Cloud que cubran las necesidades de escalabilidad y rendimiento del sistema de evaluación crediticia.

El sistema cuenta con características que aseguran su usabilidad y escalabilidad, buscando garantizar su rendimiento adecuado ante cambios bruscos en la demanda de los usuarios que lo utilicen.

El objetivo del proyecto a largo plazo es integrar todos los procesos de la ONG, tanto administrativos como operativos, centralizando toda la información manejada por la misma y brindando un foco de creación de herramientas para toma de decisiones adecuadas.

Este trabajo se divide en cinco capítulos, los cuales se detallan a continuación:

El primer capítulo, “Planteamiento del problema”, explica con buen nivel de detalle el problema identificado, así como los objetivos generales y específicos para el desarrollo de este trabajo de tesis. Asimismo, se menciona la importancia de elaboración de este proyecto de desarrollo.

El segundo capítulo, “Marco teórico”, describe los antecedentes del problema, tanto a nivel de trabajos similares como a nivel del estado de la organización que solicitó la creación de este sistema. Del mismo modo, se explican algunos conceptos necesarios para comprender el escenario de desarrollo de esta solución.

El tercer capítulo, “Metodología”, sustenta las metodologías de gestión del proyecto y de desarrollo del sistema que se eligieron y las razones de su elección.

El cuarto capítulo, “Análisis y diseño de la solución”, presenta los requerimientos que se tomaron en cuenta para el proceso de desarrollo. En este capítulo se realiza un análisis de la viabilidad técnica y económica del desarrollo del sistema de información y estipula un cronograma de actividades. Así también, se expone la arquitectura propuesta a emplear por el sistema y la infraestructura de nube que la soporta. Por último, se presenta el diseño de la base de datos y los primeros diseños de la interfaz gráfica del aplicativo.

Finalmente, el quinto capítulo, “Construcción”, muestra un resumen de las tecnologías empleadas, las configuraciones aplicadas y breves capturas relevantes para el desarrollo de la solución. Asimismo, se describen en este capítulo la estrategia de pruebas y los resultados obtenidos de las mismas. Los resultados obtenidos luego de aplicarse una encuesta relativa a usabilidad a 19 usuarios fueron: En desacuerdo 3.6 %, no tan de acuerdo 5,3%, parcialmente de acuerdo 14,9%, muy de acuerdo 35,4% y totalmente de acuerdo 40,7%.

Con respecto a escalabilidad, al analizar los resúmenes de uso del servicio proporcionados por Microsoft Azure, se comprobó que el sistema soporta 800 usuarios concurrentes, y en rendimiento un tiempo de respuesta promedio de 470 ms.

El autor

CAPÍTULO I
PLANTEAMIENTO DE ESTUDIO

En este capítulo se detalla el contexto del problema al que se pretende dar solución con este estudio, para su mejor comprensión. Asimismo, se detallan los objetivos generales y específicos que guían esta tesis y se explican las razones que justifican un trabajo de este tipo.

1.1. PLANTEAMIENTO Y FORMULACIÓN DEL PROBLEMA

1.1.1. PLANTEAMIENTO DEL PROBLEMA

La ONG en estudio opera en la ciudad de Lima y tiene por objetivo fomentar y facilitar la educación de los profesionales que no cuenten con los medios económicos para acceder a maestrías y diplomados, mediante el otorgamiento de becas y créditos estudiantiles. Ha realizado la adquisición de los servicios para el desarrollo de un sistema web, denominado “Sistema de Administración de Beneficios Educativos”. Este sistema ha sido pensado para funcionar de forma integral dentro de la organización, cubriendo los procesos netamente internos y de más alto nivel, así como también aquellos referidos a la evaluación de solicitudes de becas y créditos.

Durante el proceso de análisis del sistema, saltaron observaciones acerca del proceso de evaluación de solicitudes de postulantes, motivo por el cual llegaron al acuerdo de que el sistema no cumpliría esta función, sino más bien iba a estar enfocado a controlar los procesos de recursos humanos y otros internos dentro de la organización.

Actualmente, con más de 55 años de existencia, la ONG se encuentra la búsqueda de expansión de sus aliados estratégicos, integrando nuevas instituciones educativas a la lista de aquellas que les brindan los programas de estudio. Por otro lado, están en la búsqueda de ampliación de su fondo de crédito educativo, objetivo que piensan lograr gracias a los acuerdos con ciertas entidades bancarias. Lamentablemente, estos acuerdos y la integración de nuevos aliados se han visto suspendidos debido a ciertas exigencias por parte de estos actores. Las entidades bancarias exigen una tasa mínima de atención, que actualmente la ONG no puede cubrir debido al tipo de manejo que poseen de la información. A esto se suma que algunas instituciones educativas les exigen un canal compartido para gestionar las solicitudes de los postulantes que deseen atender cursos de sus instituciones.

La decisión de dejar de lado la creación de los módulos de atención y evaluación de solicitudes para becas en diplomados y postgrados ha derivado en:

- Uso de tablas de Microsoft Excel y documentos de Microsoft Word, usualmente con información redundante o incompleta, que se manejan a través de carpetas compartidas o son enviados por correo electrónico para su modificación.
- Retrasos excesivos en las respuestas y atención de las solicitudes.
- Elevada tasa de errores.
- Demoras en la entrega de información.
- Extensión de los plazos de toma de decisiones.
- Incremento significativo en los costos.
- Falta de control sobre los cambios realizados en los documentos compartidos.
- Riesgo de seguridad de cara a terceros.

Bajo este escenario, la ONG ha solicitado la creación de un sistema que le permita realizar la evaluación de las solicitudes antes mencionadas. El sistema cumple inicialmente esa función, aunque se plantea que, en un mediano plazo, pueda integrarse más funcionalidad al mismo para que integre las funciones del sistema actual.

Por este motivo se ha realizado el diseño, desarrollo y despliegue de un sistema para evaluación crediticia estudiantil, para superar los problemas antes descritos, los cuales dificultaban incluso su crecimiento como organización, el alcance a nuevos postulantes y la formación de nuevas relaciones estratégicas.

1.1.2. FORMULACIÓN DEL PROBLEMA:

¿Cómo mejorar la evaluación crediticia estudiantil de una ONG de la ciudad de Lima?

Problemas específicos:

¿Cómo obtener alta aceptación del sistema encargado del proceso de registro de los estudiantes en el sistema de evaluación crediticia?

¿Cómo atender la alta variabilidad e incremento de demanda de postulantes de evaluación crediticia estudiantil?

1.2. OBJETIVOS

1.2.1. OBJETIVO GENERAL

Implementar un sistema de evaluación crediticia estudiantil en la nube para una ONG de la ciudad de Lima con características orientadas a la aceptación y variabilidad de demanda.

1.2.2. OBJETIVOS ESPECÍFICOS:

- Diseñar la interfaz y componentes de software del sistema de evaluación crediticia en la nube que permitan el uso agradable, intuitivo y tiempo de aprendizaje aceptable por parte del estudiante y de los colaboradores de la ONG, expresado en valoración de navegación, accesibilidad, funcionalidad, ayuda y prevención de errores.
- Identificar, diseñar y desplegar los recursos Cloud que cubran las necesidades de escalabilidad y rendimiento del sistema de evaluación crediticia de una ONG de la ciudad de Lima.

1.3. JUSTIFICACIÓN E IMPORTANCIA:

La herramienta a desarrollar significa un importante aporte para esta ONG, la cual no veía únicamente afectadas sus actividades, sino que también dificultaban el proceso de adopción de becas a los postulantes, tanto para los programas de postgrado como de diplomado ofrecido. La elaboración de este proyecto sirve como base al momento de considerar ciertos mecanismos o tecnologías, así como su integración, para proyectos similares.

El proyecto no sólo beneficia a la ONG en cuestión, sino que también les permite a los postulantes tener conocimiento del estado de su solicitud y realizar por sí mismos la carga de su información y realizar la comprobación de las mismas, quitando esa responsabilidad a los operadores que cumplen esa función y permitiéndoles invertir mejor su tiempo en atender una mayor cantidad de solicitudes.

El desarrollo de este proyecto garantiza tener un mejor control sobre la información manejada de forma interna por parte del equipo de la ONG, así como para evitar inconsistencias y retrasos con la información manejada. El resultado de este punto es beneficioso tanto para postulantes, como para los colaboradores de la ONG.

CAPÍTULO II
MARCO TEÓRICO

En este capítulo se brindará un acercamiento más cercano al contexto en el espacio temporal en el que se está realizando la investigación, partiendo de estudios de tesis relacionados a las tecnologías o metodologías descritas. También, se explicará un poco de la teoría en la que se basa el desarrollo del trabajo presentado y se incluirá un pequeño glosario para definir los términos que son necesarios para comprender la tesis.

2.1. ANTECEDENTES DEL PROBLEMA

2.1.1. TESIS ANTERIORES

Para el presente trabajo se han consultado dos trabajos de tesis extranjeras y una nacional.

2.1.1.1. Trabajos Internacionales

a. **Tascón Millán, Jessica Mercedes y Sanabria Romero, Alejandro Arturo** (1) presentaron su trabajo de tesis denominado "Diseño e implementación de la arquitectura orientada a servicios (SOA), en el desarrollo de un software para la empresa CONCIVIN LTDA.", para optar al título de Ingeniero de Sistemas en la Universidad San Buenaventura, en la ciudad de Bogotá D.C., Colombia.

El objetivo general de este trabajo fue el de diseñar una solución basada en una Arquitectura Orientada a Servicios (SOA, en inglés) para la empresa CONCIVIN, la cual utilizaba herramientas como hojas de cálculo de Microsoft Excel para el manejo de su información, cayendo en problemas como redundancia de datos, inseguridad por accesos externos, tiempos altos de respuesta, entre otros.

El aporte de la tesis citada se da considerando que desarrollaron un sistema SOA para mejorar un proceso que se daba de forma manual, situación muy parecida al problema identificado en la presente tesis.

Entre las conclusiones a las que llevó este estudio se encuentran:

- El uso de SOA implica obtener un profundo entendimiento de los procesos del negocio para optimizar y modificar la menara en que se dan ciertos procesos.
- El uso de sistemas de información adecuados permite a las organizaciones tener una respuesta rápida a las condiciones del mercado, tan cambiantes en la actualidad.
- Resulta de vital importancia que los servicios se construyan teniendo en mente su reutilización, ya que es uno de los principios fundamentales de SOA.

b. **Arévalo Navarro, José Manuel** (2) presentó su trabajo de tesis denominado “Cloud Computing: fundamentos, diseño y arquitectura aplicados a un caso de estudio”, para optar al grado de Máster en Tecnologías de la Información y Sistemas informáticos en la Universidad Rey Juan Carlos, en la ciudad de Madrid, España.

El objetivo de este trabajo de investigación fue el de estudiar el paradigma de Cloud Computing, aplicando este paradigma a un caso real de estudio, usando un sistema basado en servicios para lograrlo (SOA).

La importancia de citar este trabajo se da debido a que en él se hace un análisis exhaustivo sobre las implicaciones de desplegar sistemas SOA en entornos Cloud, haciendo una comparativa de costos y detallando las dificultades que se presentan en este tipo de escenarios.

Entre las conclusiones a las que llevó este estudio se encuentran:

Los costos de implementación de una arquitectura compleja suelen ser constantes y elevados, pero ofrecen una característica importante de escalabilidad real de los sistemas.

La integración de sistemas basados en SOA implica un esfuerzo mínimo para su adaptabilidad a entornos de nube.

2.1.1.2. Trabajos Nacionales

a. Cedillo Crisosto, Franco Eduardo (3), presentó su trabajo de tesis denominado “Análisis, diseño, implementación e integración de un sistema de gestión de casos y un SoftPhone Web para un centro de contacto virtual con múltiples casos de comunicación” para obtener el título de Ingeniero Informático en la Pontificia Universidad Católica del Perú (PUCP), ubicada en la ciudad de Lima, Perú.

   El objetivo general de este trabajo fue el de realizar la implementación de los componentes de software para un centro de Contacto Virtual con múltiples medios de comunicación, para proveer de una herramienta simple y de fácil uso para la comunicación con la central destinada a la atención de los clientes, todo mediante el uso de un navegador web.

   La importancia de este trabajo de tesis para el desarrollo del presente trabajo radica en el uso de la metodología dX, una variante similar a Agile Unified Process (AUP), la que alinearon con la metodología SCRUM. En el presente trabajo se usó una variante de AUP alineada a la metodología SCRUM para el desarrollo del Sistema de Evaluación Crediticia para la ONG.

A nivel de arquitectura, se propuso una solución basada en tres capas: de datos, de negocio y de presentación, haciendo uso del protocolo IAX2 para comunicar los servidores VoIP, encargados de realizar el registro de los usuarios y el envío del identificador de usuario al que se realizaría la llamada.

Entre las conclusiones a las que llevó este estudio se encuentran:

- La gestión del proyecto de desarrollo es importante para lograr los objetivos establecidos en el tiempo planificado.
- El uso de la metodología SCRUM le permitió realizar un avance paralelo de cara a las tareas que tenían programadas durante todas las etapas de desarrollo. Además, esta metodología le permitió adaptarse al uso de diversas herramientas y brindó flexibilidad al momento de realizar la toma de decisiones.
- Si bien es cierto que la funcionalidad es un factor crítico para el cierre exitoso de un proyecto, hay factores de la presentación del mismo, como la usabilidad y facilidad de acceso que también son altamente deseables.

2.1.2. ANTECEDENTES DE LA ORGANIZACIÓN

Durante el desarrollo del presente trabajo, se evitará mencionar el nombre de la organización para la que se está desarrollando el sistema, debido a que no se poseen los permisos para divulgar su identidad.

La ONG para la que se desarrollará el sistema tiene una trayectoria que supera los 55 años, y está dedicada a la promoción y la creación de facilidades para individuos que no poseen los recursos económicos necesarios para continuar con su educación. Actualmente se encuentra ampliando sus convenios con universidades y entidades bancarias para ampliar su fondo de crédito educativo y, por consiguiente, tener mayores posibilidades de otorgar facilidades a más estudiantes.

La ONG posee un sistema desarrollado, el Sistema de Administración de Beneficios Educativos, el cual se planteó con el objetivo de facilitar el cruce de información necesario entre la ONG, las entidades bancarias, las universidades y los postulantes, con el objetivo de atender las solicitudes de becas a los postulantes para las diferentes modalidades a las que se presentaban. Durante el desarrollo de este sistema, se levantaron algunas observaciones y determinaron que no se atenderían las solicitudes para postgrados y diplomados. El Sistema de Administración de Beneficios Educativos fue ganando peso y funcionalidades a medida que se fue desarrollando, orientándose a cubrir los procesos internos de la organización principalmente, pasando a utilizar documentos en tablas de Microsoft Excel para atender las solicitudes que no serían contempladas por este sistema. Esta decisión derivó en los siguientes problemas: (Fuente: ONG en estudio)

- Tiempos prolongados para la atención de las solicitudes.
- Redundancia de información.
- Alta frecuencia de errores introducidos en diversos archivos de Excel.
- Demoras en la entrega de información a las áreas correspondientes y en la atención de las solicitudes.
- Extensión de los plazos de toma de decisiones.
- Incremento de los costos.
- Falta de control sobre los cambios realizados en la información.
- Riesgo de acceso a la información confidencial por parte de actores externos a estos procesos o a la misma organización.
- Poca flexibilidad para elaboración de cambios en los formatos de los documentos.
- Múltiples versiones de los documentos manejándose en paralelo en diferentes áreas.
- Dificultad para presentar la información correspondiente y segregada a las universidades y bancos.

Por este motivo, la ONG solicitó la creación de un sistema integral, que tendría inicio con la creación de un módulo de atención de las solicitudes de los postulantes a becas para postgrado y diplomados. Asimismo, este sistema

debería integrarse como un módulo externo en un trabajo posterior y permitir enfrentar los problemas antes mencionados de forma exitosa. Se debió considerar que los usuarios del sistema tienen las siguientes formas de acceso por su naturaleza:

Tabla 1: Formas de acceso por usuario.

USUARIO	TIPO DE ACCESO
Colaboradores de la ONG	Local / Web
Universidades	Web
Bancos	Web
Estudiantes	Web

Fuente: ONG en estudio.

2.1.3. RESTRICCIONES DE LA ORGANIZACIÓN

a. La ONG en estudio tiene muy claro que su rubro de negocio es el Educativo, por lo tanto, no tiene ninguna intención de mantener sus propios servidores.

b. Es Política de la ONG implantar sus sistemas utilizando Microsoft Azure, ya que poseen una alianza estratégica comercial, la cual genera costos mucho menores de hosting y soporte comparado con otras soluciones (On-Premise, Google Cloud, Amazon Web Services, entre otras).

c. Los plazos que habían previsto estaban ajustados debido a que las entidades bancarias solicitaban, a corto plazo, un mecanismo efectivo para el control de las solicitudes de los postulantes. Este era un requisito indispensable para poder ampliar el fondo de crédito disponible. Del mismo modo, los diplomados y los programas de postgrado ofrecidos por las universidades eran otro punto de presión, motivo por el cual se tuvo que recurrir a métodos de desarrollo que permitan flexibilidad y brinden entregas funcionales para ser evaluadas por el equipo de Tecnologías de la Información (TI) de la ONG.

d. A todo esto, se suma que, en vista de las dificultades que tuvieron con la administración de servidores y datos físicos en el pasado, se estaba

realizando un contrato para adquisición de licencias de nube, específicamente para los servicios de Microsoft Azure. Por todas estas razones se analizó y planteó el escenario como se detalla en el presente documento.

2.1.4. ESTADO DE ATENCIÓN DE SOLICITUDES

La información provista por la ONG muestra los registros realizados desde 07 de marzo del 2016 hasta 12 de marzo del 2016, en una semana típica. Esta información se muestra en la Tabla 2.

Tabla 2: Solicitudes atendidas por la ONG con proceso tradicional.

HORA	LUNES	MARTES	MIÉRCOLES	JUEVES	VIERNES	SÁBADO
9:00 - 9:30	0	5	0	4	2	5
9:30 - 10:00	5	3	2	2	5	2
10:00 - 10:30	4	1	1	2	1	1
10:30 - 11:00	5	0	5	1	3	4
11:00 - 11:30	4	1	0	3	4	0
11:30 - 12:00	4	4	5	1	0	5
12:00 - 12:30	0	0	0	2	3	2
12:30 - 13:00	3	5	2	2	4	2
13:00 - 13:30	0	0	0	0	0	0
13:30 - 14:00	0	0	0	0	0	0
14:00 - 14:30	0	1	4	4	4	3
14:30 - 15:00	0	2	2	4	5	5
15:00 - 15:30	3	0	2	4	3	3
15:30 - 16:00	0	0	4	5	5	5
16:00 - 16:30	5	5	0	1	0	4

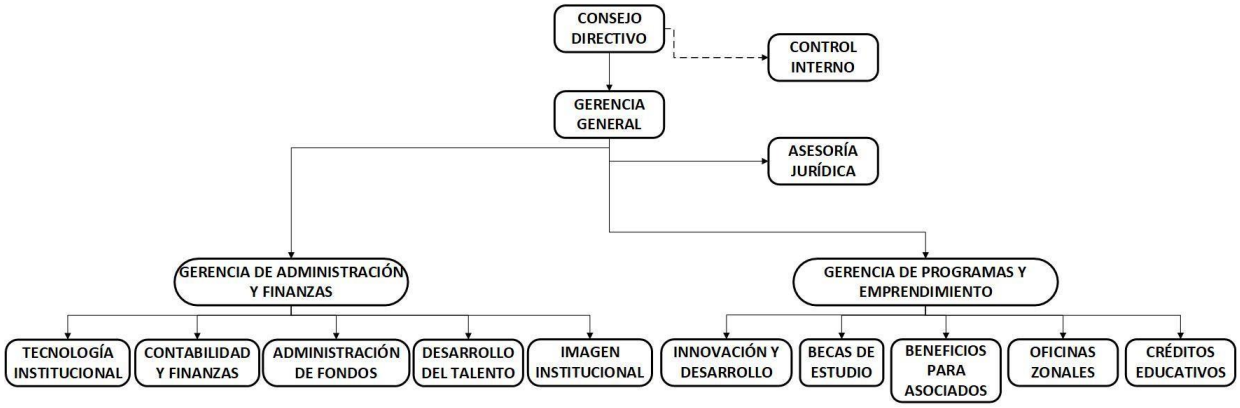
16:30 - 17:00	3	0	4	3	3	3
17:00 - 17:30	4	1	4	0	2	0
17:30 - 18:00	3	5	5	4	3	0

Fuente: ONG en estudio.

2.1.5. ORGANIGRAMA

La Figura 1 muestra el organigrama de la organización, del cual se desprenden las áreas de Tecnología Institucional, Becas de estudio y Créditos Educativos, que son aquellas con las que se involucrará el sistema en esta primera etapa.

Figura 1: Organigrama

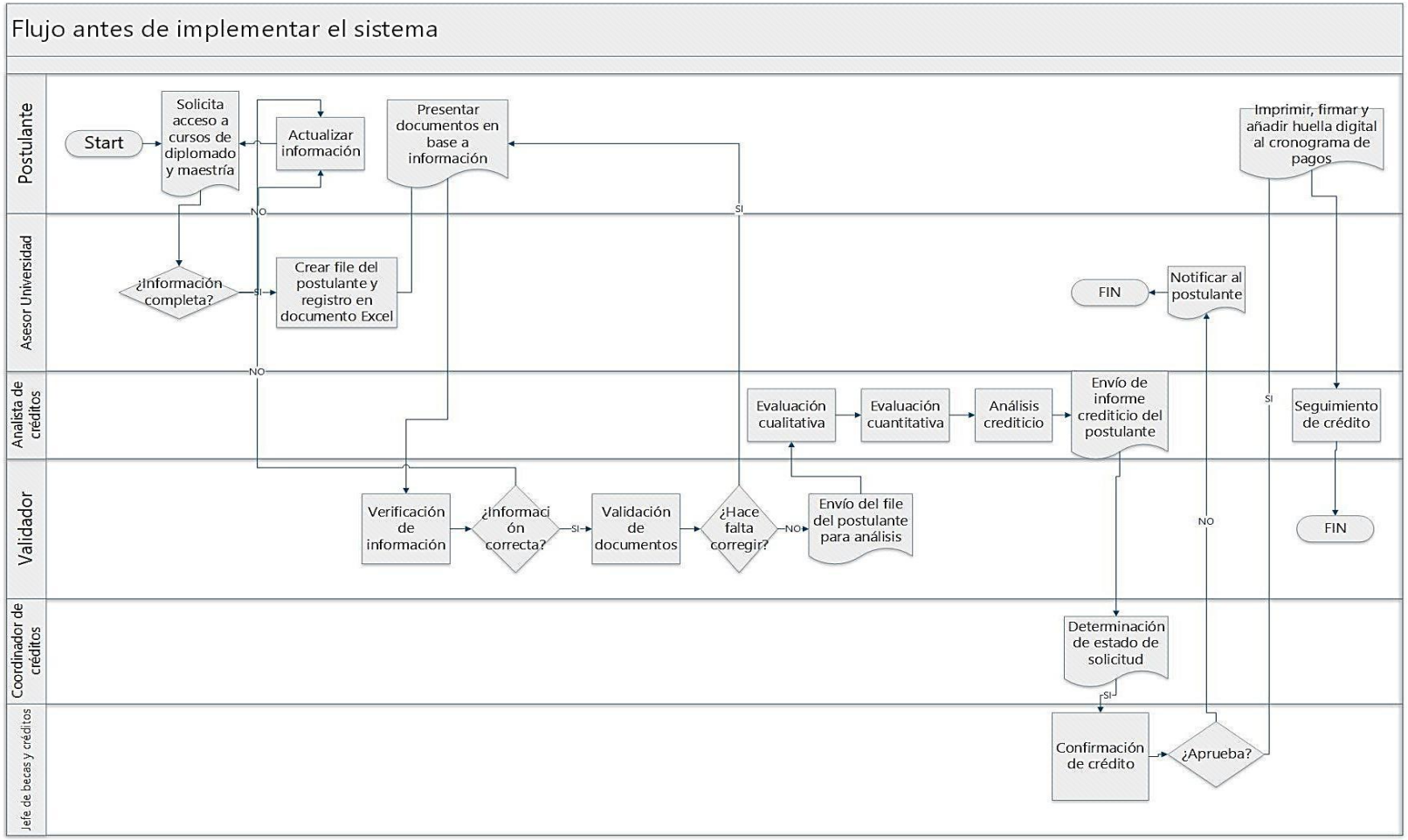


Fuente: ONG de estudio

2.1.6. MAPA DEL PROCESO PREVIO

La Figura 2 muestra el mapa del proceso previo de la organización, destacándose el uso de documentos, que suelen ser enviados por correo electrónico sin ser centralizados, derivando en múltiples versiones de la misma información, correos perdidos y retrasos en respuestas.

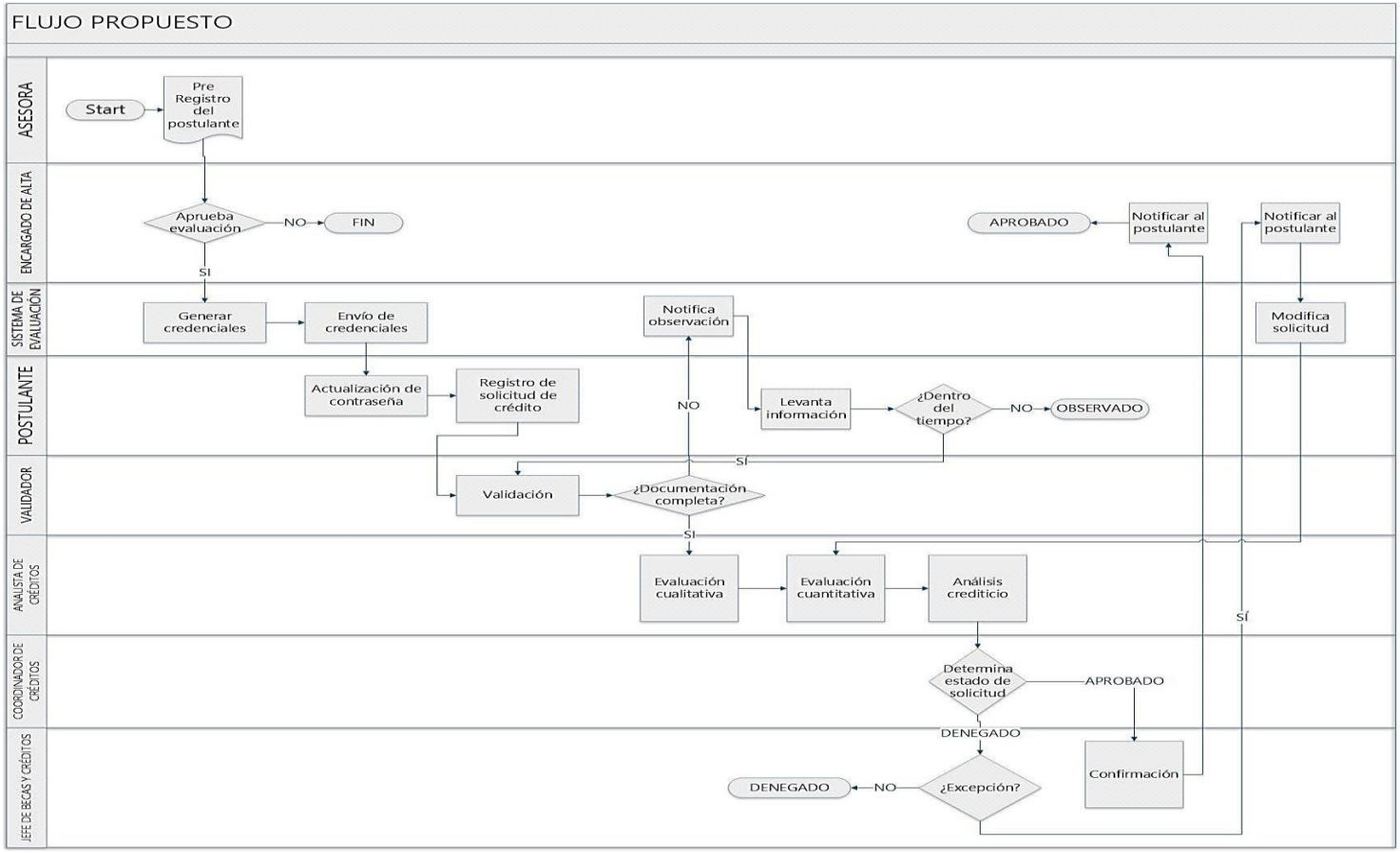
Figura 2: Mapa del proceso actual



Fuente: Elaboración propia

2.1.7. MAPA DEL PROCESO PROPUESTO

Figura 3: Mapa de proceso propuesto



Fuente: ONG de estudio

Como se muestra en la Figura 3, el sistema contemplará tres procesos principales:

- Registro de usuario
- Registro de solicitud
- Evaluación

En cuanto a los actores involucrados dentro del proceso, tenemos a los siguientes:

- Asesor (Universidad)
- Encargado de alta de usuarios (ONG)
- Sistema de Evaluación Crediticia
- Postulante
- Validador (ONG)
- Analista de créditos (ONG)
- Coordinador de créditos (ONG)
- Jefe de becas y créditos (Universidad)

El asesor de la universidad realiza un pre registro de los postulantes para un programa específico. El encargado de alta de usuarios de la ONG, en revisión de los documentos que tiene disponibles, y en coordinación con la universidad, realiza la aprobación de los postulantes cargados por el asesor a través del sistema, el cual genera las credenciales de acceso para los postulantes aprobados y se las envía por correo electrónico y un mensaje de texto dándole la bienvenida, invitando a realizar la carga de su información.

En esta etapa del proceso, el postulante hace su primer ingreso y el sistema le solicita el cambio de contraseña. Una vez realizado el cambio de contraseña, se le manda a la pantalla en la que debe realizar la carga de su información y su carga documentaria. Con el registro de la información completado, un validador de la ONG

recibe la solicitud para realizar la validación de la información y de los documentos cargados por el postulante.

En este punto existen dos casuísticas: en caso de que la información no esté completa, el validador realiza una notificación al postulante mediante el sistema, estableciendo comunicación hasta por tres ocasiones, correspondiendo al postulante levantar las observaciones dentro del plazo previsto, caso contrario se observará la solicitud y no podrá acceder al programa elegido; por otro lado, si la documentación está completa y no existen observaciones, se procede con la evaluación cualitativa del postulante. Las evaluaciones cualitativas y cuantitativas del postulante obedecen a procedimientos internos de la ONG y su análisis no está contemplado por el sistema, siendo responsabilidad del analista de créditos de la ONG indicar si la solicitud procedería o no, todo mediante un informe en el que se especifiquen las observaciones correspondientes.

El coordinador de créditos de la ONG recibe los informes y ejecuta la aprobación o el rechazo de la solicitud. En este punto también existen dos casuísticas: en caso de que la solicitud sea aprobada, el jefe de becas y créditos de la universidad realiza la confirmación y el sistema notifica al postulante mediante correo electrónico y mensaje de texto, finalizando con el proceso. En caso de que la solicitud sea rechazada, es responsabilidad del jefe de becas y créditos de la universidad determinar si la solicitud del postulante pasará por excepción o no. De pasar por excepción, el sistema notifica al postulante para que realice una modificación en su solicitud de crédito e inicia un nuevo proceso de evaluación cuantitativa; caso contrario, la solicitud es rechazada.

2.2. BASES TEÓRICAS

2.2.1. SISTEMAS DE INFORMACIÓN

Rafael Lapiedra (4) hace una distinción importante entre los sistemas informáticos y los sistemas de información. En líneas generales, describe al sistema informático como todo aquel sistema elaborado a base de hardware y software, que recibe un input y devuelve un output determinado. Por el otro lado, reconoce al sistema de

información como aquel que involucra a los recursos humanos, los procesos y los recursos de telecomunicaciones para dar valor al momento de realizar acciones de toma de decisiones.

Asimismo, recalca el papel fundamental de los sistemas de información en organizaciones y empresas de todo tamaño, siendo ineficiente una toma de decisiones sin tener el respaldo de un sistema holístico o, al menos, involucrado con las áreas relacionadas a la decisión a tomar; siendo esta una ventaja competitiva bastante importante.

Según Lapiedra, el papel fundamental de los sistemas de información es el de proporcionar acceso al personal autorizado a la información necesaria, en el momento apropiado y respaldado por una estructura acorde a esas necesidades. Esta entrega tiene cumplir con uno de los siguientes fines: toma de decisiones, control estratégico o puesta en práctica de decisiones adoptadas.

A. CADENA DE VALOR

La cadena de valor se entiende como el conjunto de actividades que se deben realizar para poder ofrecer un servicio o un producto determinado. Estas cadenas poseen dos tipos de actividades:

- Actividades primarias: directamente relacionadas con el proceso de creación.
- Actividades de apoyo: que brindan facilidades, a nivel de infraestructura o inputs
- a las actividades primarias para el desarrollo del valor.

Figura 4: Modelo de cadena de valor

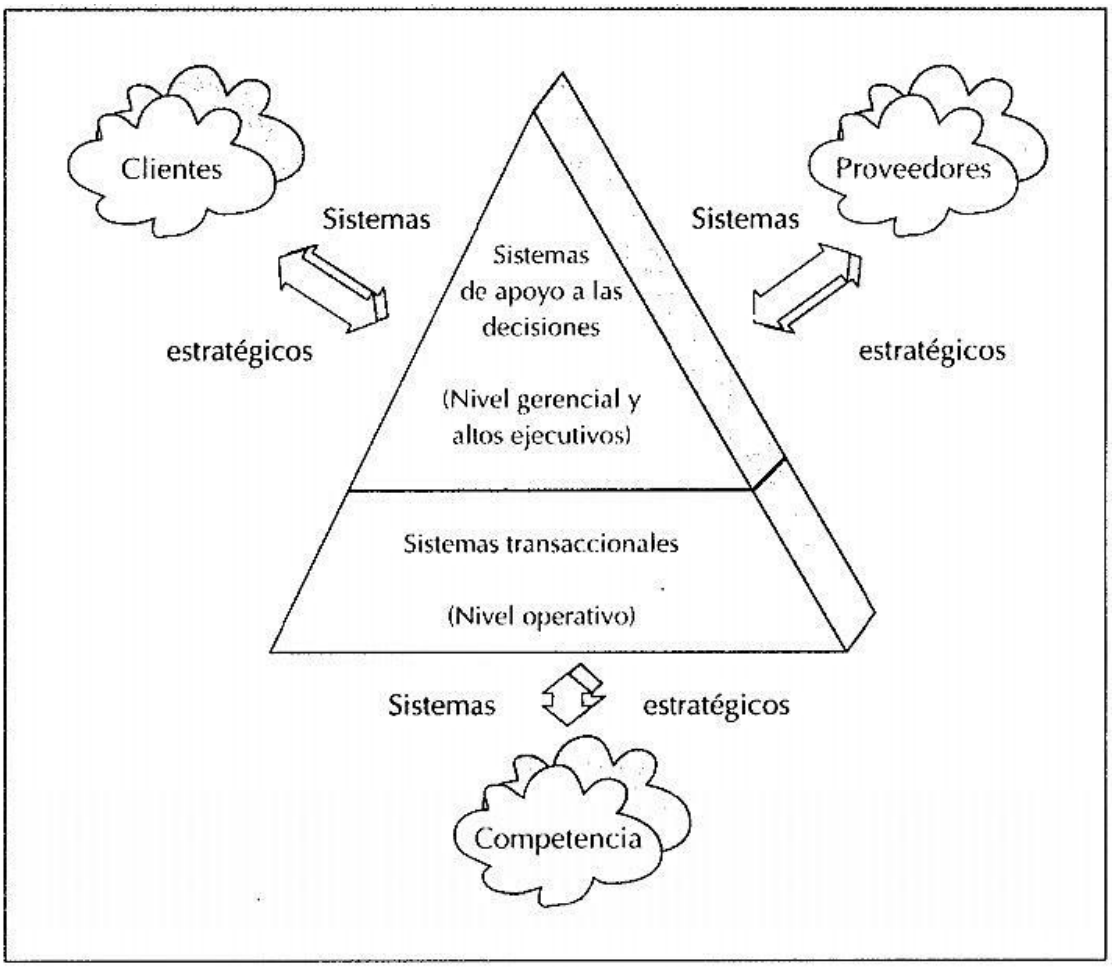
Fuente: Lapiedra, 2011: 19

En la Figura 4, se muestran las actividades divididas, siendo las primarias aquellas agrupadas en la última fila, y las de apoyo aquellas ubicadas en las filas superiores. Es importante destacar este punto, debido a que los sistemas de información, por definición, forman parte de la infraestructura de la empresa, siendo inevitable que todas las actividades necesiten apoyo de estos sistemas para la realización de alguna de sus funciones.

B. TIPOS DE SISTEMAS DE INFORMACIÓN

Según Daniel Cohen (5), los sistemas se pueden agrupar en tres tipos, dependiendo de la función que desempeñen dentro de los procesos de la organización (ver Figura 5): sistemas transaccionales, sistemas de apoyo a las decisiones y sistemas estratégicos.

Figura 5: Tipos de sistemas de información.



Fuente: Cohen, 2000: 9

Sistemas transaccionales: son aquellos que buscan lograr ahorros a nivel de mano de obra, ya que tienen como fin principal el buscar automatizar labores operativas. Una de sus principales características es que suele ser el tipo de sistema que se implanta de entrada en la mayoría de organizaciones, evolucionando a sistemas más complejos, que pueden llegar a brindar mecanismos de control y herramientas para toma de decisiones. Esta facilidad de decisión para implementarlos se da debido a que los resultados de su implementación son rápidos y visibles con facilidad, siendo posible realizar una evaluación del impacto a nivel de ingresos y costos.

Estos sistemas suelen tener funcionalidad simple y repetitiva y generan información en la medida que consumen datos de entrada. Además, funcionan como puerta de entrada de información relevante y mayoritaria en las organizaciones, la cual continúa usándose en las actividades de apoyo (puestos gerenciales).

Como ejemplo de estos sistemas, tenemos aquellos destinados a tareas de facturación, contabilidad, inventarios, etc.

Sistemas de apoyo a las decisiones: son sistemas que brindan las herramientas necesarias para poder realizar una toma de decisiones correcta, tanto a la alta administración como a los puestos de mando intermedios. Por esta razón es que suelen ser implementados luego de los sistemas transaccionales más importantes para las empresas; además, no centran sus funciones en la creación de interfaces para entradas y salidas de datos, sino en el cálculo intenso en base a la información que se maneja a través de los otros sistemas.

Cabe destacar que estos sistemas suelen ser desarrollados poniendo especial atención a brindar una interfaz gráfica amigable, ya que la información procesada debe ser presentada a usuarios finales de la forma más amena, clara y directa posible. Pueden ser clasificados en:

- Decision Support System (DSS), que son sistemas de apoyo para toma de decisiones.
- Group Decision Support System (GDSS), que son sistemas de apoyo a la toma de decisiones grupales.
- Executive Information Systems (EIS), que son sistemas de información para ejecutivos.
- Expert Decision Support System (EDSS), que son sistemas expertos para el apoyo a la toma de decisiones.

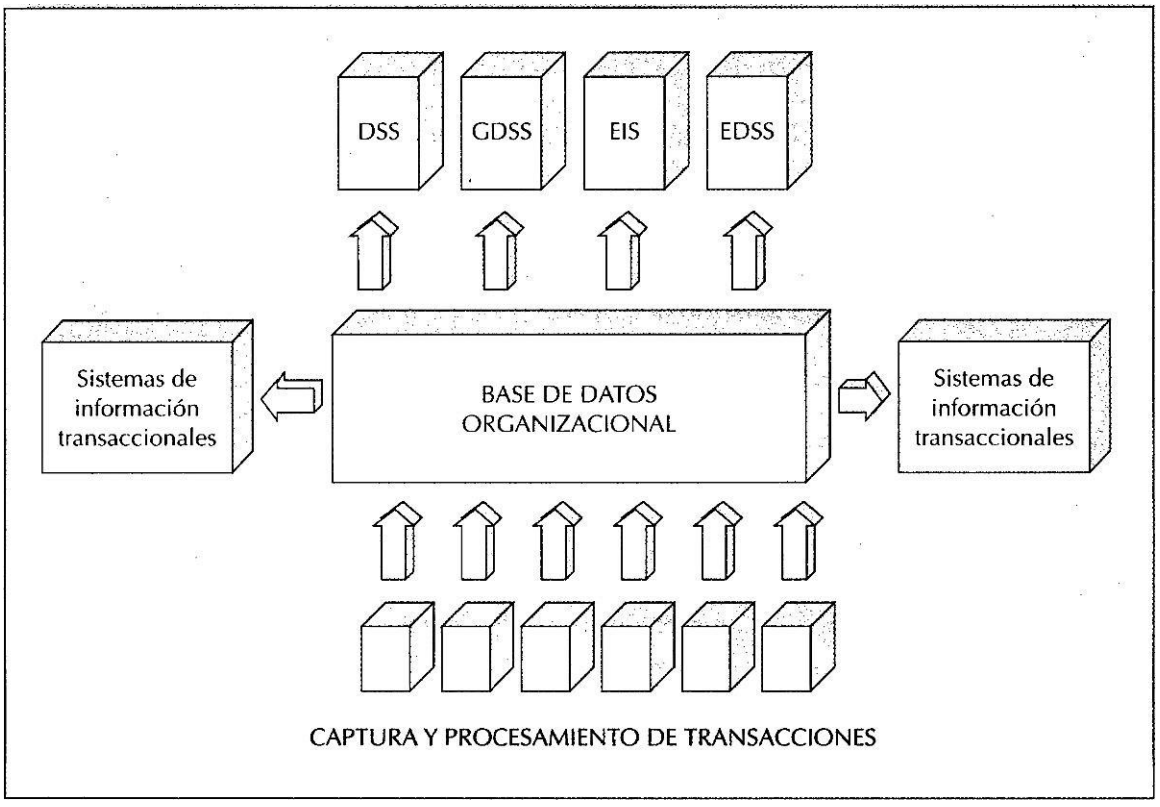
Sistemas estratégicos: estos sistemas no brindan apoyo a la toma de decisiones, ni suelen estar orientados a la automatización de procesos, aunque pueden cumplir esa función en algunos casos. Su orientación principal es brindar a las organizaciones puntos clave en cuanto a ventajas competitivas en su sector. Por esta razón, suelen

ser sistemas desarrollados de forma interna, in house, ya que no suelen adaptarse de forma sencilla a otros sistemas o procesos externos.

Usualmente, dan inicio cubriendo funciones simples, hasta convertirse en sistemas integrales para segmentos específicos de la organización a medida que se integran nuevas funciones o procesos.

Estos sistemas no brindan una ventaja permanente para las organizaciones, ya que, en el mundo empresarial, la competencia tratará de cubrir las ventajas que poseen sus competidores imitando o mejorando sus procesos, llegando a convertirse (en muchos casos) en estándares en la industria.

Figura 6: Clasificación de los sistemas de información



Fuente: Cohen, 2000: 12

2.2.2. SISTEMAS DE INFORMACIÓN ESTRATÉGICOS Y COMPETITIVIDAD EMPRESARIAL

Federico Plancarte (6) recoge un interesante análisis sobre los sistemas de información estratégicos (SIS, por sus siglas en inglés). Citando textualmente: “Un SIS se caracteriza por su habilidad para cambiar significativamente la manera de dirigir un negocio para dar ventaja estratégica (VE) a la empresa. Cualquier Sistema de Información (SI) –EIS, OIS, TPS, KMS- que cambie los objetivos, los procesos, los productos, o las relaciones ambientales para ayudar a una organización a ganar ventaja competitiva (VC) o reducir una desventaja competitiva (DC) es un Sistema de Información estratégica.”.

Es claro que las organizaciones buscan una ventaja competitiva cuando establecen sus estrategias, sean estas a nivel de velocidad, calidad o costos. Bajo este enfoque, los sistemas estratégicos buscan aumentar esta ventaja competitiva mediante el fortalecimiento de sus objetivos estratégicos.

Los cambios violentos que se dan en el sector tecnológico y en el ambiente de negocios pueden afectar de forma positiva o negativa en la implementación de un sistema de información estratégico. Siendo esto así: “Cuando el SIS tiene éxito, puede causar enormes ventajas y ganancias. Pero cuando falla, el costo puede ser sumamente alto. En algunos casos, el fracaso del SIS puede ser tan alto que puede llevar a una empresa a la quiebra como resultado.” [Cita textual] (6)

Para realizar una identificación adecuada de las necesidades inmediatas a nivel estratégico que conduzcan a la implementación de sistemas de esta naturaleza, es necesario detectar los problemas conocidos y las áreas o los grupos de trabajo en las que la implementación de estos sistemas pueda generar un impacto positivo a nivel estratégico, tomar la decisión correspondiente y generar un soporte apropiado en TI. Luego, desplegar tecnologías disponibles, procurando concordar estas tecnologías con los procesos y los modelos usados por la organización.

2.2.3. CLOUD COMPUTING Y ESCALABILIDAD

Peter Mell y Tim Grance (7), indicaron la definición formal que le da el NIST (National Institute of Standards and Technology), la cual se traduce en: “La computación en la nube es un modelo que permite acceso de red ubicuo, conveniente y bajo demanda a un grupo compartido de recursos informáticos configurables (por ejemplo, redes, servidores, almacenamiento, aplicaciones y servicios) que pueden aprovisionarse y lanzarse rápidamente con un mínimo esfuerzo administrativo o la interacción del proveedor de servicios. Este modelo de nube se compone de cinco características esenciales, tres modelos de servicio y cuatro modelos de implementación.”

Las cinco características esenciales identificadas por el NIST, son:

- Auto servicio bajo demanda, la cual se refiere a la capacidad del consumidor del servicio de extender su consumo de recursos informáticos de forma unilateral, según lo necesite y de forma automática, sin la intervención humana directa por parte de los proveedores de servicios.
- Acceso amplio a la red, que se refiere a la disponibilidad de las capacidades que ofrece la nube para que puedan ser aprovechadas con el uso de mecanismos estandarizados, incentivando su consumo desde plataformas heterogéneas.
- Agrupación de recursos, se refiere a la distribución de los recursos mediante agrupación, lo cual permite brindar un servicio a múltiples consumidores mediante la implementación de un modelo multi-tenant. Estos recursos se deben poder asignar de forma dinámica basándose en la demanda de los consumidores.
- Elasticidad rápida, la cual hace referencia a la flexibilidad en los procesos de aprovisionamiento y liberación de recursos, la cual puede darse de forma manual o automática, dependiendo del contexto. Esto de cara al consumidor, se traduce en la aparente carencia de limitaciones y asignaciones de cualquier cantidad de recursos.
- Servicio medido, que es la capacidad del servicio de brindar información precisa y transparente sobre el uso de los recursos y su consumo. Esto se debe brindar

con la finalidad de otorgar herramientas fiables de monitoreo, control e información, tanto por el lado del proveedor, como del consumidor. Asimismo, el servicio debe, por sí mismo, contar con controles automáticos para la optimización del uso de los recursos para los consumidores.

A. MODELOS DE SERVICIO

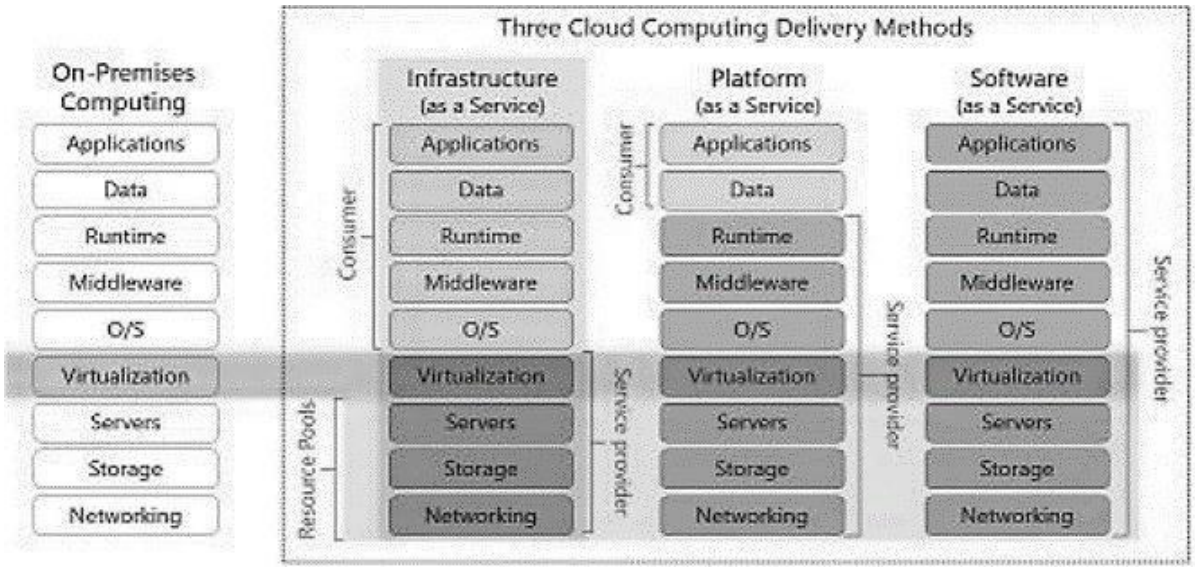
Dentro de los modelos de servicio, la computación en la nube ofrece:

1. Software as a Service (SaaS). Mediante este modelo de servicio, el cliente puede consumir las aplicaciones desplegadas por un proveedor en una infraestructura de nube. Estas aplicaciones deberían ser accesibles desde diversos tipos de dispositivos, usualmente mediante una interfaz gráfica. En este modelo de servicio, el usuario no posee control más allá de las configuraciones de su cuenta en la aplicación, sin conocer ni tener influencia sobre la infraestructura de la nube en la que se encuentra desplegada la aplicación.

2. Platform as a Service (PaaS). En este modelo de servicio, al cliente se le permite desplegar aplicaciones y servicios desarrollados por él mismo o adquiridos a terceros a los recursos preparados en la infraestructura de nube del proveedor. Estas aplicaciones deben estar desarrolladas en lenguajes, frameworks, herramientas y servicios que se encuentren soportados por el proveedor. Más allá de contar con los recursos para el despliegue de sus soluciones, el cliente no posee control alguno sobre la configuración de la infraestructura de nube que soporta estos recursos.

3. Infrastructure as a Service (IaaS). A nivel de infraestructura, el servicio brinda a los consumidores la posibilidad de desplegar su propia infraestructura y sus recursos de cómputo (máquinas virtuales, networking, almacenamiento, etc.) para preparar la arquitectura necesaria para soportar el despliegue de sus sistemas. El control que poseen sobre los recursos es superior, pero siguen sin tener control sobre la infraestructura de la nube que soporta el servicio del proveedor.

Figura 7: Modelos de servicio de Computación en la Nube



Fuente: Shah, Manan; Dhiman, Charusmita, 2015: 122 (8)

En la Figura 7, se muestra el detalle de las capacidades otorgadas a los consumidores de estos servicios, comparando los tres modelos de servicio antes mencionados: SaaS, PaaS e IaaS.

B. MODELOS DE DESPLIEGUE

La Computación en la Nube ofrece hasta cuatro modelos de despliegue diferentes, cada uno enfocado a diferentes necesidades. Estos son: Nube Privada (*Private Cloud*), Nube Comunitaria (*Community Cloud*), Nube Pública (*Public Cloud*) y Nube Híbrida (*Hybrid Cloud*), los cuales se describen de la siguiente manera:

1. **Nube Privada.** Aquellas en las que la infraestructura es otorgada a una organización de forma aislada para sus grupos de recursos, asegurando múltiples unidades de negocio. Este modelo de despliegue puede ser administrado y operado por la organización, por terceros o una combinación de ambos. Es usual que existan únicamente bajo un modelo Off-Premise, aunque pueden existir excepciones a este patrón.

2. **Nube Comunitaria.** La infraestructura de esta nube se otorga a un grupo específico de consumidores con objetivos, intereses y preocupaciones comunes. La administración de los recursos otorgados a este modelo de despliegue se da, usualmente, en coordinación de actividades entre las organizaciones de la comunidad.

3. **Nube Pública.** La infraestructura de nube se provee para uso público libre. Pueden ser administradas y operadas por empresas, personas independientes u organizaciones gubernamentales.

4. **Nube Híbrida.** La infraestructura para este modelo es una combinación de dos o más modelos de despliegue, que pueden variar entre el privado, público y el comunitario. Cuentan con tecnología estándar que admite la comunicación y portabilidad de los datos y aplicaciones.

Tabla 3: Comparación entre los diferentes modelos de despliegue Cloud

Las características de cada uno se detallan en la tabla:**Atributo**	**Pública**	**Privada**	**Híbrida**	**Community**
Costo de construir el datacenter en el servicio	No hay costos iniciales, pero es más caro a largo plazo.	Costo inicial alto, pero mínimo a largo plazo.	Costo inicial medio.	Depende del número de cooperadores
Virtualización	Beneficios de eficiencia a través de la virtualización del servidor.	Beneficios de eficiencia a través de la virtualización del servidor.	Beneficios de eficiencia a través de la virtualización del servidor.	Ganancias de eficiencia mediante virtualización de servidores.
Control y flexibilidad de infraestructura	Configuración limitada	Completa en hardware y software	Completa en la parte privada, limitada en la pública.	Alta, pero limitada por las políticas de la comunidad.

Accesibilidad	Accesibilidad extendida	Accesibilidad limitada	Accesibilidad media	Depende del número de cooperativas
Costos de operación y mantenimiento	Bajo	Alto	Medio	Alto
Espacio	No requiere espacio dedicado para el datacenter	Requiere un amplio espacio dedicado para el datacenter	Requiere un espacio promedio dedicado para el datacenter	El espacio requerido depende del número de cooperativas dedicadas al datacenter
Carga de trabajo	Adecuado para manejar picos altos de carga de trabajo	Inadecuado para manejar picos altos de carga de trabajo	Adecuado para manejar picos altos de carga de trabajo	Adecuado para manejar picos altos de carga de trabajo
Localización de la infraestructura	Off-premise	On-premise	Ambos, Off-premise y On-premise	Dentro de las facilidades de las cooperativas
Dueño de la infraestructura	Proveedor	Consumidor	El proveedor es dueño de la parte pública y el consumidor lo es de la parte privada	Compartido entre las cooperativas

Fuentes:Oqail Ahmad, Mohammad; Zaman Khan, Rafiqul, 2015: 4072 (9)

Hamdaqa, Mohammad; Tahvildari, Ladan, 2012: 41-85 (10)

C. ESCALABILIDAD

Por su parte, Kriushanth, M., Justy Mirobi, G. y Arockiam, L. (11) definen la Computación en la Nube como una tecnología que provee recursos informáticos desde centros de datos extensos, como el paradigma para el intercambio de información y capacidad de computación sostenido en una red escalable de nodos. Destacan las facilidades del pago en marcha, los beneficios de poder recurrir a su uso sin realizar una fuerte inversión de capital a nivel de infraestructura y centran la publicación a la escalabilidad automática que brindan los proveedores de servicios de

nube como factor decisivo al momento de optar por infraestructuras que permitan generar un mayor valor competitivo.

A nivel de escalabilidad de nube, existen dos dimensiones:

La escalabilidad horizontal es la capacidad de ampliar la cantidad de servidores o unidades de trabajo (a nivel de software o hardware) para que cumplan las funciones como una unidad lógica. También es conocida como “scaling-out”.

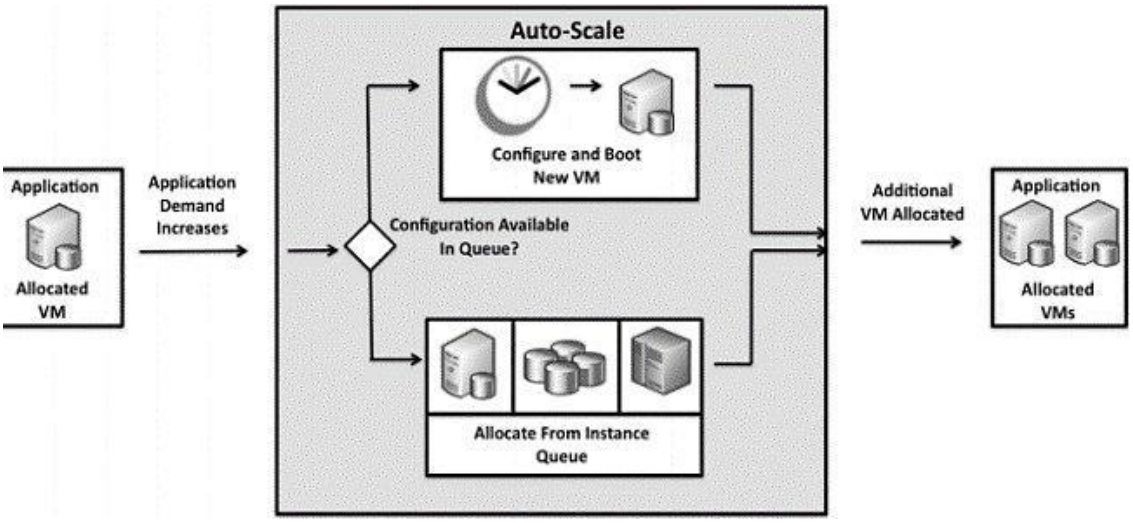
La escalabilidad vertical es la capacidad de expandir la capacidad de los recursos de hardware existentes añadiendo más recursos. Un claro ejemplo es el incremento del disco duro para aumentar la capacidad de almacenamiento. Este tipo de escalabilidad le brinda al sistema y a las aplicaciones acceso a más recursos. También es conocida como “scaling-up”.

D. ESCALADO AUTOMÁTICO

El escalado automático está referido a la capacidad de los recursos de realizar un escalamiento vertical u horizontal, incrementando o decrementando su capacidad y la cantidad de recursos en base a ciertas condiciones pre definidas por el consumidor. Esto permite asegurar que las instancias de las soluciones desplegadas se verán incrementadas en número o capacidad para atender picos altos de demanda de sus usuarios; garantizando, también, que se vean reducidas en cantidad o potencia cuando exista una demanda menor, ofreciendo un mejor control de los costos.

La figura Z muestra el flujo de escalamiento automático para servicios en la nube.

Figura 8: Escalado Automático en una Infraestructura de Nube



Fuente: Kriushanth, M., Justy Mirobi, G. y Arockiam, L., 2013: 2872 (11)

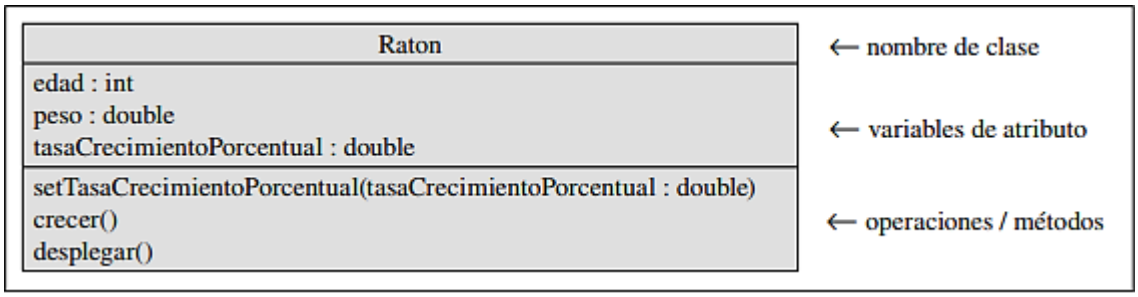
2.2.4. PROGRAMACIÓN ORIENTADA A OBJETOS (POO)

Para poder definir con precisión el concepto de Programación Orientada a Objetos, es necesario definir primero lo que es un objeto.

Según John S. Dean y Raymond H. Dean (12), para definirlo es necesario conocer lo que se quiere representar y modelar en el programa, sean entidades físicas o existan en un plano conceptual. Dado este paso, es necesario identificar sus propiedades y determinar los comportamientos que tienen estas entidades. El aglomerado de estas propiedades y comportamientos es lo que definimos como objeto.

Dicho de otro modo, un objeto es una aglomeración de datos que guardan relación y permiten identificar las características que definen al objeto, sumada al aglomerado de sus comportamientos. Se debe tener en cuenta que los objetos pueden cambiar sus los valores de sus estados o propiedades mediante sus comportamientos. Por ende, en POO, los objetos representan sus estados o propiedades a través de "datos", haciendo lo propio con los comportamientos a través de "métodos".

Figura 9: Representación de una clase abreviada en UML.



Fuente: John S. Dean y Raymond H. Dean, 2009: 180 (12)

En la Figura 9, se representa a la clase “Ratón”, la cual posee como datos a los campos edad (de tipo entero), peso (de tipo double) y tasa Crecimiento Porcentual (de tipo double). Asimismo, se muestran tres métodos para la misma clase, los cuales son set Tasa Crecimiento Porcentual (que recibe una variable de tipo double), crecer y desplegar.

Habiendo dicho esto, podemos hacer una referencia a Olinda Velarde de Barraza, Mitzi Murillo de Velásquez, Ludia Gómez de Meléndez y Felícita Castillo de Krol (13) para definir a la POO como la técnica de programación que usa objetos, como los definidos anteriormente, como bloques de construcción.

Para que se esté realizando el uso real de un paradigma de desarrollo de software en base a la Programación Orientada a Objetos, es necesario que se cumplan ciertas características:

a) ABSTRACCIÓN

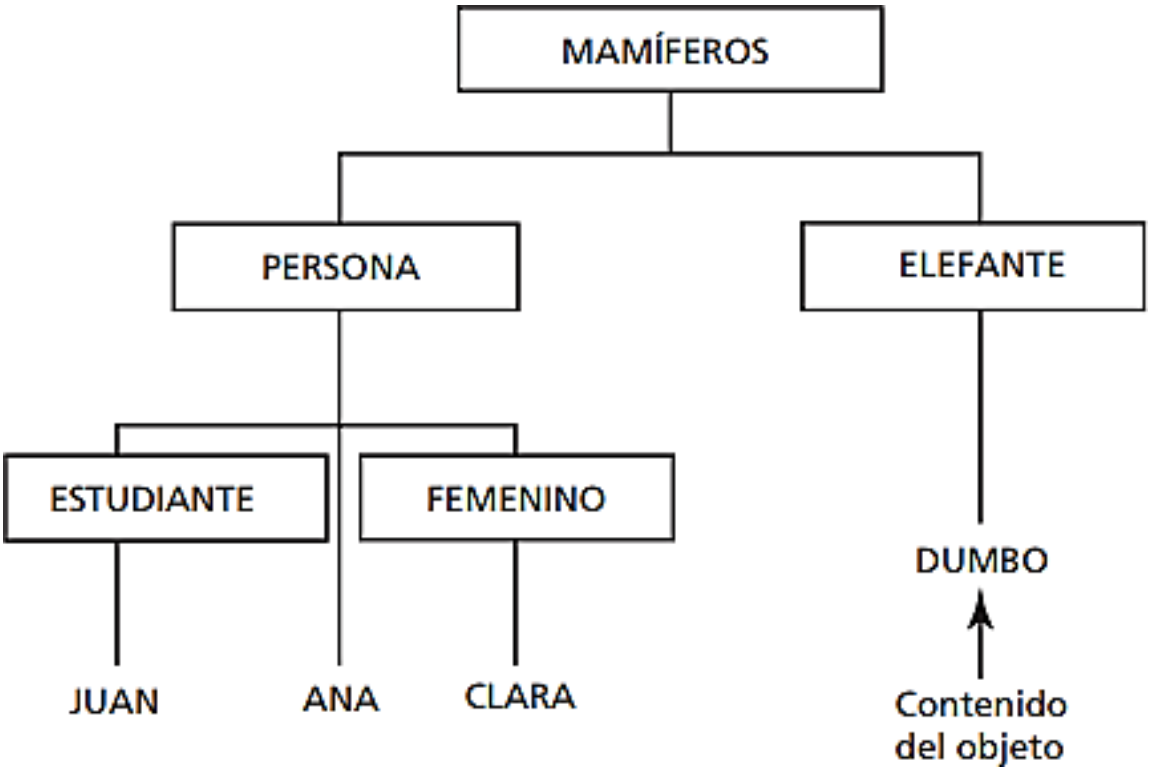
La abstracción se define como la capacidad de crear nuevos tipos de datos, los cuales han de ser definidos por el desarrollador. Es aquí donde entra la estructura llamada clase, la cual está formada por los datos y las funciones de un objeto.

Esta característica suele ir muy ligada al encapsulamiento, el cual permite ocultar los detalles de implementación de las clases y sus propiedades privadas, exponiendo únicamente las interfaces o las propiedades y métodos públicos protegidos, que respeten el funcionamiento interno de la clase y oculten la información que en ella se maneja de los objetos y clases que no tengan el permiso adecuado.

b) HERENCIA

La herencia es la capacidad de dar origen a clases de objetos nuevas, basadas en las clases de objetos ya existentes. Dicho esto, se puede comprender que la herencia brinda la posibilidad de “heredar” los métodos y los datos desde una clase existente, con la posibilidad de extender las mismas en una nueva clase.

Figura 10: Herencia



Fuente: Olinda Velarde de Barraza, Mitzi Murillo de Velásquez, Ludia Gómez de Meléndez y Felícita Castillo de Krol, 2006: 12 (13)

c) POLIMORFISMO

El polimorfismo hace referencia a la capacidad de los objetos para responder de formas distintas al mismo mensaje. Esto se traduce en la posibilidad de realizar diferentes implementaciones de un mismo método, lo cual se consigue manteniendo el nombre del método, siendo este su medio de acceso, y variando los argumentos

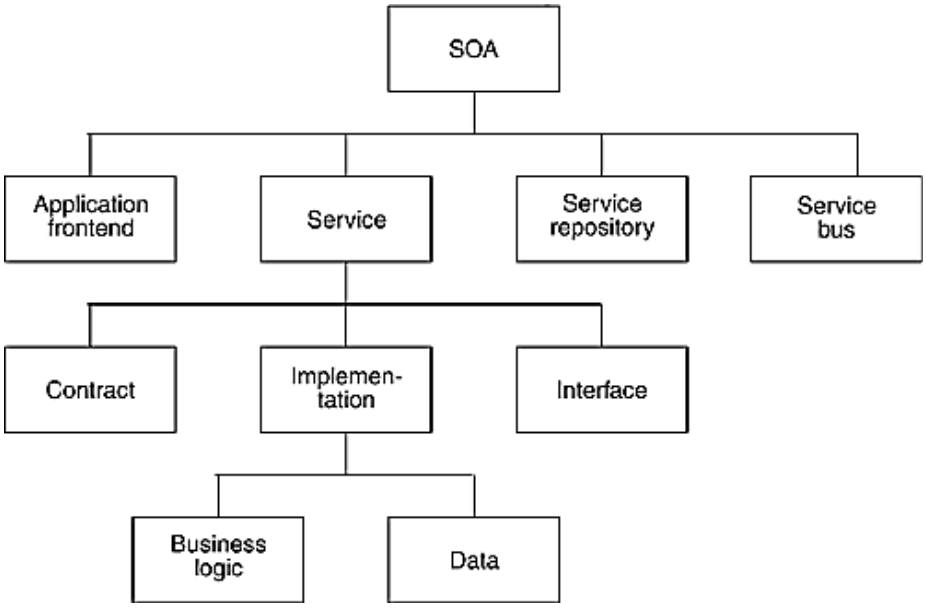
enviados a los métodos. Esta característica está íntimamente relacionada con la herencia.

2.2.5. SERVICE ORIENTED ARCHITECTURE (SOA)

SOA son las siglas empleadas para referirse a la Arquitectura Orientada al Servicio (Service- Oriented Architecture). SOA se define, según Dirk Krafzig, Karl Banke, Dirk Slama (14), como un paradigma para la arquitectura de software basado en ciertos conceptos clave, como los de capa de presentación, servicio, repositorio de servicio, entre otros. Un servicio consiste en un contrato, una o más instancias del servicio y su respectiva implementación.

Cuando se piensa en un servicio a nivel de arquitectura, no debemos confundirlo con las operaciones independientes que posee el negocio, sino entenderlo como el conjunto de operaciones destinadas a cubrir un bloque del modelo de negocio. Para lograr esto, SOA sugiere desacoplar los componentes de las aplicaciones. Es a través de este desacoplamiento que se permita la reusabilidad de diversos componentes, enfocándose principalmente en la capa de servicio, que proveerá de los beneficios del mismo a través de la capa de presentación que se elabore para los usuarios. Debe entenderse, entonces, que la capa de servicio es agnóstica de la capa de presentación, ya que pueden existir diversas aplicaciones que consuman los datos y los procesos provistos por los servicios SOA.

Figura 11: Elementos de SOA

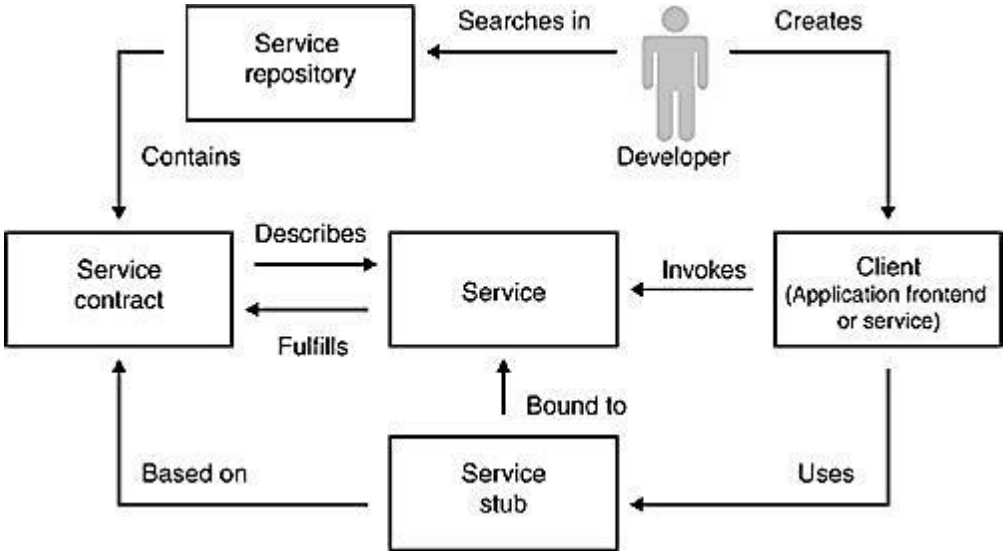


Fuente: Dirk Krafzig, Karl Banke, Dirk Slama, 2005 (14)

Un punto importante a destacar es que, por lo general, se consideran los siguientes como los elementos de SOA:

- Frontend: es el elemento que entrega el valor de los procesos y la información brindada por un servicio SOA a los usuarios finales.
- Servicio: Conjunto de contratos, interfaces, lógica de negocios y data, que busca encapsular un concepto de negocio de alto nivel.
- Repositorio de servicios: elemento que registra los servicios, así como sus atributos para facilitar el consumo y el descubrimiento de los servicios.
- Service Bus: es una infraestructura encargada de brindar cierta funcionalidad de integración entre aplicaciones y servicios, tales como: transformación de protocolos, manejo y entrega de eventos del negocio, seguridad y gestión de comunicación entre servicios.

Figura 12: Flujo de implementación y consumo de SOA



Fuente: Dirk Krafzig, Karl Banke, Dirk Slama, 2005 (14)

En la guía de Patterns & Practices, de Microsoft (15), mencionan que cada aplicación SOA expone funciones del negocio de alto nivel como servicios que serán consumidos por otras aplicaciones. Debido a la complejidad provista por la implementación de las aplicaciones SOA, deben cumplir algunas funciones específicas añadidas:

- Hacer a los servicios localizables en tiempo de ejecución. Esto debido a que existe un alto grado de complejidad al tratar de ubicar servicios en una arquitectura orientada a servicios de nivel empresarial, sobre todo considerando que este tipo de aplicaciones suelen estar distribuidas en diferentes computadoras, redes y demás.
- Hacer que los consumidores y el servicio respeten un formato. Tras localizar el servicio adecuado, la aplicación cliente deberá ser capaz de determinar el protocolo adecuado para acceder al servicio de forma dinámica, cómo dar formato a una solicitud y qué respuestas esperar del servicio.

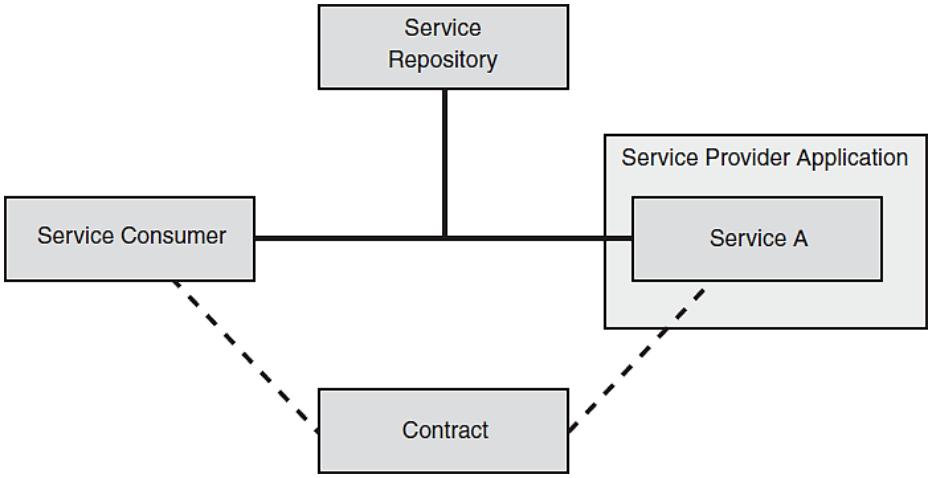
1) CONTRATOS DEL SERVICIO

Es normal que, dentro de las aplicaciones comunes, la firma de un método le brinde al método que hace la llamada un entendimiento claro sobre el “acuerdo” que existe en esa llamada. Por poner un ejemplo, podemos citar los tipos y la cantidad de parámetros, así como el tipo de dato que debe esperar de retorno una vez que la llamada se complete. Esto es posible debido a que estos métodos pueden hacer suposiciones en base a este “entendimiento”, como que ambos se ejecutan en el mismo proceso y comparten el mismo espacio en la memoria del ordenador, que utilizan el mismo lenguaje de programación y el método que ha llamado seguirá ejecutándose una vez que se completen las acciones del método llamado. Sin embargo, dentro de las aplicaciones de tipo SOA distribuidas, la mayoría de estas suposiciones no resultan ser correctas y se utiliza un contrato de servicio para poder establecer estos puntos importantes de comunicación entre los métodos.

Un contrato de servicio es aquel que brinda información concerniente a la implementación del canal de comunicación entre los clientes y las aplicaciones del proveedor de los servicios a ser consumidos (el protocolo de red, por poner un ejemplo). Este contrato también especifica los tipos de mensajes que el servicio podrá procesar o producir, todo especificado en un esquema altamente detallado por cada mensaje.

En SOA, un servicio puede tener varios contratos definidos, esto con el objetivo de brindar acceso a funciones e información sensible para la empresa únicamente a sus usuarios internos, mediante el uso de un contrato no público; al mismo tiempo, para brindar la información no sensitiva y los métodos públicos a organizaciones y usuarios externos a la organización mediante el uso de un contrato público que determinen aquellos métodos expuestos.

Figura 13: Invocación de un servicio en SOA



Fuente: Microsoft Corporation, 2003: 268 (15)

La Figura 13 muestra los elementos y los pasos requeridos en una implementación SOA:

- Descubrimiento. Un consumidor (aplicación cliente) consulta en el repositorio de servicios, quien le otorga la dirección del servicio deseado.
- Negociación. La aplicación cliente y el proveedor del servicio realizan el acuerdo de la implementación de la comunicación mediante el contrato de servicio provisto por el proveedor del mismo.
- Invocación. El cliente invoca al servicio respetando el contrato acordado.

2.3. DEFINICIÓN DE TÉRMINOS BÁSICOS

Para poder comprender el sistema desarrollado en la tesis elaborada en este documento, es necesario conocer ciertos conceptos que enmarcan gran parte del contexto de la misma. En esta sección se explicarán los conceptos principales para su comprensión plena.

1. **Arquitectura de Software.** Es un conjunto de sentencias que establecen los componentes de software para una aplicación específica, o un conjunto de

aplicaciones, y asignan la funcionalidad que tendrán dichos componentes en relación con los demás. (14)

2. **Lenguaje de programación.** Es un conjunto de símbolos y palabras que siguen reglas para la construcción de sentencias que respetan la sintáctica y semántica del mismo, las cuales son entendidas por el ordenador como instrucciones ejecutables. (16)

3. **Blob.** Es un conjunto masivo de datos no estructurados, tal como texto o data binaria, ideal para ofrecer imágenes o documentos, almacenar archivos para acceso distribuido, streaming multimedia, escritura de archivos log, etc. (17)

4. **Container.** Encargado de organizar blobs, tal como hacen las carpetas en un sistema de archivos. Todo blob existe dentro de un container, el cual puede poseer un número ilimitado de blobs. (17)

5. **Azure Storage.** Es la solución de Microsoft Azure para la nube, optimizada para el almacenamiento de Containers de Blobs. Un Azure Storage es capaz de almacenar una cantidad ilimitada de Containers. (17)

6. **WCF. WCF** es el acrónimo de Windows Communication Foundation, el cual es un framework para el desarrollo de aplicaciones bajo el paradigma SOA, cuyas implementaciones proveen estándares requeridos para el intercambio correcto de mensajes entre servicios, denominados endpoints.

7. **XML.** Es el acrónimo de Extensible Markup Language (Lenguaje de Etiquetado Extensible), el cual hace referencia a un formato de texto altamente flexible y simple. Este formato ha sido derivado de SGML (ISO 8879) y se utiliza para almacenar y transportar información. (18)

8. **PaaS.** Es el acrónimo utilizado para Platform as a Service. Es un conjunto de servicios orientados a desarrolladores, que les quita la carga de preocuparse por la infraestructura que soporta a este tipo de servicios, enfocándose en el desarrollo y las pruebas de sus aplicaciones. (7)

CAPÍTULO III
METODOLOGÍA

En este capítulo se dará una descripción detallada de las metodologías empleadas para el desarrollo de la solución planteada, tanto a nivel de gestión de solución, como para el desarrollo mismo de la aplicación.

3.1. METODOLOGÍA APLICADA PARA LA GESTIÓN DE LA SOLUCIÓN

La metodología empleada en el presente proyecto es SCRUM, empleada en la gestión de proyectos de desarrollo de software, que destaca por ser ágil flexible, moderna y ampliamente extendida para el desarrollo de software en entornos corporativos.

SCRUM se define como un marco de trabajo orientado al desarrollo y mantenimiento de productos que poseen un alto grado de complejidad. Este marco de trabajo es aplicado como metodología estandarizada para la gestión de proyectos de software. Además, los artefactos descritos por esta metodología permiten alinear fácilmente la misma con la metodología de gestión de proyectos PMBOK, la cual es más genérica.

SCRUM define tres roles principales:

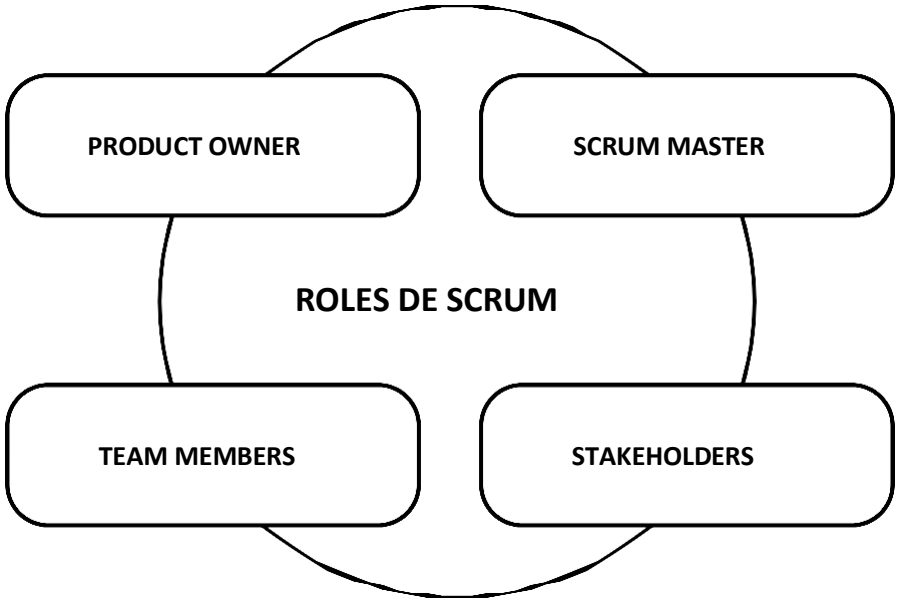
a. Product Owner (Dueño del producto). Es el encargado de la gestión del Product Backlog (lista de producto), con la finalidad de orientar al equipo de desarrollo en la consecución del valor del producto y su trabajo.

b. Scrum Master. Es el encargado de asegurar que la aplicación de la metodología SCRUM se esté aplicando y sea adoptada por el equipo de forma correcta. Asimismo, gestiona las interacciones de agentes externos con los miembros del equipo para evaluar el impacto y el grado de beneficio que pueden otorgar dichas interacciones al proyecto.

c. Team members (Miembros del equipo). Usualmente, dentro de este rol, se considera al equipo de desarrollo. Son los responsables de realizar la entrega incremental del producto con la intención de añadir características al entorno de “Producción” al término de cada Sprint, hasta su finalización.

d. Los Stakeholders engloban a Clientes, Administración y Usuarios, los cuales son roles que pueden ser considerados dentro de la metodología SCRUM.

Figura 14: Roles de SCRUM



Fuente: Elaboración propia

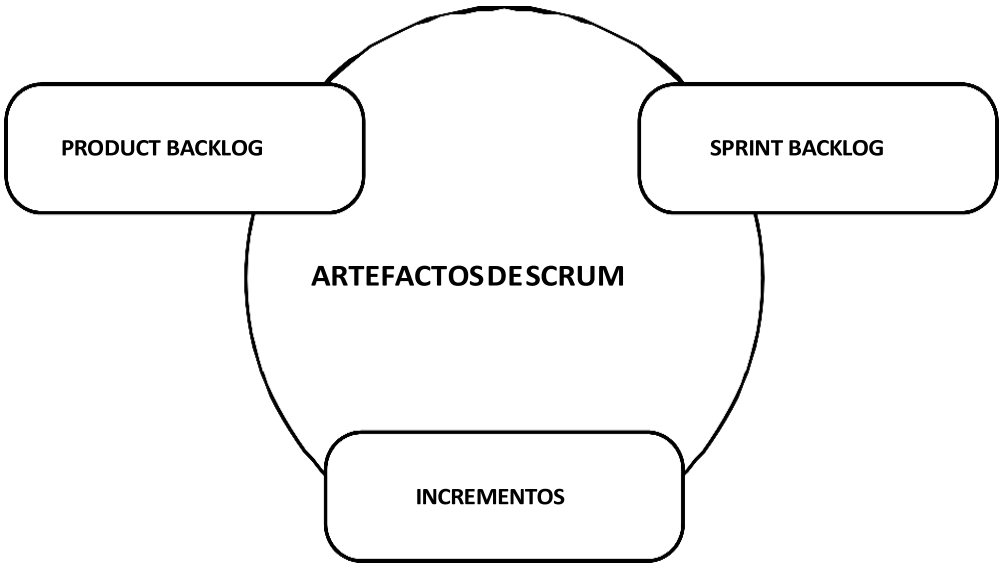
De la misma forma, la metodología SCRUM describe tres artefactos principales:

a. **Product Backlog** (Lista de producto). Es una lista que posee el detalle de las funcionalidades a implementar del sistema, que posee los niveles correspondientes de prioridad para cada funcionalidad. Durante el desarrollo del producto final pueden ser añadidas nuevas funcionalidades a esta lista. Es mantenida de forma directa por el Product Owner.

b. **Sprint Backlog** (Lista de pendientes del Sprint). Es el conjunto de los pendientes, correspondientes al Product Backlog, seleccionados para cada Sprint. Además de la lista de los elementos para ese Sprint, puede ir acompañado del plan para realizar la entrega de los incrementos del producto hasta la finalización del mismo. En este documento no se admiten actividades extensas, pero no hay un límite para la cantidad de las mismas.

c. **Incrementos.** Un incremento es la sumatoria de los elementos del Product Backlog completados durante cada Sprint. A la finalización de un Sprint, este artefacto debe estar "Finalizado", lo cual se traduce en la posibilidad de ser empleado para el despliegue en los entornos correspondientes.

Figura 15: Artefactos de SCRUM



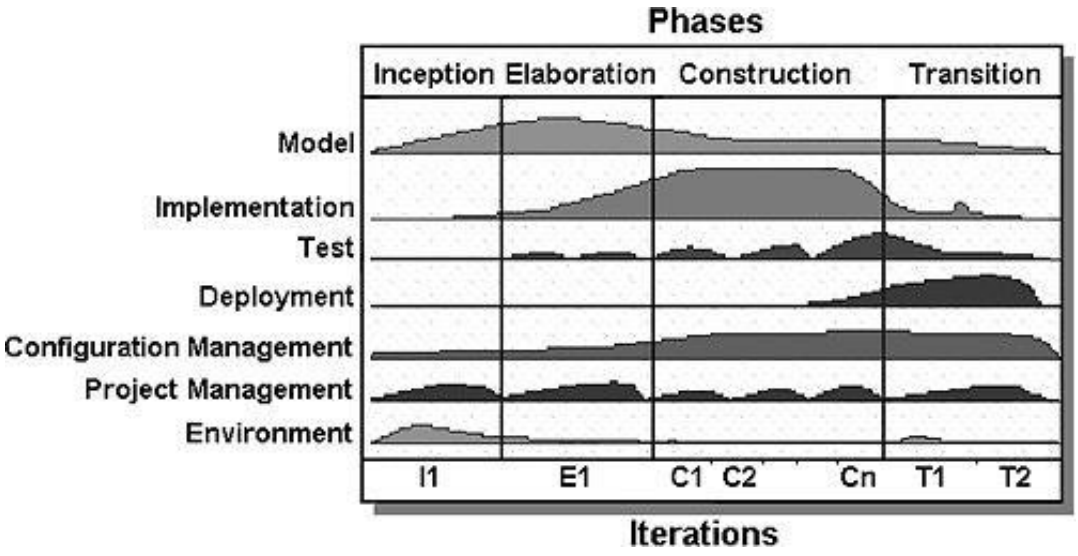
Fuente: Elaboración propia

3.2. METODOLOGÍA AUP APLICADA PARA EL DESARROLLO

La metodología que se empleó en el desarrollo del sistema de información fue una variante de Agile Unified Process (AUP), la cual es, a su vez, una versión minimizada de la metodología Rational Unified Process (RUP), llamada dX, la cual utiliza prácticas de metodologías ágiles de desarrollo similares a Extreme Programming (XP). AUP concentra las etapas de “Modelado de negocio, “Obtención de Requerimientos” y la de “Análisis y diseño” en una sola fase, llamada “Modelo”, manteniendo las de “Implementación”, “Pruebas de calidad” y “Despliegue”.

A continuación, se muestra un gráfico que muestra las fases correspondientes a AUP:

Figura 16: Fases de AUP



Fuente: Scott W. Ambler, 2005 (19)

a. INCEPTION

Al finalizar esta fase, se debería poseer una idea clara y establecida de cuáles son los casos de uso principales para el desarrollo de la solución y los primeros modelos de la arquitectura. Bajo esto en mente, los pasos a seguir serían:

Escribir los Casos de Uso (CU) principales en fichas, en colaboración del jefe de proyecto del cliente y en coordinación directa con los desarrolladores.

Creación de prototipos sencillos, basándose en los CU.

Los prototipos se usan para medir determinar el alcance y el detalle de los CU y para dar inicio a la afinación de las especificaciones de los CU.

Los prototipos se emplean, del mismo modo, para empezar a medir el impacto de ciertas arquitecturas propuestas.

b. ELABORATION

En esta fase se da inicio a la elaboración del diseño y a la programación.

Se finaliza con la elaboración de las fichas de CU. Los desarrolladores del proyecto estiman los esfuerzos a nivel de cargas de horas de trabajo para cada CU.

Es responsabilidad del cliente estimar la prioridad de los CU, considerando a los que posee un riesgo mayor como principales.

Se planifican las diferentes iteraciones correspondientes al proyecto, cada una de las cuales no deberá poseer más de una semana de duración. En este punto, se debe considerar que la suma total de la cantidad estimada de trabajo no supere al total estimado para la iteración. Al finalizar, se divide la cantidad de CU desarrollados entre la cantidad de CU estimada para desarrollo en dicha iteración; esto con el fin de obtener la ratio de factor de carga.

Las iteraciones son diseñadas para no exceder los límites establecidos por la arquitectura.

Al ser una metodología que mantiene un ritmo de modificaciones elevado, brinda alta importancia a la elaboración de pruebas. Por esto, los desarrolladores deben alternar su tiempo entre la elaboración de los casos de pruebas y sus actividades de programación.

El código puede ser modificado de forma directa por cualquier miembro del equipo encargado de desarrollar el sistema, independientemente del módulo afectado.

La integración de los componentes desarrollados se realiza con una frecuencia diaria.

c. CONSTRUCTION

En esta etapa, el equipo genera un cronograma de lanzamientos basándose en el factor de carga obtenido en la fase previa. Cabe resaltar que esta fase le brinda un mayor énfasis a la estabilidad de la arquitectura elegida y al plan del proyecto.

d. TRANSITION

Esta fase, de transición, tiene el fin de servir de retroalimentación, ya que inicia desde el primer lanzamiento hasta la finalización del producto, planificando las iteraciones en plazos cortos. Una vez realizada la primera iteración, se tendrá un sistema quepuede funcionar independientemente de las siguientes iteraciones, sin llegar a ser el producto final, por lo cual puede estar sometido a todo tipo de modificaciones.

CAPÍTULO IV
ANÁLISIS Y DISEÑO DE LA SOLUCIÓN

El capítulo está enfocado en la identificación de los requerimientos funcionales y no funcionales del producto. Asimismo, se detallan los casos de uso derivados de estos requerimientos. Luego, se realiza un análisis completo a la solución propuesta, el cual incluye un estudio técnico y económico. Como parte final del capítulo, se presenta la etapa de diseño, de la cual deriva la arquitectura e infraestructura elegidas, los mockups de la interfaz gráfica propuestos y el diseño lógico de la base de datos.

4.1. IDENTIFICACIÓN DE REQUERIMIENTOS:

A continuación, se definirán los requerimientos obtenidos de la comunicación directa con la ONG para la que se desarrollará la solución, de los cuales se generarán los casos de uso en una sección posterior. Los requerimientos aquí descritos están separados de acuerdo a grupos de actividades, identificándose tres principalmente:

- Gestión general
- Evaluación crediticia
- Administración de la documentación

4.1.1. REQUERIMIENTOS FUNCIONALES

Tabla 4: Requerimientos funcionales de la solución

CÓDIGO	REQUERIMIENTO	DESCRIPCIÓN
RF-01	El sistema controlará los accesos	Se requiere identificar al usuario que ingresa al sistema mediante: • Validación de su existencia en la tabla de usuarios. • Validación de contraseña. • Validación de estado (habilitado – no habilitado) Se requiere, también, que el sistema solicite el cambio de las contraseñas para usuarios que accedan por primera vez al sistema.
RF-02	El sistema permitirá la creación de postulantes	Se requiere habilitar la carga masiva para el registro de postulantes: • Se debe utilizar un formato de carga masiva. • Se debe generar un log de carga masiva. Se requiere también, poder generar nuevos postulantes de forma individual.
RF-03	El sistema realizará envío de alertas	• Se requiere habilitar el envío de notificaciones durante el flujo de evaluación: o Notificación interna tras la asignación de una solicitud de algún postulante. o Notificación externa para la solicitud de documentos a los postulantes. o Notificación externa tras la aceptación o el rechazo de su solicitud. • Notificaciones internas mediante correo electrónico. • Notificaciones externas mediante correo electrónico y mensajes de texto.

RF-04	El sistema debe realizar actividades de verificación de solicitudes	Se requiere que el sistema sea capaz, de forma automática y periódica, de realizar actividades de verificación del estado de las solicitudes, notificando a los postulantes sobre la proximidad de la fecha de vencimiento de sus solicitudes. Asimismo, se requiere que se “eliminen” de forma lógica las solicitudes que superaron su fecha de vencimiento.
RF-05	El sistema debe permitir el envío de mensajes de bienvenida	Se requiere poder enviar, de forma automática, correos y mensajes de texto a los nuevos usuarios registrados, el mensaje debe contener: • Mensaje de bienvenida. • Credenciales (sólo en el caso del correo) • Dirección web de acceso al sistema
RF-06	El sistema debe permitir el registro de planes de postgrado	Se requiere permitir el registro de los planes de postgrado, que son administrados por las instituciones educativas superiores que trabajan con la ONG.
RF-07	El sistema debe permitir la creación de usuarios	Se requiere administrar la carga masiva para el registro de usuarios: • Se debe utilizar un formato de carga masiva. • Se debe generar un log de carga masiva. Se requiere también, poder generar nuevos usuarios de forma individual.
RF-08	El sistema debe facilitar la administración de perfiles y roles para la ONG	Se requiere habilitar la segmentación de perfiles y accesos, de acuerdo a roles para los usuarios de la ONG.
RF-09	El sistema debe permitir el registro de datos maestros	Se requiere permitir el registro de datos maestros para la aplicación, como los datos de uso general durante el proceso de evaluación crediticia.

RF-10	El sistema debe permitir el registro de datos de postulante	Se requiere permitir el registro de los datos generales del postulante, para su posterior evaluación.
RF-11	El sistema debe permitir asignar las solicitudes a los responsables disponibles	Dependiendo de la etapa del proceso de evaluación de la solicitud, se requiere que el sistema permita asignar las solicitudes a los perfiles y responsables disponibles para el proceso de evaluación.
RF-12	El sistema debe generar un log de eventos para auditoría	Se requiere que el sistema guarde un registro de eventos para procesos de auditoría.
RF-13	El sistema deberá permitir habilitar el ingreso de información de ingresos del postulante	Se requiere permitir el ingreso manual de los datos de información sobre los ingresos del postulante: • Ingresos personales. • Ingresos conyugales.
RF-14	El sistema deberá permitir habilitar el ingreso de información de egresos del postulante	Se requiere permitir el ingreso manual de los datos de información sobre los egresos del postulante: • Egresos personales. • Egresos conyugales. • Egresos del cónyuge.
RF-15	El sistema permitirá realizar el registro de comunicaciones	Se requiere permitir el registro de las llamadas de seguimiento con los postulantes para la verificación de documentación e información y las citas programadas.
RF-16	El sistema debe permitir el registro de la información para la evaluación cuantitativa de los postulantes	Se requiere permitir el registro de la información cuantitativa del postulante (capacidad de endeudamiento, capacidad de pago, etc.)
RF-17	El sistema debe permitir el registro de la información para la evaluación cualitativa de los postulantes	Se requiere permitir el registro de la información cualitativa del postulante (calificación en el sistema financiero, información laboral, etc.)

RF-18	El sistema debe permitir la generación del reporte crediticio	Se requiere la generación de un resultado integral de la evaluación financiera para ser presentado a los coordinadores de créditos.
RF-19	El sistema debe realizar la generación del cronograma de pagos.	Se requiere que el sistema realice la generación del cronograma de pagos, posterior a la evaluación crediticia. La generación del cronograma debe realizarse mediante el cálculo de la cantidad de cuotas necesarias para el repago de crédito en el tipo de pago establecido por programa.
RF-20	El sistema debe permitir el registro de la autorización de descuentos	Se requiere la posibilidad de registrar el porcentaje de descuento al que el postulante tiene la capacidad de acceder.
RF-21	El sistema debe permitir aprobar, devolver o rechazar las solicitudes	Se requiere que el sistema otorgue la posibilidad de aceptar, rechazar o devolver a un estado previo las solicitudes enviadas por los postulantes, con una observación sobre la decisión.
RF-22	El sistema deberá permitir la habilitación de la carga documental	Se requiere habilitar la posibilidad de adjuntar los documentos requeridos para la realización de la evaluación crediticia de acuerdo a cada tipo de perfil del postulante en base a la información proporcionada por el mismo.
RF-23	El sistema deberá permitir realizar la verificación de los documentos adjuntos por parte del postulante	Se requiere que el sistema permita realizar la verificación de la cantidad y la validez de los documentos requeridos frente a la cantidad de documentos cargados por el postulante antes de ser enviado al proceso de evaluación.
RF-24	El sistema permitirá realizar la generación de la estructura documental	Se requiere que cuando se registre un postulante a ser evaluado, se genere la estructura para el almacenamiento de los documentos requeridos y los documentos generados.

RF-25	El sistema permitirá generar un documento de estado de solicitud	Se requiere que el sistema permita realizar la generación de un documento de estado de solicitud, el cual es el Informe Crediticio del postulante firmado y validado por el coordinador.

Fuente: Elaboración propia

Tabla 5: Relación de requerimientos con actores del sistema

CÓDIGO	USUARIO
RF-01	Sistema
RF-02	Asesor - Universidad
RF-03	Sistema
RF-04	Sistema
RF-05	Sistema
RF-06	Administrador
RF-07	Administrador
RF-08	Administrador
RF-09	Administrador
RF-10	Postulante
RF-11	Asesor-Universidad
RF-12	Sistema
RF-13	Postulante
RF-14	Postulante
RF-15	Validador
RF-16	Analista de créditos
RF-17	Analista de créditos
RF-18	Analista de créditos
RF-19	Analista de créditos

RF-20	Analista de créditos
RF-21	Validador, Analista de créditos, Coordinador de becas y créditos, Jefe de Becas y créditos
RF-22	Administrador
RF-23	Validador
RF-24	Sistema
RF-25	Coordinador de Becas y Créditos

Fuente: Elaboración propia

4.1.2. REQUERIMIENTOS NO FUNCIONALES

Tabla 6: Requerimientos no funcionales de la solución

CÓDIGO	REQUERIMIENTO	PROPÓSITO
RNF-01	El sistema deberá contemplar la alta disponibilidad del aplicativo web	El portal contará con acceso fuera de la organización
RNF-02	El sistema deberá soportar los controles elaborados en HTML5 y CSS3 en los principales navegadores	El portal deberá ser desarrollado con HTML5 y CSS3, siendo compatible con IE 11, Chrome y Firefox en sus últimas versiones.
RNF-03	El sistema se ejecutará sobre servicios Cloud.	Todas las capas de la aplicación deberán ser desplegadas en sus correspondientes servicios Cloud para un alto nivel de escalabilidad y tolerancia a fallos.
RNF-04	La infraestructura de servicios deberá ser fácilmente replicable.	Si se necesita desplegar el sistema en otros entornos, o con otros propósitos, la infraestructura de servicios Cloud debería poder replicarse sin dificultad.

RNF-05	El sistema deberá ser altamente escalable	La configuración de los servicios de nube permitirá que se realice un escalamiento vertical en caso de presentarse un incremento en la demanda de solicitudes de usuarios de forma concurrente, volviendo a su estado normal una vez finalice esta demanda.

Fuente: Elaboración propia

4.1.3. ESPECIFICACIÓN DE REQUERIMIENTO

Tabla 7: Historia de usuario 1. Controlar accesos.

HISTORIA DE USUARIO	
Requerimiento: RF-01	**Usuario:** Sistema
Nombre de historia: Controlar accesos	
Prioridad en negocio: Alta	**Riesgo en Desarrollo:** Alto
Puntos estimados: 8	**Iteración asignada:** 1
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: El sistema debe identificar al usuario que ingresa al sistema e identificar sus roles.	
Validación: El sistema valida a los usuarios y le brinda los permisos correspondientes a su perfil.	

Fuente: Elaboración propia

Tabla 8: Historia de usuario 2. Crear postulantes.

Requerimiento: RF-02	**Usuario:** Asesor de Universidad
Nombre de historia: Crear postulantes	
Prioridad en negocio: Alta	**Riesgo en Desarrollo:** Bajo
Puntos estimados: 5	**Iteración asignada:** 2

Programador responsable: Adolfo F. Ibarra Landeo
Descripción: Necesito poder registrar postulantes de forma individual o de forma inmediata mediante un archivo en Excel.
Validación: El asesor puede registrar postulantes por dos medios: carga masiva mediante archivo de Excel pre estructurado e individualmente, mediante formulario.

Fuente: Elaboración propia

Tabla 9: Historia de usuario 3. Enviar alertas.

HISTORIA DE USUARIO	
Requerimiento: RF-03	**Usuario:** Sistema
Nombre de historia: Enviar alertas	
Prioridad en negocio: Baja	**Riesgo en Desarrollo:** Bajo
Puntos estimados: 8	**Iteración asignada:** 3
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: El sistema debe enviar notificaciones internas y externas, dependiendo del estado de las solicitudes. Todo esto a modo de correos electrónicos y mensajes de texto.	
Validación: El sistema envía correos electrónicos y mensajes de texto cuando se realizan asignaciones y cambios a las solicitudes, así como al encontrarse en una fecha cercana al vencimiento de la solicitud.	

Fuente: Elaboración propia

Tabla 10: Historia de usuario 4. Verificar vencimiento de solicitudes.

Requerimiento: RF-04	**Usuario:** Sistema
Nombre de historia: Verificar vencimiento de solicitudes	
Prioridad en negocio: Media	**Riesgo en Desarrollo:** Medio
Puntos estimados: 3	**Iteración asignada:** 4
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: El sistema debe verificar las fechas límite de las solicitudes para determinar si requiere una “limpieza” o se necesita emitir un recordatorio.	
Validación: El sistema valida adecuadamente las solicitudes en base a su fecha límite en los escenarios requeridos.	

Fuente: Elaboración propia

Tabla 11: Historia de usuario 5. Enviar mensajes de bienvenida.

HISTORIA DE USUARIO	
Requerimiento: RF-05	**Usuario:** Sistema
Nombre de historia: Enviar mensajes de bienvenida	
Prioridad en negocio: Baja	**Riesgo en Desarrollo:** Bajo
Puntos estimados: 3	**Iteración asignada:** 3
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: El sistema debe enviar un correo electrónico a los nuevos usuarios de forma automática, indicando los datos de acceso.	
Validación: El sistema realiza el envío de las credenciales de acceso a todos los usuarios nuevos registrados, tanto de manera individual, como masiva.	

Fuente: Elaboración propia

Tabla 12: Historia de usuario 6. Registrar planes de postgrado.

HISTORIA DE USUARIO	
Requerimiento: RF-06	**Usuario:** Administrador
Nombre de historia: Registrar planes de postgrado	
Prioridad en negocio: Alta	**Riesgo en Desarrollo:** Bajo
Puntos estimados: 3	**Iteración asignada:** 1
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: Necesito tener la capacidad de registrar los planes de postgrado en base a las listas de planes enviados por las universidades.	
Validación: El administrador puede registrar planes de postgrado y visualizarlos en una lista.	

Fuente: Elaboración propia

Tabla 13: Historia de usuario 7. Crear usuarios.

HISTORIA DE USUARIO	
Requerimiento: RF-07	**Usuario:** Administrador
Nombre de historia: Crear usuarios	
Prioridad en negocio: Alta	**Riesgo en Desarrollo:** Bajo
Puntos estimados: 3	**Iteración asignada:** 1
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: Necesito registrar usuarios de forma individual o empleando un documento que contenga la lista de los datos de usuarios.	
Validación: El administrador puede registrar postulantes por dos medios: carga masiva mediante archivo de Excel pre estructurado e individualmente, mediante formulario.	

Fuente: Elaboración propia

Tabla 14: Historia de usuario 8. Gestionar perfiles y roles.

HISTORIA DE USUARIO	
Requerimiento: RF-08	**Usuario:** Administrador
Nombre de historia: Gestionar perfiles y roles	
Prioridad en negocio: Media	**Riesgo en Desarrollo:** Bajo
Puntos estimados: 5	**Iteración asignada:** 1
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: Necesito controlar los roles asignados a los usuarios y los permisos que éstos tendrán.	
Validación: El administrador puede asignar roles a los usuarios y asignar permisos a dichos roles.	

Fuente: Elaboración propia

Tabla 15: Historia de usuario 9. Registrar datos maestros.

HISTORIA DE USUARIO	
Requerimiento: RF-09	**Usuario:** Administrador
Nombre de historia: Registrar datos maestros	
Prioridad en negocio: Media	**Riesgo en Desarrollo:** Medio
Puntos estimados: 5	**Iteración asignada:** 1
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: Necesito registrar datos genéricos, como el nombre de las universidades, los puestos laborales, etc.	
Validación: El administrador puede registrar los datos maestros del sistema y ver cada tipo de datos en una lista independiente.	

Fuente: Elaboración propia

Tabla 16: Historia de usuario 10. Registrar información personal.

HISTORIA DE USUARIO	
Requerimiento: RF-10	**Usuario:** Postulante
Nombre de historia: Registrar información personal	
Prioridad en negocio: Alta	**Riesgo en Desarrollo:** Medio
Puntos estimados: 5	**Iteración asignada:** 2
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: Necesito registrar mi información personal para emitir la solicitud de crédito.	
Validación: El postulante puede registrar sus datos personales al ingresar al sistema.	

Fuente: Elaboración propia

Tabla 17: Historia de usuario 11. Asignar solicitudes a colaboradores.

HISTORIA DE USUARIO	
Requerimiento: RF-11	**Usuario:** Asesor de Universidad
Nombre de historia: Asignar solicitudes a colaboradores	
Prioridad en negocio: Alta	**Riesgo en Desarrollo:** Bajo
Puntos estimados: 3	**Iteración asignada:** 2
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: Debo poder delegar las actividades de revisión y evaluación de solicitudes a los colaboradores adecuados.	
Validación: El asesor puede visualizar la lista de colaboradores y sus perfiles para delegar las solicitudes a los usuarios correspondientes.	

Fuente: Elaboración propia

Tabla 18: Historia de usuario 12. Generar log de eventos.

HISTORIA DE USUARIO	
Requerimiento: RF-12	**Usuario:** Sistema
Nombre de historia: Generar log de eventos	
Prioridad en negocio: Baja	**Riesgo en Desarrollo:** Bajo
Puntos estimados: 5	**Iteración asignada:** 4
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: El sistema debe guardar información sobre los cambios y registros realizados para realizar las auditorías correspondientes.	
Validación: El sistema registra los cambios por usuario, fecha, nombre del equipo, tipo de acción, así como si se trata o no de una incidencia.	

Fuente: Elaboración propia

Tabla 19: Historia de usuario 13. Registrar ingresos.

HISTORIA DE USUARIO	
Requerimiento: RF-13	**Usuario:** Postulante
Nombre de historia: Registrar ingresos	
Prioridad en negocio: Alta	**Riesgo en Desarrollo:** Bajo
Puntos estimados: 3	**Iteración asignada:** 2
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: Necesito registrar detalladamente la información sobre mis ingresos y los conyugales, si hubieran.	
Validación: El postulante puede registrar el detalle de sus ingresos personales y los de su cónyuge.	

Fuente: Elaboración propia

Tabla 20: Historia de usuario 14. Registrar egresos.

HISTORIA DE USUARIO	
Requerimiento: RF-14	**Usuario:** Postulante
Nombre de historia: Registrar egresos	
Prioridad en negocio: Alta	**Riesgo en Desarrollo:** Bajo
Puntos estimados: 3	**Iteración asignada:** 2
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: Necesito registrar información sobre mis egresos, así como los del cónyuge, en caso de que exista.	
Validación: El postulante puede registrar el detalle de sus egresos personales y los de su cónyuge, así como los conyugales.	

Fuente: Elaboración propia

Tabla 21: Historia de usuario 15. Registrar comunicaciones.

HISTORIA DE USUARIO	
Requerimiento: RF-15	**Usuario:** Validador
Nombre de historia: Registrar comunicaciones	
Prioridad en negocio: Baja	**Riesgo en Desarrollo:** Bajo
Puntos estimados: 3	**Iteración asignada:** 3
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: Necesito sustentar el seguimiento que se da mediante alguna forma de registro de las llamadas que realizo a cada postulante que evalúo.	
Validación: El validador puede registrar las llamadas realizadas, en la fecha y hora que se realizaron, las observaciones y el resumen de esas llamadas.	

Fuente: Elaboración propia

Tabla 22: Historia de usuario 16. Registrar información cuantitativa.

HISTORIA DE USUARIO	
Requerimiento: RF-16	**Usuario:** Analista de créditos
Nombre de historia: Registrar información cuantitativa	
Prioridad en negocio: Alta	**Riesgo en Desarrollo:** Medio
Puntos estimados: 3	**Iteración asignada:** 2
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: Necesito registrar información relacionada con la capacidad de pago, endeudamiento y demás información cuantitativa con fines de evaluación posterior.	
Validación: El analista puede registrar la información concerniente a los detalles cuantitativos del postulante en un formulario.	

Fuente: Elaboración propia

Tabla 23: Historia de usuario 17. Registrar información cualitativa.

HISTORIA DE USUARIO	
Requerimiento: RF-17	**Usuario:** Analista de créditos
Nombre de historia: Registrar información cualitativa	
Prioridad en negocio: Alta	**Riesgo en Desarrollo:** Medio
Puntos estimados: 3	**Iteración asignada:** 2
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: Necesito registrar información relacionada con la calificación del postulante en el sistema financiero, su situación laboral y demás información cualitativa con fines de evaluación posterior.	
Validación: El analista puede registrar la información concerniente a los detalles cualitativos del postulante en un formulario.	

Fuente: Elaboración propia

Tabla 24: Historia de usuario 18. Generar reporte crediticio.

HISTORIA DE USUARIO	
Requerimiento: RF-18	**Usuario:** Analista de créditos
Nombre de historia: Generar reporte crediticio	
Prioridad en negocio: Media	**Riesgo en Desarrollo:** Medio
Puntos estimados: 8	**Iteración asignada:** 3
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: Necesito emitir un reporte de evaluación financiera al coordinador de becas en base a mi evaluación de las solicitudes.	
Validación: El analista genera un informe de forma inmediata tras determinar su decisión, mediante el uso de un botón, que es armado en base a una plantilla pre diseñada y se envía de forma automática al coordinador de becas.	

Fuente: Elaboración propia

Tabla 25: Historia de usuario 19. Generar cronograma de pagos.

HISTORIA DE USUARIO	
Requerimiento: RF-19	**Usuario:** Analista de créditos
Nombre de historia: Generar cronograma de pagos	
Prioridad en negocio: Media	**Riesgo en Desarrollo:** Alto
Puntos estimados: 13	**Iteración asignada:** 3
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: Necesito elaborar el cronograma de pagos para cada solicitud, con el fin de buscar el resultado más conveniente para el postulante. Esto lo logro en base a diversos criterios y variables que determinan los montos y que poseen cierto nivel de flexibilidad.	
Validación: El analista puede generar un cronograma de pagos, visualizando previamente el cronograma a elaborar en una vista dinámica, la cual cambiará el detalle del cronograma en base a las variables manejadas de forma directa por el analista.	

Fuente: Elaboración propia

Tabla 26: Historia de usuario 20. Registrar autorización de descuentos.

HISTORIA DE USUARIO	
Requerimiento: RF-20	**Usuario:** Analista de créditos
Nombre de historia: Registrar autorización de descuentos	
Prioridad en negocio: Baja	**Riesgo en Desarrollo:** Medio
Puntos estimados: 5	**Iteración asignada:** 3
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: Necesito establecer un porcentaje de descuento a ciertos postulantes. Esta decisión depende de mí análisis del estado financiero de los postulantes.	
Validación: El analista puede asignar descuentos a los postulantes de forma individual, observando el detalle de la información emitida por cada solicitud.	

Fuente: Elaboración propia

Tabla 27: Historia de usuario 21. Determinar estado de solicitudes.

HISTORIA DE USUARIO	
Requerimiento: RF-21	**Usuario:** Validador, Analista de créditos, Coordinador, Jefe de Becas
Nombre de historia: Determinar estado de solicitudes	
Prioridad en negocio: Alta	**Riesgo en Desarrollo:** Bajo
Puntos estimados: 5	**Iteración asignada:** 2
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: Necesito determinar si las solicitudes continuarán el proceso de evaluación, si serán rechazadas o necesitarán alguna modificación una vez que lleguen a mi área.	
Validación: Los usuarios que pertenecen a la ONG y las universidades pueden tomar decisiones sobre las solicitudes (aceptación, solicitar cambio de información, rechazo) en base a la evaluación de cada uno en la etapa correspondiente.	

Fuente: Elaboración propia

Tabla 28: Historia de usuario 22. Gestionar carga documental.

HISTORIA DE USUARIO	
Requerimiento: RF-22	**Usuario:** Administrador
Nombre de historia: Gestionar carga documental	
Prioridad en negocio: Media	**Riesgo en Desarrollo:** Alto
Puntos estimados: 21	**Iteración asignada:** 3
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: Necesito establecer el listado de documentos que se deberá enviar el postulante, basado en la información que se registró de este. Este listado es dinámico y suele cambiar cada cierto tiempo.	

Validación: El administrador puede establecer una casuística seleccionando las características que serán presentadas al usuario. En base a esa casuística, puede seleccionar la lista de documentos que requerirá el postulante que use esa misma casuística.

Fuente: Elaboración propia

Tabla 29: Historia de usuario 23. Verificar documentos de postulante.

HISTORIA DE USUARIO	
Requerimiento: RF-23	**Usuario:** Validador
Nombre de historia: Verificar documentos de postulante	
Prioridad en negocio: Alta	**Riesgo en Desarrollo:** Bajo
Puntos estimados: 5	**Iteración asignada:** 1
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: Necesito verificar si el postulante envió todos los documentos requeridos o no, así como asegurar que los documentos enviados por el postulante sean los correctos.	
Validación: El validador puede realizar la verificación de los documentos de cada solicitud de forma independiente, visualizando el contenido de cada documento adjunto.	

Fuente: Elaboración propia

Tabla 30: Historia de usuario 24. Generar estructura de documentos.

HISTORIA DE USUARIO	
Requerimiento: RF-24	**Usuario:** Sistema
Nombre de historia: Generar estructura de documentos	
Prioridad en negocio: Alta	**Riesgo en Desarrollo:** Medio
Puntos estimados: 5	**Iteración asignada:** 1

Programador responsable: Adolfo F. Ibarra Landeo
Descripción: El sistema debe mostrar la lista de los documentos establecidos para el postulante y solicitar la carga de todos ellos.
Validación: El sistema le muestra a cada postulante los documentos que corresponden con los valores que ingresó durante su proceso de registro previo.

Fuente: Elaboración propia

Tabla 31: Historia de usuario 25. Generar documento de estado de solicitud.

HISTORIA DE USUARIO	
Requerimiento: RF-25	**Usuario:** Coordinador de Becas
Nombre de historia: Generar documento de estado de solicitud	
Prioridad en negocio: Media	**Riesgo en Desarrollo:** Medio
Puntos estimados: 8	**Iteración asignada:** 3
Programador responsable: Adolfo F. Ibarra Landeo	
Descripción: Necesito generar un reporte de estado de solicitudes para el Jefe de Becas y créditos con información de varios grupos de solicitudes.	
Validación: El coordinador puede generar un reporte detallado con el total de las solicitudes, y su resultado, al jefe de becas y créditos mediante el uso de un botón que carga de forma automática la información en una plantilla diseñada para la ONG.	

Fuente: Elaboración propia

4.2. ANÁLISIS DE LA SOLUCIÓN

4.2.1. NECESIDADES DEL CLIENTE

Se plasmaron las necesidades del cliente en el listado de requerimientos, los cuales fueron identificados tras las reuniones de coordinación correspondientes y agrupados en tres grupos de actividades.

A esto queda añadir el tiempo que tiene el cliente para poder realizar la implementación de la aplicación, ya que gran parte de las decisiones y del apuro que se tuvo al realizar esta aplicación, fueron motivadas por la presión que ejercían algunas entidades bancarias para continuar dando el apoyo correspondiente y las facilidades de pago adecuadas.

4.2.2. IDENTIFICACIÓN DE CASOS DE USO

4.2.2.1. Definición De Casos De Uso Más Importantes

- Registrar planes de postgrado: Mediante esta funcionalidad, se realiza el registro de los planes de postgrado a los que los postulantes serán asignados para su evaluación.
- Asignación de solicitudes: Esta funcionalidad permite asignar una solicitud registrada a un miembro del personal para dar inicio al proceso de evaluación de la misma.
- Generar cronograma de pagos: Esta funcionalidad hace referencia a la capacidad de los evaluadores de generar un cronograma de pagos automático, dependiendo del tipo de pago y del tiempo de duración de un plan de postgrado, así como de modificar ciertos valores para acomodar los resultados a unos que sean convenientes tanto para el postulante, como para la ONG.
- Generar reporte crediticio: Al finalizar el proceso de evaluación de una solicitud, se debe generar un reporte que indique la aceptación o el rechazo de una solicitud.
- Evaluar solicitud: Las solicitudes que son asignadas e inician el proceso de evaluación, deben poder ser aceptadas, rechazadas o devolverlas al proceso anterior para su evaluación.
- Generar carga documentaria: Dependiendo del perfil del estudiante (características cualitativas y cuantitativas), se generará una

estructura documentaria para la petición de carga de los documentos en el proceso de registro de información.

4.2.2.2. Catálogo Completo De Casos De Uso

Tabla 32: Catálogo de casos de uso (CU)

CÓDIGO	CASO DE USO	REQUERIMIENTO ASOCIADO
CU-01	Iniciar sesión	RE-01
CU-02	Cambiar contraseña	RE-01
CU-03	Importar postulantes	RE-02
CU-04	Registrar postulantes	RE-02
CU-05	Enviar correos electrónicos	RE-03, RE-05
CU-06	Enviar mensajes de texto	RE-03, RE-05
CU-07	Verificar solicitudes	RE-04
CU-08	Limpiar solicitudes	RE-04
CU-09	Registrar planes de postgrado	RE-06
CU-10	Importar usuarios	RE-07
CU-11	Registrar usuarios	RE-07
CU-12	Administrar perfiles y roles	RE-08
CU-13	Registrar datos globales	RE-09
CU-14	Registrar datos personales del postulante	RE-10
CU-15	Asignación de solicitudes	RE-11
CU-16	Registrar eventos	RE-12
CU-17	Registrar ingresos del postulante	RE-13
CU-18	Registrar egresos del postulante	RE-14
CU-19	Registrar comunicaciones	RE-15
CU-20	Registrar información cuantitativa del postulante	RE-16
CU-21	Registrar información cualitativa del postulante	RE-17

CU-22	Generar reporte crediticio	RE-18
CU-23	Generar cronograma de pagos	RE-19
CU-24	Registrar descuentos	RE-20
CU-25	Asignar descuentos	RE-20
CU-26	Evaluar solicitud	RE-21
CU-27	Cargar archivos	RE-22
CU-28	Visualización de archivos	RE-23
CU-29	Evaluación de archivos	RE-23
CU-30	Generar carga documentaria	RE-24

Fuente: Elaboración propia

4.2.3. VIABILIDAD DEL SISTEMA

Los requerimientos descritos son totalmente alcanzables, pudiendo ser satisfechos con una solución distribuida a través de dos aplicativos: un servicio web que se comunique con una base de datos y un storage para el almacenamiento de archivos, y una aplicación web que se comunique con el servicio web, todos montados en el servicio de nube de Microsoft Azure. Se planteó la posibilidad de desplegar el sistema desarrollado en el entorno de producción interno que manejan, pero las limitaciones de almacenamiento y capacidad del servidor que tenían montado, la falta de presupuesto para la obtención de nuevos recursos que no serían aprovechados al 100% y la preocupación de cara a los accesos remotos dejaron en evidencia la necesidad de la adquisición de servicios de nube, sobre todo por la flexibilidad que estos ofrecen a nivel de costos.

4.2.4. ANÁLISIS TÉCNICO Y ECONÓMICO

4.2.4.1. ANÁLISIS TÉCNICO

El sistema está pensando para ser integrado, a futuro, con el sistema principal que el cliente posee, el cual registra las actividades y el estado de los beneficios educativos de los postulantes, de cara a las entidades con las que tienen relación actualmente, teniendo pensado integrarse ambos sistemas a futuro. Bajo este esquema, se planteará el esquema y el diseño de la base de datos, así como las propiedades de sus campos, de acuerdo al esquema actual que poseen. La elección de la nube de Microsoft Azure para el despliegue del sistema ha sido tomada en cuenta por criterios que van más allá del espectro económico que se planteó anteriormente. La escalabilidad es un factor importante para el cliente, debido a que se encuentran incrementando constantemente la cantidad de instituciones educativas con las que realizan acuerdos. Por esta razón, y debido a las restricciones y criterios propios de cada institución educativa, la elección de un despliegue en la nube representa un beneficio, ya que es factible y sencillo realizar un clonado de configuración mediante archivos en formato JSON (provistos por los servicios de nube de Microsoft), realizar las modificaciones necesarias a nivel de código para su compilación, configurar las cadenas de destino del despliegue de las soluciones y proceder con el despliegue correspondiente. Además, el uso de la nube de Microsoft Azure facilita la integración con herramientas de gestión de proyectos de software que contemplan la metodología SCRUM, como lo es Visual Studio Team Services (VSTS). Ambas soluciones se integran, además, de forma nativa con el entorno de desarrollo integrado (IDE, por sus siglas en inglés), Visual Studio.

Cabe destacar que los servicios a ser utilizados a nivel de aplicativo, como son el Storage, Web App y Cloud Service, son agnósticos a la tecnología de desarrollo del mismo. Siendo la base de datos, la única que estaría ligada de forma directa al uso de SQL Server como sistema de gestión de base de datos. Cada uno de estos servicios se ejecutan como instancias de una

máquina virtual, aunque el monto es considerablemente menor que el de contratar una máquina virtual como tal, de forma directa. Por la naturaleza del proyecto no ha sido necesario realizar la adquisición de hardware demasiado potente, pero sí de aquellos planes de hardware en la nube que se encuentran en los planes más básicos.

Otro de los factores que se tuvo en cuenta al momento de elegir la tecnología es la curva de aprendizaje del equipo, que ya tenía un fuerte background en el desarrollo con tecnologías Microsoft. De haberse visto incrementada esta curva de aprendizaje, el proyecto habría sufrido un desfase considerable, provocando problemas para ambos bandos, tanto para el cliente como para el equipo de desarrollo.

Con respecto a los envíos de mensajes de texto, se optó por el uso del servicio de mensajería de Twilio, el cual posee planes flexibles y puede integrarse de manera sencilla en el archivo de opciones de configuración de las aplicaciones ASP.NET, “web.config”.

4.2.4.2. ANÁLISIS ECONÓMICO

Para el análisis económico, no se está tomando en cuenta los ahorros o ingresos obtenidos por el lado del cliente tras el despliegue del sistema, debido a que la solicitud para el desarrollo del mismo fue presentada tras un análisis interno por parte de la ONG, en coordinación con su área de Tecnologías de Información. Luego del análisis interno, se otorgó el presupuesto solicitado y, en base a ese presupuesto, iniciaron la búsqueda de un proveedor de soluciones.

Asimismo, el análisis de los costos por el desarrollo del sistema no puede ser presentado en la presente tesis, debido a que existen cláusulas acordadas de confidencialidad y privacidad con respecto a estos puntos. Por esta razón, la cobertura de este análisis se limitará a conocer el costo de mantener los sistemas en un entorno de despliegue y de desarrollo. Luego, se evaluará los costos totales de la contratación de los servicios en

la nube, debido a que el escenario que exigen este tipo de servicios es diferente de cara a individuales y empresas.

Consumo de computación en la nube con Microsoft Azure

Los egresos derivados del consumo de servicios en la nube poseen cierta elasticidad. Mantener la aplicación desplegada en estos servicios constituye un gasto mensual proyectado, el cual es calculado considerando la adquisición de “licencias”, cada una de las cuales tiene un valor de USD100.00 (cien dólares). Un punto clave a considerar aquí es que el costo de mantener el entorno de desarrollo es menor al que se obtiene en un ambiente de despliegue. Además, no es proyectado por el período de un año, sino que se realiza sólo por los meses que dure el desarrollo de la solución de software, en este caso se considerará el período de cuatro meses.

Tabla 33: Cálculo de costos para implementación de infraestructura de desarrollo

TIPO DE SERVICIO	REGIÓN	DESCRIPCIÓN	COSTO ESTIMADO	CANTIDAD
Web App	West US 2	Standard Tier; 1 S1 (1 Core(s), 1.75 GB RAM, 50 GB Storage) x 730 Hours; Windows OS	$43.80	1
Cloud Services	West US 2	A2 v2, 2 Core(s), 4 GB RAM, 20 GB Temporary Storage, 1 Instance(s), 730 Hours	$66.43	1
Storage	West US 2	Block Blob Storage, General Purpose V2, LRS Redundancy, Hot Access Tier, 1000 GB Capacity, 100,000 Write operations, 100,000 List and Create Container Operations, 100,000 Read operations, 1 Other operations. 1,000 GB Data Retrieval, 1,000 GB Data Write	$19.44	1
Azure SQL Database	West US 2	Single Database, DTU Purchase Model, Standard Tier, S1: 20 DTUs, 250 GB included storage per DB, 1 Database(s) x 730 Hours, 5 GB Retention	$29.43	2
Soporte		Soporte	$0.00	
		Total Mensual	**$188.59**	
		Total durante desarrollo	**$754.36**	

Fuente: Elaboración propia.

Teniendo el costo total derivado por el consumo de los servicios de nube para el entorno de desarrollo, se procede con el cálculo de estos mismos costos, con los mismos servicios, para el entorno de despliegue de la solución.

Tabla 34: Cálculo de costos para implementación de infraestructura de producción.

TIPO DE SERVICIO	REGIÓN	DESCRIPCIÓN	COSTO ESTIMADO	CANTIDAD
Web App	West US 2	Standard Tier; 1 S1 (1 Core(s), 1.75 GB RAM, 50 GB Storage) x 730 Hours; Windows OS	$73.00	2
Cloud Service	West US 2	A2 v2, 2 Core(s), 4 GB RAM, 20 GB Temporary Storage, 1 Instance(s), 730 Hours	$99.28	2
Storage	West US 2	Block Blob Storage, General Purpose V2, LRS Redundancy, Hot Access Tier, 1000 GB Capacity, 100,000 Write operations, 100,000 List and Create Container Operations, 100,000 Read operations, 1 Other operations. 1,000 GB Data Retrieval, 1,000 GB Data Write	$19.44	1
Azure SQL Database	West US 2	Single Database, DTU Purchase Model, Standard Tier, S1: 20 DTUs, 250 GB included storage per DB, 1 Database(s) x 730 Hours, 5 GB Retention	$29.43	2
Traffic Manager	West US 2	2 million DNS queries/mo, 2 Azure endpoint(s), 0 Fast Azure endpoint(s), 0 External endpoint(s), 0 Fast External endpoint(s), 2 million(s) of user measurements, 2 million(s) of data points processed.	$9.80	2
Soporte		Soporte	$0.00	
		Total Mensual	**$442.46**	
		Total Anual	**$5309.52**	

Fuente: Elaboración propia

En base al cálculo obtenido en los cuadros anteriores, se calcula la cantidad de licencias necesarias, las cuales tienen validez por el período

de un año, no acumulable. En este proceso obtenemos el costo total anual para soportar el sistema desplegado.

Tabla 35: Cálculo total de costos de licenciamiento para entornos de desarrollo y producción.

CONCEPTO	COSTO	CANTIDAD	COSTO TOTAL	TIPO
Licencias Open para Microsoft Azure para desarrollo	$100.00	8	$800.00	Pago único
Licencias Open para Microsoft Azure para despliegue	$100.00	54	$5400.00	Anual

Fuente: Elaboración propia

Consumo de servicio de mensajería instantánea de Twilio

Los costos derivados de este servicio ofrecen, del mismo modo, un alto nivel de elasticidad, considerando un pago fijo por cada mensaje de texto enviado. El proceso de atención actual y la necesidad de la atención personal de los postulantes para la recolección de documentos y la entrega en formato impreso entre los actores involucrados limita la cantidad de solicitudes procesadas mensualmente a sólo 150, pero proyectan un total de 600 a 800 solicitudes atendidas mediante la implementación de este sistema. Esto es debido a que serán los postulantes los que realizarán la carga digital de sus documentos en los plazos establecidos y no será necesario hacer la entrega física de los documentos a las áreas correspondientes, así como tampoco será menester recibir las respuestas en formato impreso.

Tabla 36: Cálculo total de costos anuales por envío de mensajes de texto.

COSTO POR SMS	CANTIDAD POR SOLICITUD	SOLICITUDES/MES	COSTO MENSUAL	TOTAL ANUAL
$0.0075	3 mensajes	600	$13.50	$162.00
$0.0075	3 mensajes	800	$18.00	$266.00

Fuente: Elaboración propia

Tal y como se aprecia en la Tabla 36, el costo total del envío de mensajes de texto oscilará entre los $162.00 y $266.00 dólares anuales, dependiendo de la cantidad de solicitudes que se procesen, ya que en todos los casos serán 3 mensajes de texto enviados a los postulantes.

4.2.5. FUNCIONES

El sistema desarrollado cubrirá todos los requerimientos especificados en este documento, que fueron recogidos en un proceso persistente de comunicación con el equipo de TI de la ONG. En el backend, el sistema proveerá servicios de mensajería instantánea mediante la plataforma Twilio, almacenamiento de archivos en una unidad de almacenamiento preparada para este aplicativo en la nube de Microsoft Azure, métodos de comunicación entre la aplicación web y el servicio web, métodos para el envío de correos electrónicos y comunicación con Azure WebJobs programadas para la verificación diaria de solicitudes que se encuentren cerca al límite de la fecha de presentación de documentos, entre otros.

Asimismo, los App Services del tenant de Microsoft Azure, propiedad del cliente, serán configurados para realizar una copia de seguridad de la aplicación de forma semanal, lo cual permitirá contar con un plan de contingencia si los balanceadores de carga y las dos instancias configuradas para el despliegue de cada aplicativo no se encuentran disponibles. En este tipo de escenarios, se usará un archivo de replicación proporcionado por Microsoft Azure, que permitirá montar la misma arquitectura de servicios, con sus configuraciones correspondientes para ser usadas de inmediato.

4.2.6. CRONOGRAMA DE ACTIVIDADES

Tabla 37: Cronograma de actividades

NOMBRE DE TAREA	DURACIÓN	INICIO	TÉRMINO	RESPONSABLE
PROYECTO: Sistema Evaluación Crediticia	**85 days**	**Mon 04-04-16**	**Wed 03-08-16**	
Visión y Alcance	**1 day**	**Mon 04-04-16**	**Mon 04-04-16**	Adolfo F. Ibarra Landeo
Diseño	**10 days**	**Tue 05-04-16**	**Mon 18-04-16**	Adolfo F. Ibarra Landeo
Desarrollo	**68 days**	**Tue 19-04-16**	**Fri 22-07-16**	Adolfo F. Ibarra Landeo
Sprint 01: Datos Maestros	**10 days**	**Tue 19-04-16**	**Mon 02-05-16**	Adolfo F. Ibarra Landeo
Sprint 02: Flujo de Evaluación	**23 days**	**Tue 03-05-16**	**Thu 02-06-16**	Adolfo F. Ibarra Landeo
Sprint 03: Observaciones y Pendientes	**15 days**	**Tue 07-06-16**	**Mon 27-06-16**	Adolfo F. Ibarra Landeo
Modificación de términos de Campos	1 day	Tue 07-06-16	Tue 07-06-16	Adolfo F. Ibarra Landeo
Lógica de Descuento por Cuota Inicial	8 days	Wed 08-06-16	Fri 17-06-16	Adolfo F. Ibarra Landeo
Campo de OBSERVACIÓN para el VALIDADOR	8 days	Wed 08-06-16	Fri 17-06-16	Adolfo F. Ibarra Landeo
Sección de DESCUENTO Manual	3 days	Wed 08-06-16	Fri 10-06-16	Adolfo F. Ibarra Landeo
Generación de Cronograma de pagos	7 days	Mon 13-06-16	Tue 21-06-16	Adolfo F. Ibarra Landeo
Generación de PDF por Facturación Mensual	2 days	Wed 22-06-16	Thu 23-06-16	Adolfo F. Ibarra Landeo
Generación de PDF por coordinador	1 day	Wed 22-06-16	Wed 22-06-16	Adolfo F. Ibarra Landeo
Generación de reporte de estatus crediticio	1 day	Thu 23-06-16	Thu 23-06-16	Adolfo F. Ibarra Landeo
Despliegue y validación de llamadas a pestañas de movimientos y patrimonios	1 day	Thu 23-06-16	Thu 23-06-16	Adolfo F. Ibarra Landeo
Modificación de campos de evaluación para uso del analista	1 day	Thu 23-06-16	Thu 23-06-16	Adolfo F. Ibarra Landeo
Envío de notificaciones internas con cada etapa culminada	2 days	Fri 24-06-16	Mon 27-06-16	Adolfo F. Ibarra Landeo

Sprint 04: Reportes	**2 days**	**Tue 28-06-16**	**Thu 30-06-16**	Adolfo F. Ibarra Landeo
Sprint 05: Controles de Cambio	**16 days**	**Fri 01-07-16**	**Fri 22-07-16**	Adolfo F. Ibarra Landeo
Carga de Archivos Adicional	6 days	Fri 01-07-16	Fri 08-07-16	Adolfo F. Ibarra Landeo
Pre visualización de documentos requeridos.	4 days	Mon 11-07-16	Thu 14-07-16	Adolfo F. Ibarra Landeo
Notificación de compromiso por correo electrónico.	3 days	Fri 15-07-16	Tue 19-07-16	Adolfo F. Ibarra Landeo
Visualización del estado de la solicitud.	1 day	Wed 20-07-16	Wed 20-07-16	Adolfo F. Ibarra Landeo
Regresión de estados de solicitud	2 days	Thu 21-07-16	Fri 22-07-16	Adolfo F. Ibarra Landeo
Estabilización y Pruebas	**5 days**	**Mon 25-07-16**	**Tue 02-08-16**	Adolfo F. Ibarra Landeo
Despliegue	**2 days**	**Tue 02-08-16**	**Wed 03-08-16**	Adolfo F. Ibarra Landeo
Cierre del Proyecto	**1 day**	**Wed 03-08-16**	**Wed 03-08-16**	Adolfo F. Ibarra Landeo

Fuente: Elaboración propia

4.2.7. ARQUITECTURA DE LA SOLUCIÓN

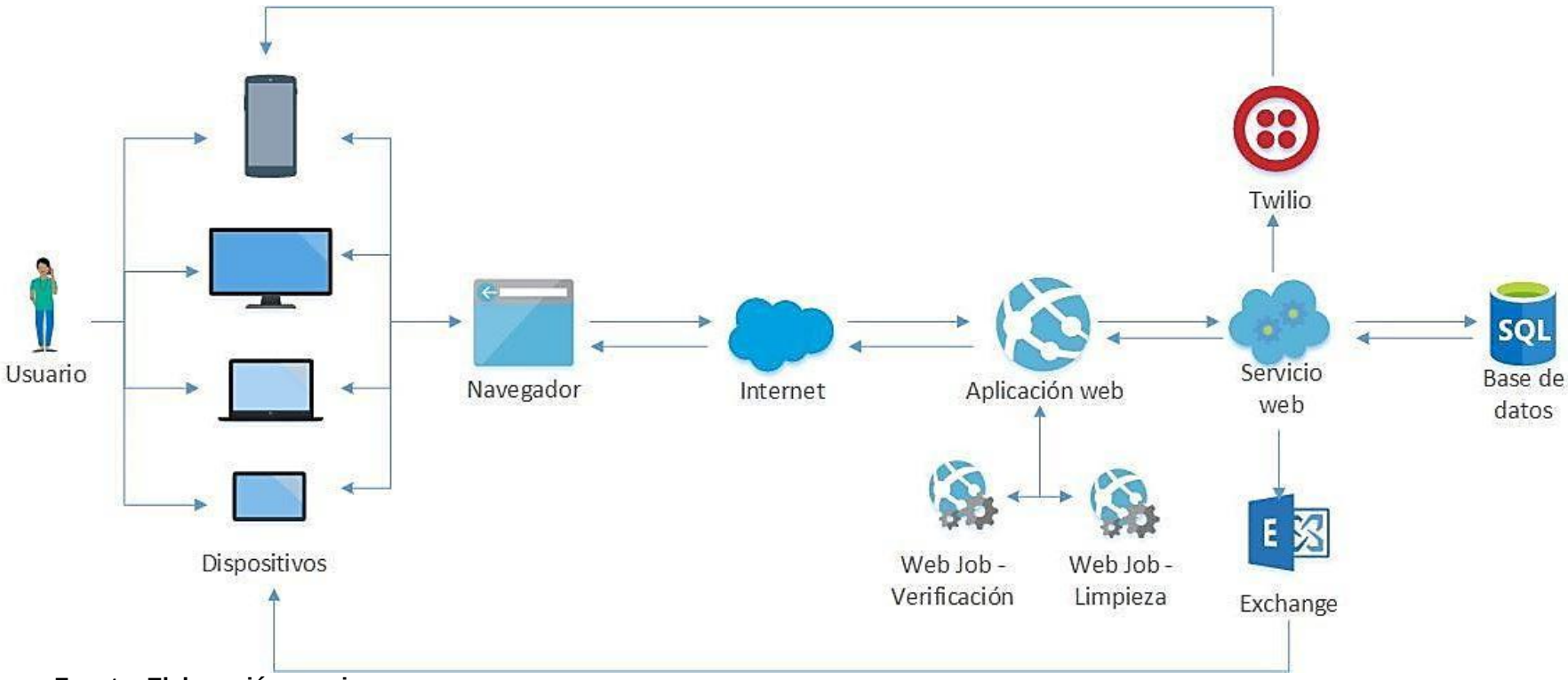
4.2.7.1. Representación De La Arquitectura

La arquitectura a elegida consideró los lineamientos del paradigma SOA, orientado a servicios web. En este esquema de trabajo, las funciones y la carga de trabajo irán orientadas hacia el lado del servicio principalmente. Sin embargo, se delegó una carga mínima a la aplicación web para atender las necesidades del CU-23, “Generar cronograma de pagos”. Con estas consideraciones aseguraremos la disponibilidad del servicio y la accesibilidad fluida desde un dispositivo con acceso a internet.

La Figura 17 nos muestra la arquitectura propuesta, la cual describe el acceso de los usuarios al servicio desde cualquier dispositivo con acceso a internet, todo mediante el uso de un navegador web. Este servicio web se comunica con diferentes componentes provistos por Microsoft Azure para el manejo y el despliegue de sus capas y componentes, así como también con

servicios de terceros como Twilio, para el envío de mensajes de texto, y Microsoft Exchange, para el envío de correos electrónicos.

Figura 17: Representación de la arquitectura general



Fuente: Elaboración propia

.

La arquitectura debe considerar:

Respetar el paradigma de la programación orientada a objetos (OOP, por sus siglas en inglés). Para lograr flexibilidad en el manejo de cambios y que los mismos no acaben afectando porciones o módulos del programa que no comparten los mismos tipos de recursos y objetos, se empleará el lenguaje de programación C#, empleando una propuesta de diseño que asegure el uso de los datos mediante la encapsulación de los objetos y las clases relacionados, haciendo llamadas a los métodos correspondientes

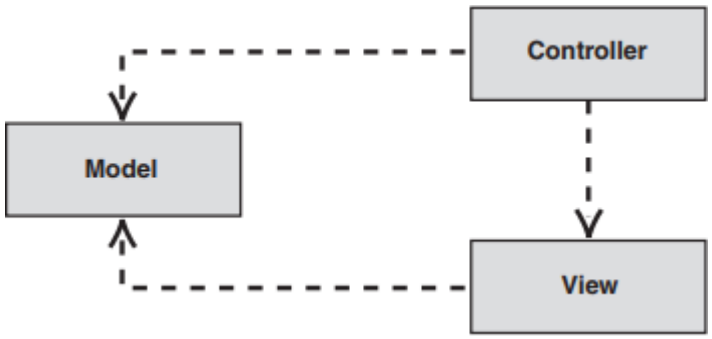
La capa de lógica de negocio utilizará una arquitectura basada en el patrón Modelo de Dominio. En este patrón se crean objetos aplicables al negocio, que actúan como representaciones de las entidades del dominio, así como también las relaciones que poseen. La característica de este patrón es que recrea el flujo de trabajo en base a las reglas de negocio. Asimismo, existe un buen nivel de desacoplamiento con las otras capas, ya que la capa de lógica de negocio es agnóstica de las tecnologías y patrones de las otras capas.

En cuanto a la capa de datos, el patrón elegido fue el de Repository. Este patrón encapsula los objetos de la base de datos, así como los métodos correspondientes a las operaciones que ha de realizar, de lectura y escritura. Con esto se logra extraer de la capa de lógica de negocio la responsabilidad que le corresponde a la capa de datos.

La capa de aplicación se realizará utilizando el patrón Model-View-Controller (MVC) (ver Figura 18). El patrón MVC permite el desacoplamiento en tres componentes: Modelo, Controlador y Vista. Sin embargo, con el uso del framework Knockout.JS en la capa de presentación, que utiliza el patrón Model-View-ViewModel (MVVM), y habiéndose creado un servicio para manejar la lógica del negocio, la capa Controller de la aplicación será utilizada para la comunicación entre el servicio y la presentación, mientras que en la capa de presentación se

usarán los ViewModel requeridos por este framework, así podrán ser invocados desde la vista de este mismo aplicativo.

Figura 18: Componentes y comportamientos de MVC



Fuente: Microsoft Corporation, 2003: 36 (15)

Bajo estos enfoques, se utilizará la arquitectura de N-Capas, ya que encaja perfectamente con la necesidad que se tiene de lograr un desacoplamiento fuerte entre las capas de presentación, la lógica de negocio y la base de datos. Además, cada una de estas capas posee componentes que pueden comunicarse sin problemas dentro de la misma capa o con aquellos que se encuentren en las capas inferiores. Este tipo de arquitectura resulta perfecto, debido a que en el entorno de nube que se propondrá, las capas se encontrarán físicamente separadas a nivel de diversos servicios comunicados, lo cual no es problema para una arquitectura distribuida bajo estos planteamientos.

El uso de una arquitectura de N-Capas permitirá, también, obtener una mayor flexibilidad al momento de realizar cambios al sistema, ya que las responsabilidades de cada capa se encuentran separadas y bien definidas, acelerando el proceso de corrección de los errores. Esta separación de componentes otorgará, además, la facilidad de desplegar múltiples instancias de dichas capas para realizar un balanceo de carga y aumentar la tolerancia a fallos dentro del entorno Cloud. Dicho esto, cabe resaltar que

los requerimientos no funcionales están siendo cubiertos por la arquitectura propuesta, tal y como lo muestra la Tabla 38:

Tabla 38: Requerimientos no funcionales cubiertos por la arquitectura propuesta.

CÓDIGO	REQUERIMIENTO	SOLUCIÓN
RNF-01	El sistema deberá contemplar la alta disponibilidad del aplicativo web	El uso de soluciones Cloud, como Microsoft Azure, permitirá cumplir con este requerimiento, al aplicar múltiples instancias del aplicativo y manejar balanceadores de carga.
RNF-02	El sistema deberá soportar los controles elaborados en HTML5 y CSS3 en los principales navegadores	Al estar separados los componentes, el desarrollo de la capa de presentación se hará considerando estos criterios sin afectar ni verse afectado por las otras capas.
RNF-03	El sistema se ejecutará sobre servicios Cloud.	La separación de capas facilita el despliegue de este tipo de soluciones a entornos Cloud.
RNF-04	La infraestructura de servicios deberá ser fácilmente replicable.	Los archivos JSON generados por el servicio de nube de Microsoft Azure, permiten realizar una réplica exacta de la infraestructura sin dificultad, siendo posible utilizarlos en diversos entornos y situaciones.
RNF-05	El sistema deberá ser altamente escalable	El uso de los servicios de Microsoft Azure y la configuración adecuada del servicio desarrollado en WCF permitirá la escalabilidad adecuada ante un pico alto o bajo de usuarios en cualquier situación.

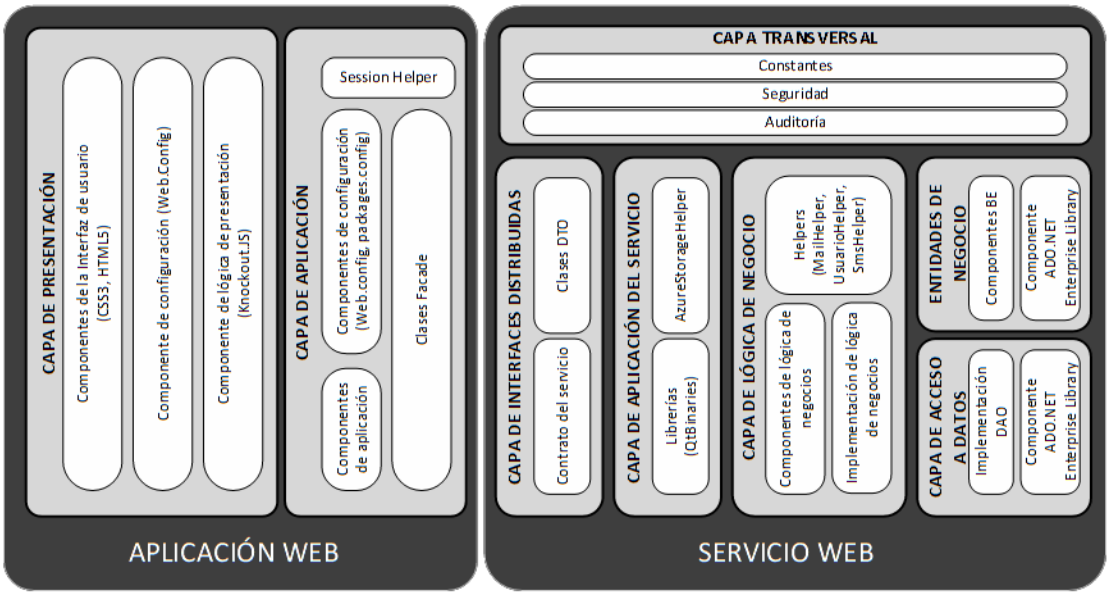
Fuente: Elaboración propia

4.2.7.2. Vista Lógica

La Figura 19 nos muestra la representación lógica del sistema a desarrollar, con las cuatro capas principales mostradas en la sección anterior, así como

con algunos componentes y secciones adicionales, todos necesarios para su funcionamiento.

Figura 19: Arquitectura propuesta para la solución



Fuente: Elaboración propia

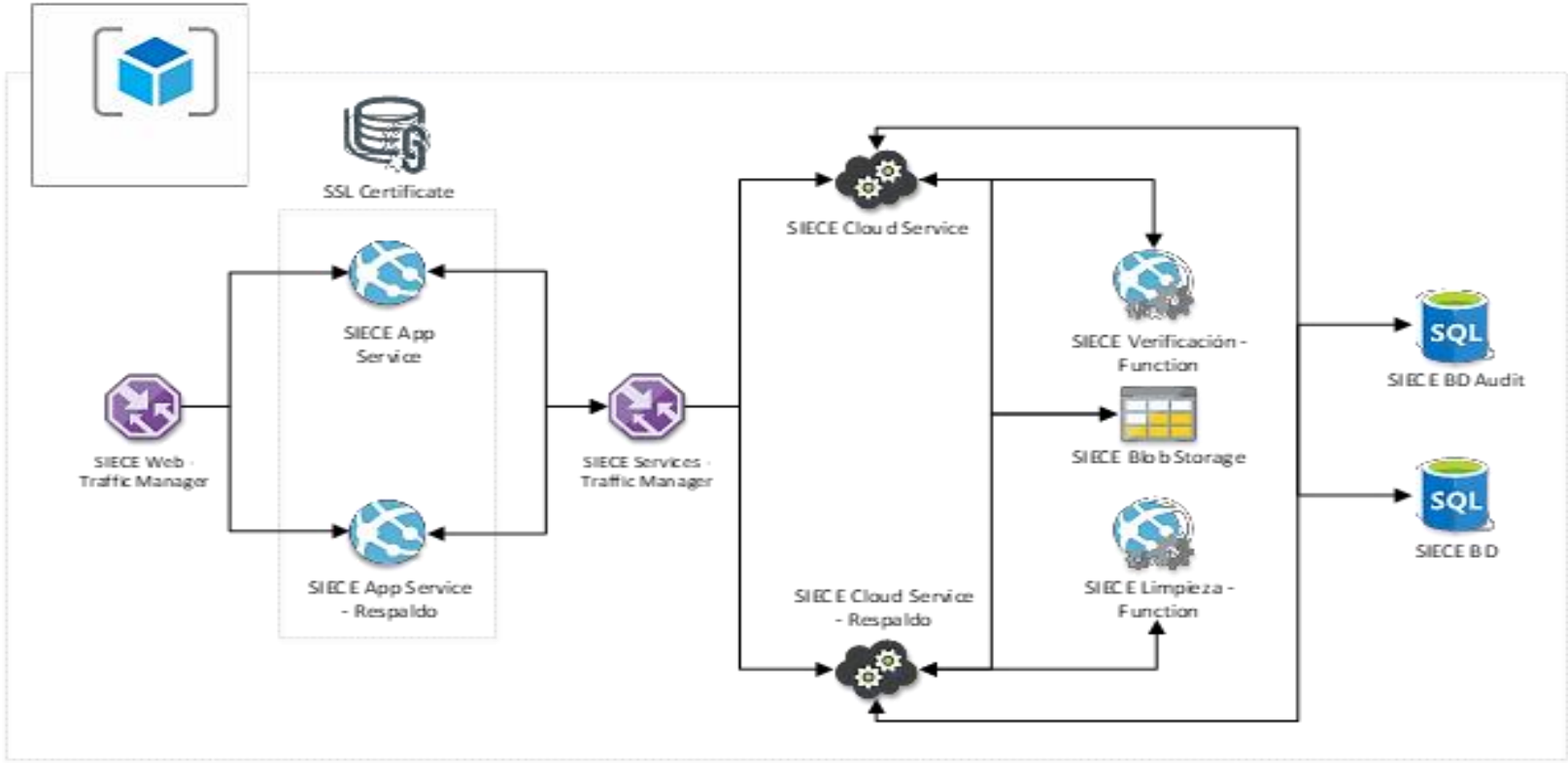
4.2.8. DIAGRAMA DE INFRAESTRUCTURA EN LA NUBE

El usuario se conectará mediante un navegador web al Traffic Manager de la aplicación web, el cual se encargará de dirigir el tráfico a una de las dos instancias del Web App en el que estará desplegada la solución web. Cabe mencionar que esta aplicación web respaldará su validez mediante la instalación de un certificado SSL en los Web Apps correspondientes.

La aplicación desarrollada se comunicará con una de las instancias disponibles del servicio web, desarrollado con Windows Communication Foundation (WCF), a través del Traffic Manager del servicio. La información será consultada y enviada a las bases de datos desde ambos servicios, así como del Blob Storage, en el que se almacenarán los documentos emitidos por el postulante, relacionados a una solicitud específica.

Asimismo, los servicios web mantendrán comunicación con dos servicios de Azure WebJobs, los cuales estarán encargados de la validación y limpieza de las solicitudes de forma periódica y automática. Estos servicios web se comunicarán de forma directa con el servicio de Twilio para realizar el envío de notificaciones por mensajes de texto (SMS). La Figura 20 muestra el diagrama a desplegar en el servicio de nube de Microsoft Azure para el entorno de producción.

Figura 20: Diagrama de Infraestructura para servicios Cloud en Microsoft Azure.

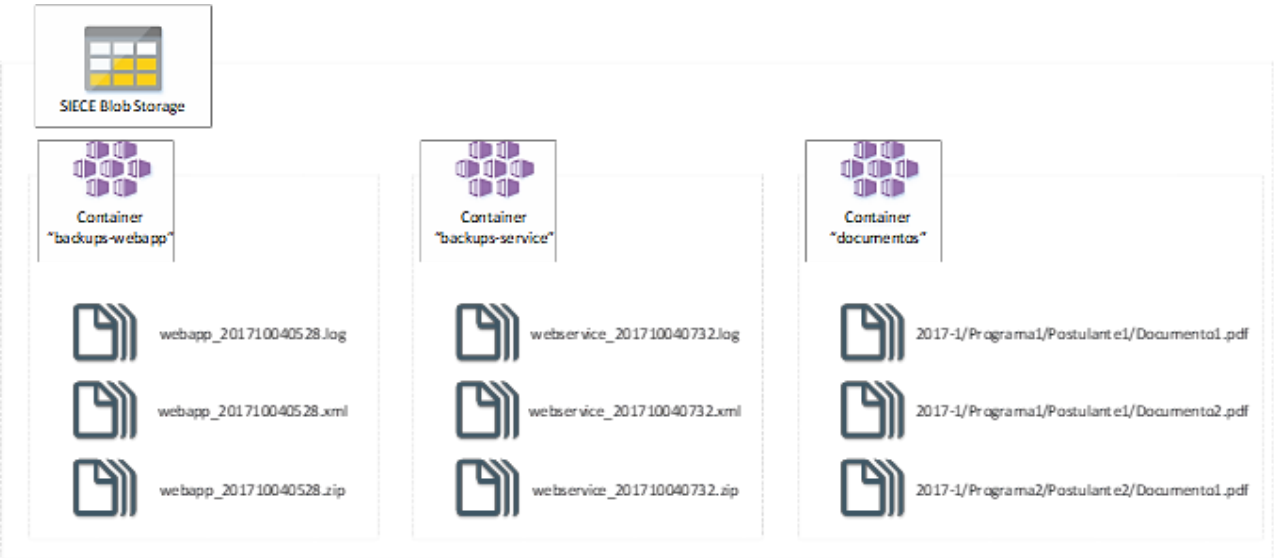


Fuente: Elaboración propia

4.2.9. ESTRUCTURA DE ARCHIVOS

La estructura de archivos e información almacenada en el Storage de Microsoft Azure, perteneciente al tenant del cliente, considerará las copias de seguridad de la solución integral, así como los archivos cargados por los postulantes y los informes de resultado del proceso de evaluación. El servicio de Azure Storage presentará la estructura de archivos y carpetas mostrada en la Figura 21.

Figura 21: Estructura de archivos propuesta para Azure Storage.



Fuente: Elaboración propia

Para la estructura de las copias de seguridad, se ha realizado la creación de dos Containers denominado “backups-webapp” y “backups-service” dentro del Storage, el cual es un espacio físico (similar a las carpetas de archivos) que agrupa Blobs (que es una agrupación de datos binarios). Dentro de ambos Containers se realizarán las cargas de las copias de seguridad de forma automática para ambos tipos de servicio. Las copias de seguridad tienen una nomenclatura estricta: “nombre del servicio”_”fecha y hora de carga”. Cada una de las copias de seguridad se guardan en formatos ZIP, y vienen acompañadas de dos archivos, un manifiesto en formato XML y un log de sucesos en formato LOG, tal y como se muestra en la Figura 21.

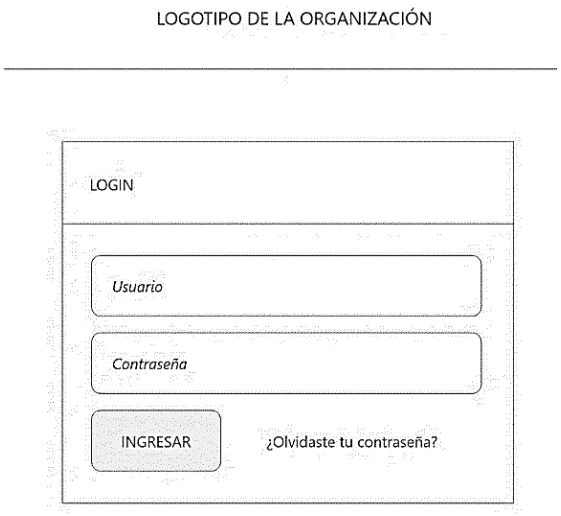
En cuanto a la estructura de los archivos cargados por parte de los postulantes y los generados por el sistema, serán cargados a un Container llamado “documentos”. La estructura de los Blobs dentro de un Container es diferente a la de los sistemas de archivos comunes, organizando su jerarquía en base a su nomenclatura. Dicho esto, la nomenclatura elegida será la siguiente: “Año-Periodo”/”Nombre del programa”/”Nombre del postulante”/”Nombre del archivo”. Uno de los documentos generados y almacenados en este servicio de almacenamiento, el de la solicitud, se puede ver en el Anexo A.5. Documento de Solicitud Generado.

4.2.10. DISEÑO DE INTERFAZ GRÁFICA

La interfaz gráfica para la aplicación debía cumplir el objetivo de ser sencilla en uso y simple, utilizando los colores de la institución. Teniendo esto en mente, se desarrollaron los mockups mostrados en esta sección, los cuales mostrarán la distribución visual de los controles a desarrollar para la solución. Estos diseños fueron elaborados empleando el software Adobe XD CC (Experience Design).

Como se observa en la Figura 22, la interfaz del portal de inicio de sesión es simple y no permite el registro de nuevos usuarios, ya que los mismos se crearán desde el portal de administración.

Figura 22: ***Mockup - Inicio de sesión.***



Fuente: Elaboración propia

En la Figura 23, se muestra el formulario diseñado para el registro de los postulantes, el cual debería dar inicio a todo el proceso de creación y seguimiento de solicitudes. El formulario brinda la posibilidad de realizar la carga masiva de postulantes, en base a un formato específico.

Figura 23: Mockup - Registro de postulantes

Fuente: Elaboración propia

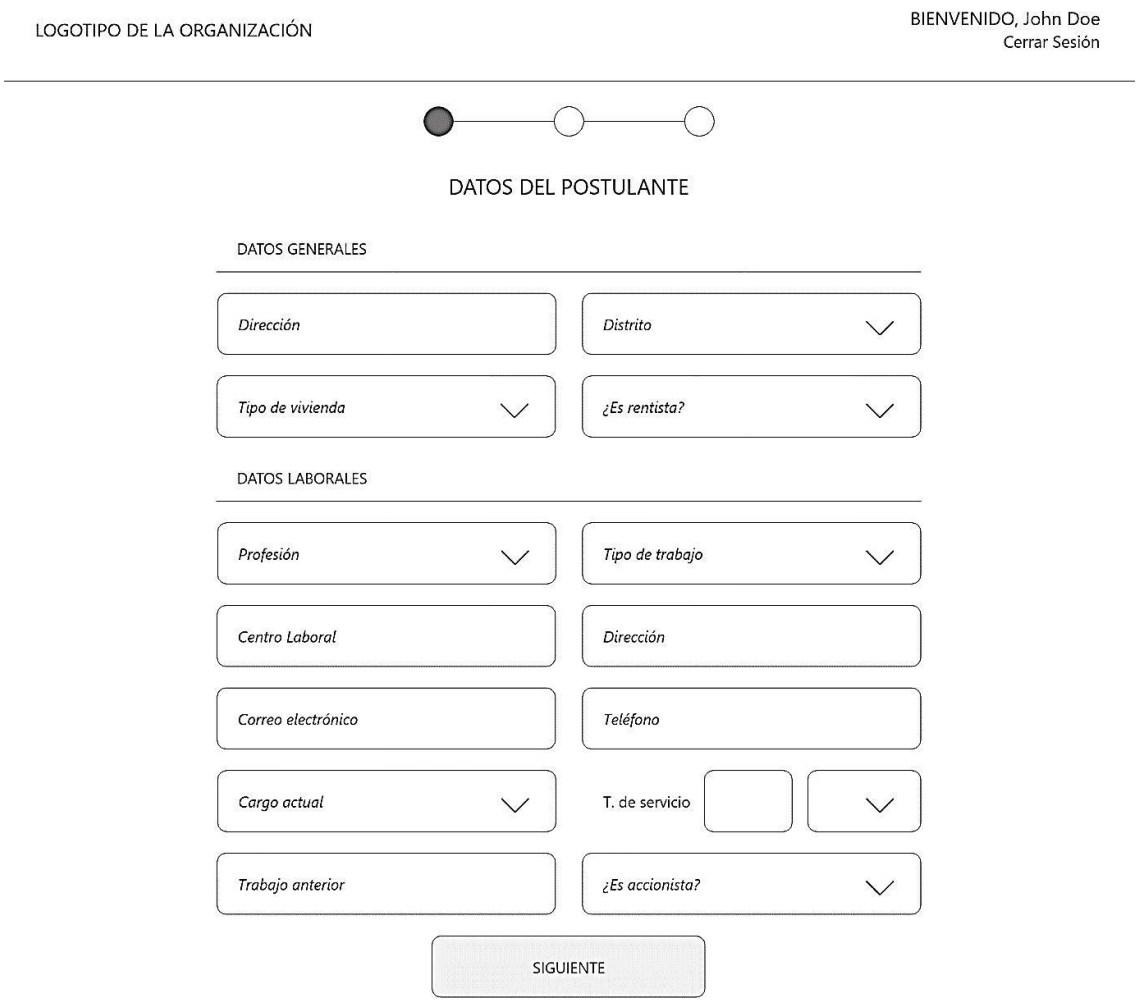
En la Figura 24, se muestra el formulario de carga masiva de postulantes, el cual debe admitir un archivo en formato “.XLSX”.

Figura 24: Mockup - Carga masiva de usuarios

Fuente: Elaboración propia

En la Figura 25, se muestra el diseño del formulario al que tiene acceso el postulante. En este formulario debe finalizar el llenado de información personal y laboral. El formulario completo para esta parte del proceso incluye tres pantallas que incluyen información relacionada a los movimientos de ingresos y egresos del postulante, del cónyuge (en caso de estar casado) y los familiares. Asimismo, se solicita información patrimonial para finalizar con la creación de la solicitud.

Figura 25: Mockup - Datos del postulante

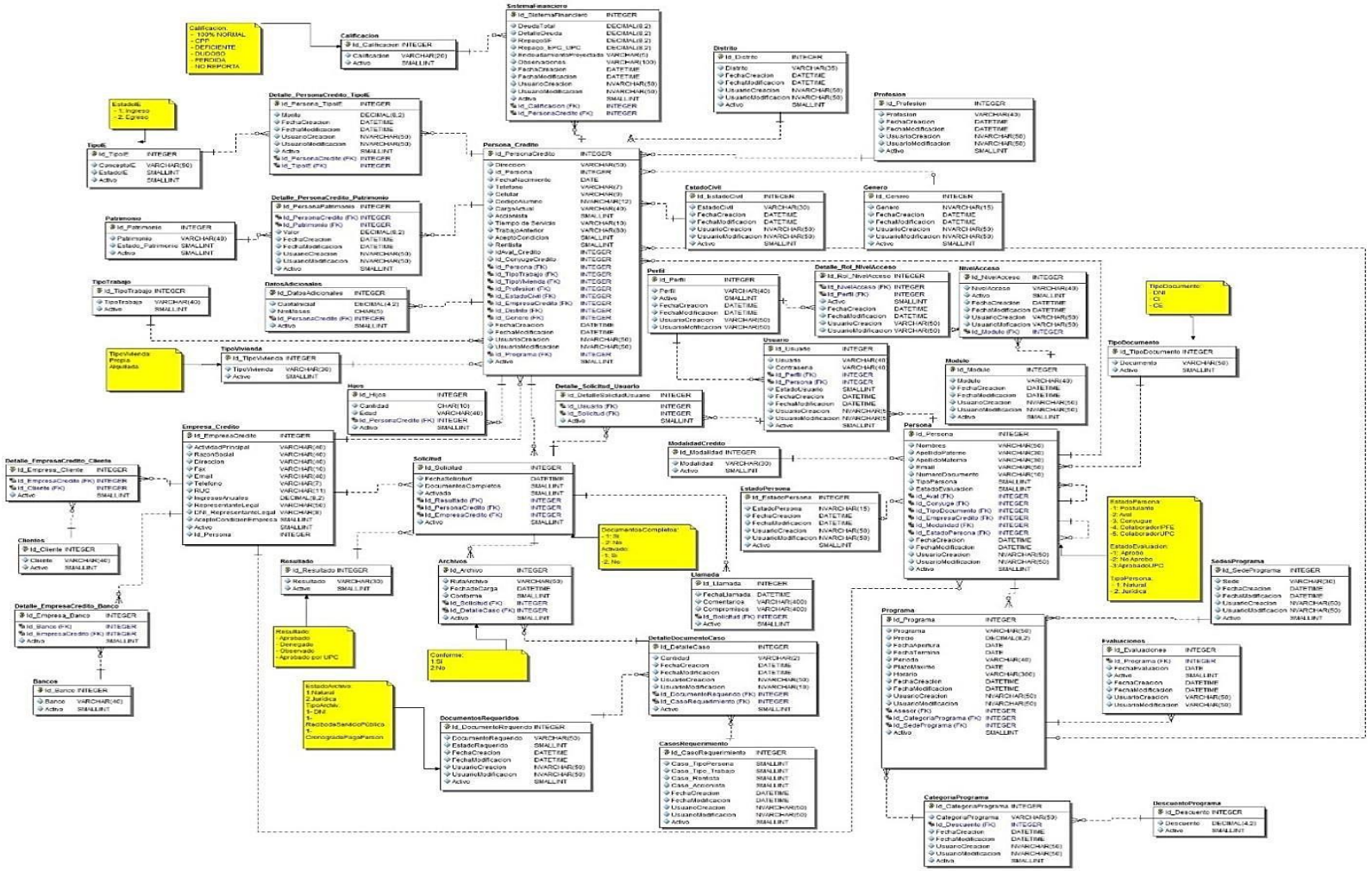


Fuente: Elaboración propia

4.2.11. DISEÑO DE BASE DE DATOS

Se presenta, en la Figura 26, las tablas principales correspondientes al diagrama de base de datos. La implementación lógica de la base de datos se puede apreciar en el Anexo A.4. Base de datos.

Figura 26: Diagrama lógico de base de datos



Fuente: Elaboración propia

CAPÍTULO V
CONSTRUCCIÓN

En este último capítulo se realiza un resumen de las tecnologías empleadas, las configuraciones aplicadas y breves capturas relevantes para el desarrollo de la solución. Asimismo, se describen la estrategia de pruebas y los resultados obtenidos de las mismas.

5.1. CONSTRUCCIÓN

5.1.1. FRAMEWORKS, LIBRERÍAS Y TECNOLOGÍAS

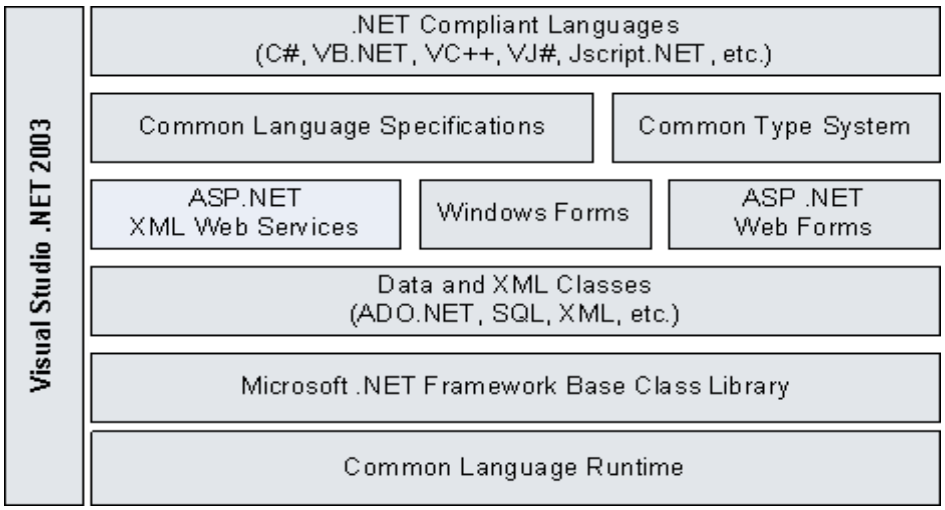
5.1.1.1. Framework De Desarrollo

En este proyecto se ha utilizado .NET Framework en su versión 4.5, que es la plataforma de desarrollo de Microsoft para la construcción de aplicaciones para entornos web, móviles, de escritorio, Internet Of Things (IoT) y sistemas conectados a la nube (20). Básicamente está compuesto por el Common Language Runtime (CLR) y el Framework Class Library (FCL). El primero de ellos es un entorno encargado de ejecutar y controlar las instrucciones de los programas escritos para la plataforma .NET; el segundo es una librería de clases, interfaces comunes y tipos de datos, que provee acceso a funcionalidad específica del sistema. La arquitectura de .NET Framework se puede ver en la Figura 27.

Entre otras, las características principales de .NET Framework son:

- Gestión de memoria. A diferencia de otros lenguajes de programación, aquellos que corran bajo .NET Framework contarán con la gestión de memoria por parte del CLR, quitando la responsabilidad a los desarrolladores de ubicar y liberar instrucciones y data en memoria.
- Sistema de tipos común. A diferencia de los lenguajes de programación tradicionales, en los que los tipos son definidos por el compilador, en .NET Framework, los tipos básicos son definidos por el propio .NET Framework, lo que facilita la interoperabilidad de los lenguajes que lo utilizan.
- Amplia biblioteca de clases. Los programadores tienen a disposición una librería con acciones y operaciones comunes de bajo nivel, lista para ser reutilizada en todo momento.
- Tecnologías de desarrollo. .NET posee bibliotecas específicas para los diferentes tipos de aplicaciones disponibles, como: ASP.NET para desarrollo web, pasando por ADO.NET para el acceso y escritura de datos, WCF para aplicaciones orientadas a servicios y Windows Presentation Foundation (WPF) para aplicaciones desktop.
- Interoperabilidad de lenguajes de programación. Los lenguajes que son compilados aprovechando el CLR de .NET Framework arrojan un código intermedio, Common Intermediate Language (CIL), el que se compila en tiempo de ejecución gracias al CLR.

Figura 27: Arquitectura *de .NET* Framework



Fuente: Microsoft Corporation, 2006 (21)

La elección de este framework para el desarrollo de la solución se justifica debido a la fuerte integración que posee con las herramientas empleadas, su soporte nativo para el servicio de nube de Microsoft Azure y la velocidad de desarrollo de la solución. Sumado a esto, se toma en cuenta que posee una curva de aprendizaje menor frente a otras tecnologías de desarrollo web, disminuyendo el tiempo empleado en el desarrollo de la solución.

Se hará uso de ASP.NET para la construcción del servicio de WCF, así como para el desarrollo de la aplicación web con MVC. Es importante tener en cuenta que los servicios desarrollados con ASP.NET poseen compatibilidad con otros sistemas debido al uso de XML y JSON como lenguajes de marcado para el intercambio de información. Asimismo,las aplicaciones web desarrolladas con ASP.NET son soportadas por los navegadores actuales.

5.1.1.2. Lenguajes De Programación

Para el desarrollo integral de la aplicación se emplearon dos lenguajes de programación. Para la capa de presentación se empleó JavaScript, con la

librería Knockout, debido a la flexibilidad que le brinda a las vistas de manejar la data recibida desde el servicio, permitiendo realizar cambios directamente en la misma sin necesidad de realizar llamadas al servicio o al aplicativo principal con cada modificación; esto fue especialmente útil para la construcción de interfaces más complejas, como la encargada de elaborar los cronogramas de pagos, que modifica los datos de forma dinámica conforme se cambien los valores manejados por la tabla en la que se muestran los datos. Por otro lado, si bien es cierto que .NET Framework permite el uso de un amplio grupo de lenguajes de programación, el tesista eligió C# como lenguaje principal por los siguientes motivos:

- Los patrones empleados en el desarrollo de la solución necesitan del uso de un lenguaje orientado a objetos de forma estricta.
- C# es un lenguaje que brinda capacidades robustas y maduras en cuanto a sintaxis, tipado y estructura de código. Sumado a esto, respeta principios de la programación orientada a objetos, tales como encapsulamiento, abstracción, herencia, sobrecarga y polimorfismo de forma más natural y avanzada que otros lenguajes de la misma familia, como Visual Basic.
- Las tecnologías, componentes y librerías de código que se integraron a la solución otorgan un mejor desempeño con soluciones desarrolladas en C# (como Enterprise Library y las librerías de comunicación con Azure Storage).
- El control de las excepciones manejado por C# es altamente estructurado.
- El uso de C# exige la declaración explícita de las variables empleadas a diferencia de otros lenguajes de programación, lo cual brinda un código fuertemente tipado. Omitir este punto dificultaría la ejecución de las pruebas que se busquen realizar en este proyecto o en futuros desarrollos, así como a la limpieza del código y al uso de buenas prácticas de programación.

5.1.2. DATA ACCESS BLOCK (DAB) VS OBJECT-RELATIONAL MAPPING (ORM)

Para la comunicación con la base de datos, el acceso, manejo y guardado de información, se contaba con dos opciones. La primera consideraba el uso de Entity Framework (EF) para el mapeo de las entidades de la base de datos de forma sencilla, el cual es un ORM completo; la segunda consideraba el uso de Enterprise Library (EL), la cual es una librería de tipo Data Access Application Block (DAAB) que requiere un poco más de codificación debido a que no realiza un mapeado directo de las entidades presentes en a base de datos.

Pese a que Enterprise Library requiere que se repliquen las clases correspondientes a las entidades de la base de datos de forma manual y no ha recibido nuevas actualizaciones desde su versión 6.0, fue la opción elegida debido a la facilidad que brinda con el manejo de los procedimientos almacenados frente a Entity Framework y a que todavía sigue funcionando de forma bastante robusta, siendo una librería madura para este tipo de tareas. Para poder realizar cambios en las consultas realizadas a la base de datos, se manejaron todas las consultas a nivel de procedimientos almacenados. Además, esto facilita la realización de cambios en los mismos, viéndose reflejados de forma inmediata en la aplicación, sin necesidad de generar cambios a nivel de código de la misma, evitando así el proceso de re compilación y despliegue continuo. Fue por esta razón que se optó por elegir EL frente a EF.

5.1.2.1. IDE

El entorno de desarrollo propuesto por excelencia para este proyecto fue el Integrated Development Environment (IDE) Microsoft Visual Studio 2015 Ultimate. Si bien es cierto, Microsoft Visual Studio posee versiones no comerciales orientadas a equipos pequeños, el equipo contaba con licencias para esta versión del mismo.

La elección de Microsoft Visual Studio como IDE de desarrollo se basó en:

- Soporte nativo para el desarrollo de software orientado a objetos con ASP.NET, además de integrar a C# como lenguaje de programación principal desde .NET Framework 4.0.
- Visual Studio posee herramientas, comandos y funciones integradas para realizar un desarrollo óptimo en todo tipo de tecnologías Microsoft, desde aplicaciones de escritorio hasta servicios y aplicativos móviles conectados a la nube de Microsoft Azure.
- Integración con servicios como Visual Studio Team Services, herramienta empleada en la gestión de las actividades, creación de casos de tareas y sprints en este proyecto. Posee editores compatibles con los estándares CSS 3, JavaScript, JSON y XML, además de facilitar la creación de vistas en HTML.
- Posee un motor potente para validación de diferentes lenguajes, lo que permitió realizar una depuración correcta del código desarrollado.
- A nivel de despliegue, Visual Studio ofrece la facilidad de integrar VSTS para configurar el despliegue continuo tras cada compilación de la solución, indicando los errores y el miembro del equipo que provocó los mismos. En caso de compilar con éxito una versión, realiza el despliegue automático a los entornos configurados, evitando pérdidas y errores.
- Desde la misma IDE de Visual Studio, se puede conectar el servicio de nube de Microsoft Azure para establecer las conexiones y realizar modificaciones de configuración y datos a los servicios de Azure SQL DB y Azure Storage, ambos empleados en el desarrollo de este sistema.

5.1.2.2. Base De Datos

La base de datos elegida fue, sin reparos, Azure SQL DB, por las siguientes razones:

Los costos de licenciamiento son bajos y se encuentran incluidos en la contratación del servicio.

Azure SQL DB no requiere el monitoreo constante del servidor, al ser ofrecido como PaaS, siendo posible dedicar más tiempo al desarrollo de la solución en lugar de dedicarlo al mantenimiento de servidores, sistema operativo o al motor de la base de datos.

Elimina la carga de realizar actualizaciones, asegurar la alta disponibilidad o realizar las copias de seguridad.

Posee soporte para la mayoría de las funciones de SQL Server.

Posee funciones integradas que contemplan la alta tolerancia a fallos, geo replicación, recuperación ante desastres y permiten el filtrado por direcciones IP para bloqueo de accesos no autorizados.

Puede ser consultada desde el IDE de Visual Studio y desde SQL Server Management Studio

5.1.2.3. Otras Librerías

Durante el desarrollo de la solución fue necesario integrar librerías que realicen la conversión de texto plano, que contenía código en HTML, a un documento en formato PDF, lo cual se logró mediante la importación de las librerías:

- Syncfusion.WebKitHtmlConverter.Base.dll
- Syncfusion.Pdf.Base.dll

Se eligió el paquete de librería de Syncfusion debido a que cuentan con abundante documentación y un soporte completo por parte del equipo de Syncfusion. Entre las ventajas que brinda este conjunto de librerías se encuentran:

- Existe una versión gratuita Community con las librerías completas para ser utilizadas.

- Convierte cualquier cadena de texto que contenga HTML a formato PDF.
- Funciona en entornos de 32 bits y 64 bits.
- Soporta la creación de cabeceras y pies de página.
- Soporta HTML5 y CSS3.
- Permite establecer las propiedades del documento, configuraciones de página, seguridad, preferencias de visualización y más.

Una de las preocupaciones principales al momento de buscar una librería para realizar este tipo de conversión, fue la de la creación de las tablas en los espacios indicados dentro del documento HTML, ya que, por la extensión, era probable que se presentara un salto de página sin haber finalizado la impresión de las tablas. Las librerías de Syncfusion nos permitieron mitigar este riesgo de la forma más sencilla posible.

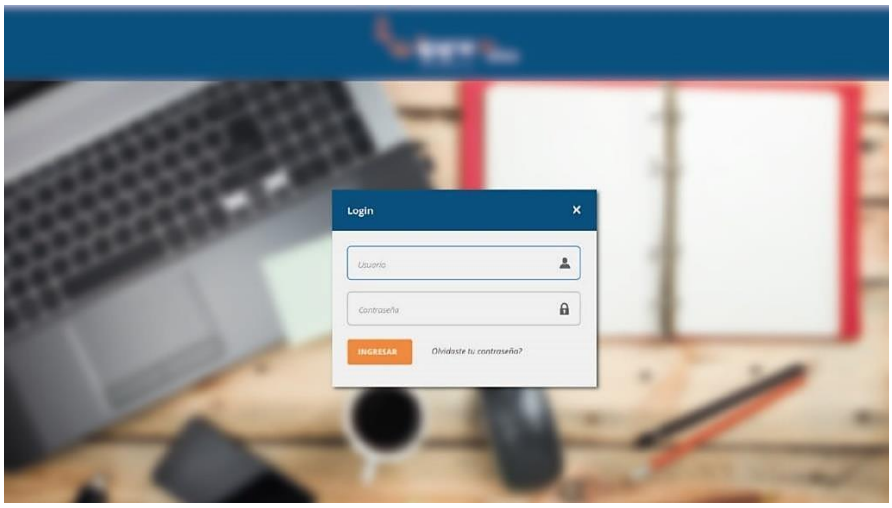
La lista de los proyectos desarrollados en la solución se muestra en el Anexo A.1 Estructura De La Solución.

5.1.3. INTERFACES GRÁFICAS

Tras la elaboración de los mockups, se continuó con la elaboración de las vistas del sistema, las cuales fueron desarrolladas empleando HTML5, CSS3 y JavaScript, este último con la librería Knockout.JS. Hay que tener en cuenta que, tratándose del desarrollo para una ONG en un ambiente real de desarrollo, se ocultará la identidad de la misma, así como los textos o valores reales utilizados por la organización, cambiándolos por valores de prueba y dejando los más importantes para el caso de estudio.

En la Figura 28 se muestra el formulario de inicio de sesión, el cual espera los valores entregados al postulante vía correo electrónico. Del mismo modo, el formulario validará si el usuario existe en la lista de usuarios de la organización y si poseen el estado “Activo”.

Figura 28: Formulario de Inicio de sesión



Fuente: Elaboración propia

El formulario de registro de postulante puede verse en la Figura 30, el cual tuvo cambios ligeros con respecto a los mockups realizados, pero que es esencial para el registro de la solicitud y la vinculación de la lista de documentos requeridos. A este formulario le siguen los de “Movimientos” y “Patrimonios”

Figura 29: Formulario de Registro de Postulante

Fuente: Elaboración propia

En la Figura 30 se muestra el formulario de carga de documentos, al que puede acceder el estudiante. Este formulario muestra la lista de los documentos solicitados al postulante, los cuales se generan en base a la información brindada en los formularios de registro de datos generales, que se detallarán en las siguientes líneas.

Figura 30: Formulario de Carga de documentos

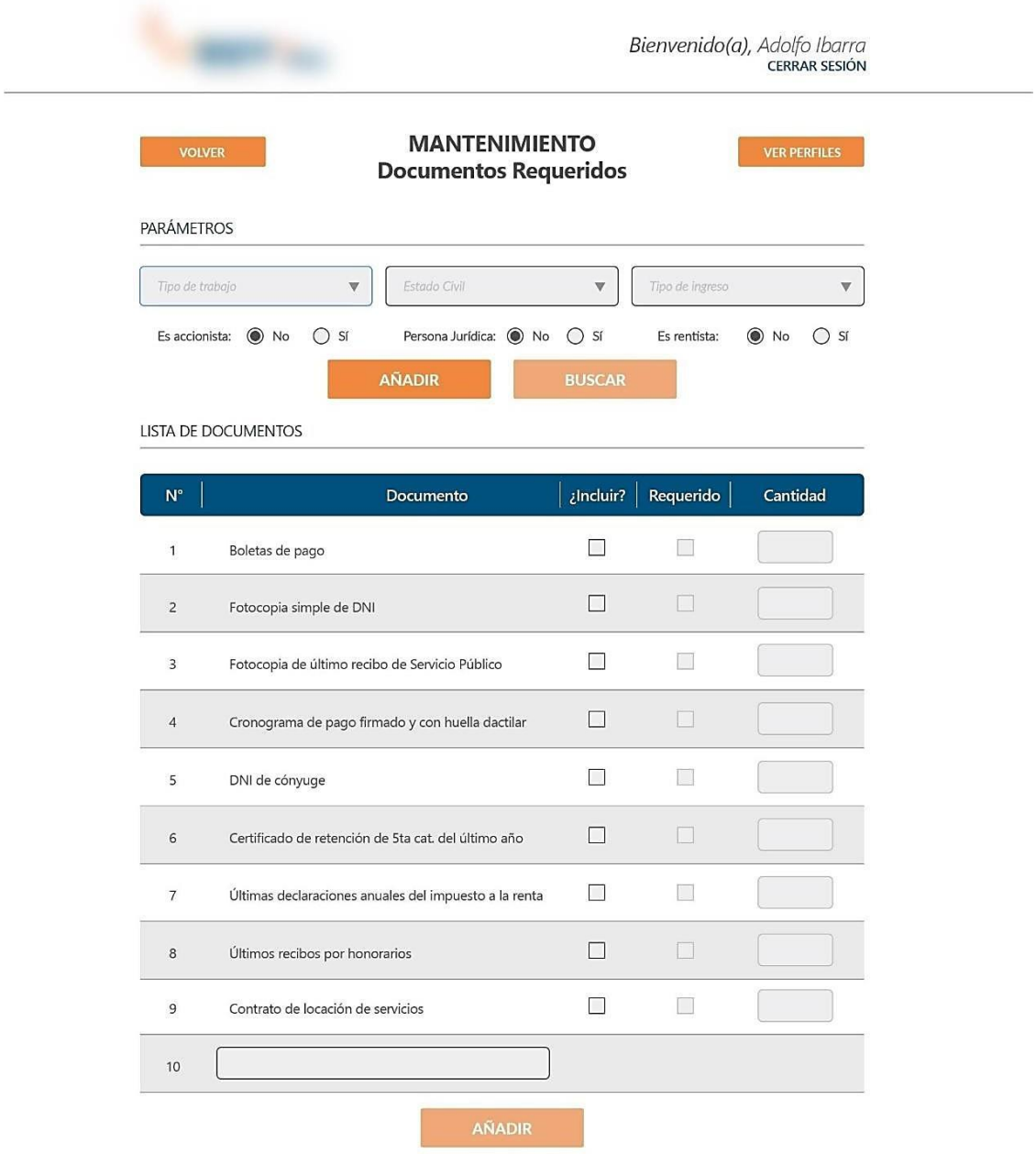
Fuente: Elaboración propia

En la Figura 31Figura 30 se muestra el formulario de mantenimientos para los documentos requeridos, perteneciente al perfil de “Administrador”.

En este formulario existen dos secciones, “Parámetros” y “Lista de documentos”. En la primera sección, se muestran los controles que permiten realizar el registro y la búsqueda de los documentos en base a los parámetros específicos para un caso determinado; esto se refiere a que la combinación de los parámetros introducidos por un estudiante al momento del registro, le asignará un listado de documentos requeridos basándose en la información introducida en este formulario. Cuando se presiona el botón “Buscar”, se muestran los documentos asociados a ese caso específico, permitiendo modificarlos en la tabla inferior.

En la segunda sección se muestran los documentos registrados y disponibles para ser asociados a las posibles combinaciones de parámetros. Además, esta segunda sección permite registrar nuevos documentos con el textbox que se encuentra al final de la misma.

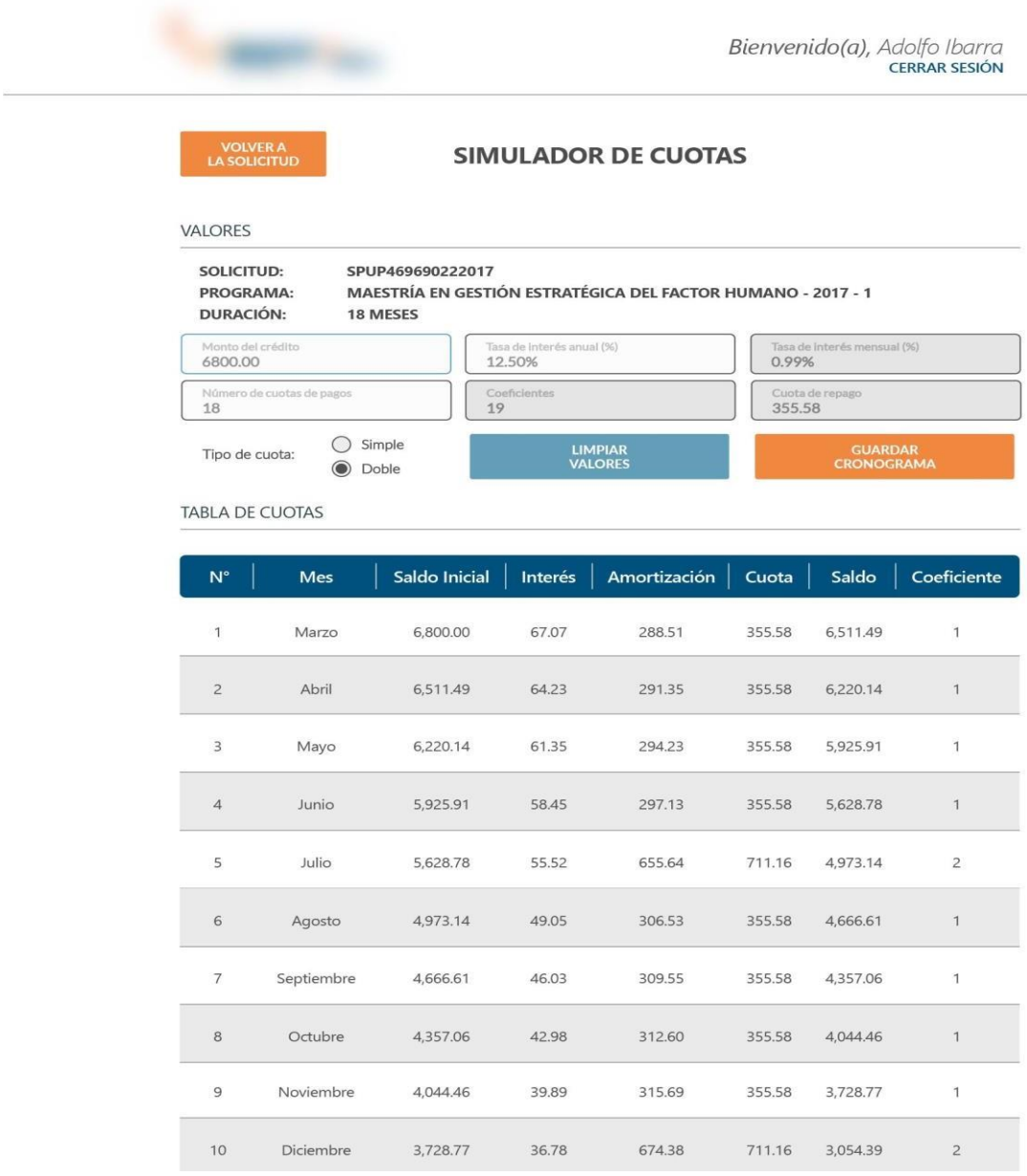
Figura 31: Formulario de Mantenimiento - Documentos requeridos



Fuente: Elaboración propia

En la Figura 32Figura 30 se presenta el formulario creado para el simulador de cuotas, el cual es particularmente diferente a los demás debido a que hace uso intensivo de la librería Knockout.JS, implementando algunas clases que se verán en la sección correspondiente, más adelante en este mismo documento. A este formulario se accede desde el detalle de cada solicitud para continuar con el proceso de evaluación. El botón “Guardar cronograma” guarda una copia de la tabla en formato PDF adjunto a la lista de los documentos del postulante y asigna las cuotas correspondientes a su solicitud.

Figura 32: Formulario de Simulador de cuotas



N°	Mes	Saldo Inicial	Interés	Amortización	Cuota	Saldo	Coeficiente
1	Marzo	6,800.00	67.07	288.51	355.58	6,511.49	1
2	Abril	6,511.49	64.23	291.35	355.58	6,220.14	1
3	Mayo	6,220.14	61.35	294.23	355.58	5,925.91	1
4	Junio	5,925.91	58.45	297.13	355.58	5,628.78	1
5	Julio	5,628.78	55.52	655.64	711.16	4,973.14	2
6	Agosto	4,973.14	49.05	306.53	355.58	4,666.61	1
7	Septiembre	4,666.61	46.03	309.55	355.58	4,357.06	1
8	Octubre	4,357.06	42.98	312.60	355.58	4,044.46	1
9	Noviembre	4,044.46	39.89	315.69	355.58	3,728.77	1
10	Diciembre	3,728.77	36.78	674.38	711.16	3,054.39	2

Fuente: Elaboración propia

El uso de la librería Knockout.JS, especialmente en el formulario de la Figura 32, permite que los cambios realizados en los campos de textos habilitados para modificación reflejen los resultados de las operaciones manejadas por los métodos implementados para trabajar con las clases correspondientes. En la Figura 33, se

muestra una porción del código que existe a nivel de implementación para la Vista-Modelo de dicho formulario. En la Figura 34, se muestra una porción de código HTML con las etiquetas de Knockout.JS, las cuales permiten realizar la unión de las clases y el resultado de los métodos implementados bajo esta librería con los valores a ser mostrados por el navegador en HTML. Asimismo, se muestran las referencias a los ficheros .JS que contienen las Vistas- Modelos.

Figura 33: Pantalla de Visual Studio, mostrando una Vista-Modelo: Método GuardarPrograma en Knockout.JS

```
GuardarPrograma: function () {
    $.ajax({
        url: '/SimuladorCuotas/GenerarPdf',
        dataType: 'json',
        type: 'POST',
        data: {
            cuotas: SimuladorVM.Payments(),
            persona: SimuladorVM.Persona(),
            parameters: new Parameters({
                PagoInicial: SimuladorVM.PagoInicial(),
                Monto: SimuladorVM.Monto(),
                SaldoAfinanciar: SimuladorVM.SaldoAFinanciar(),
                CostoProgDescuento: SimuladorVM.CostoProgDescuento(),
                PorcDescuento: SimuladorVM.PorcDescuento(),
                CuotaSimple: SimuladorVM.CuotaSimple().toFixed(2),
                TasaAnual: SimuladorVM.TasaAnual(),
                TasaMensual: SimuladorVM.TasaMensual(),
                NumeroCuotas: SimuladorVM.NumeroCuotas(),
                Coeficientes: SimuladorVM.Coeficientes().toFixed(2),
                CuotaRepago: SimuladorVM.CuotaRepago().toFixed(2),
                AnoActual: SimuladorVM.Persona().Programa().FechaApertura().getFullYear(),
            }),
            idSolicitud: parseInt($("#solicitud").val())
        },
        async: false,
        success: function (data) {
            if (data.fileName) {
                window.open("/SimuladorCuotas/DescargarCronograma?filename=" + data.fileName);
            }
        },
        error: function () {
            alert('Esta operación no puede ser procesada en este momento. Por favor intente más tarde. Si el problema persiste contacte con
        }
    });
},
```

Fuente: Elaboración propia

Figura 34: Pantalla de Visual Studio, mostrando una Vista: Método GuardarPrograma invocado desde botón

```
                        </div>
                        <div class="controls-primary">
                            <button type="button" class="btn btn-primary " data-bind="click: $root.GuardarPrograma">
                                GUARDAR CRONOGRAMA
                            </button>
                        </div>
                    </div>
                </div>
            </div>
        </div>
    </form>
</div>

@section scripts{
    <script src="@Url.Content("~/ViewModels/Utils.js")"></script>
    <script src="@Url.Content("~/ViewModels/SimuladorVM/Models.js")"></script>
    <script src="@Url.Content("~/ViewModels/SimuladorVM/SimuladorVM.js")"></script>
}
```

Fuente: Elaboración propia

5.1.3.1. Interfaces Desarrolladas

Durante el desarrollo del sistema, se desarrollaron las interfaces gráficas mostradas en la Tabla 39, las cuales atienden a los requerimientos allí mencionados.

Tabla 39: Listado de requerimientos atentidos por las interfaces desarrolladas

INTERFAZ	CARPETA	REQUERIMIENTOS
Index.cshtml	/Login	**RF-01**
CambioContrasena.cshtml	/Login	**RF-01**
Index.cshtml	/Evaluacion	**RF-16, RF-17**
_DatosPatrimoniales.cshtml	/Evaluacion	**RF-17, RF-21**
_DocumentosRequeridos.cshtml	/Evaluacion	**RF-23**
_InformacionPersonal.cshtml	/Evaluacion	**RF-16, RF-21**
_MovimientosRP.cshtml	/Evaluacion	**RF-17, RF-21**
_RegistroPostulante.cshtml	/Evaluacion	**RF-16, RF-21**
_SistemaFinanciero.cshtml	/Evaluacion	**RF-17, RF-21**
_RegistroLlamadas.cshtml	/Evaluacion	**RF-15**
Index.cshtml	/GeneracionUsuarios	**RF-02**
Index.cshtml	/Jefatura	**RF-21**
_SolicitudesAprobadas.cshtml	/Jefatura	**RF-21**
_SolicitudesConfirmadas.cshtml	/Jefatura	**RF-21**
_SolicitudesRechazadas.cshtml	/Jefatura	**RF-21**
_SolicitudesRechazadasFinalizacion.cshtml	/Jefatura	**RF-21**
Index.cshtml	/Mantenimiento	**RF-09**
_Cargos.cshtml	/Mantenimiento	**RF-09**
_CategoriaPrograma.cshtml	/Mantenimiento	**RF-09**

_DocumentosRequeridos.cshtml	/Mantenimiento	**RF-24**
_Perfil.cshtml	/Mantenimiento	**RF-08**
_Persona.cshtml	/Mantenimiento	**RF-09**
_Profesion.cshtml	/Mantenimiento	**RF-09**
_Programa.cshtml	/Mantenimiento	**RF-06, RF-20**
_Sede.cshtml	/Mantenimiento	**RF-09**
_Usuario.cshtml	/Mantenimiento	**RF-07**
_UsuarioMasivo.cshtml	/Mantenimiento	**RF-07**
Index.cshtml	/Pendientes	**RF-03**
CronogramaPagos.html	/PlantillaReporte	**RF-19**
CronogramaPagosTabla.html	/PlantillaReporte	**RF-19**
CronogramaPagosTablaFila.html	/PlantillaReporte	**RF-19**
EvaluacionCrediticia.html	/PlantillaReporte	**RF-18**
EvaluadasCoordinador.html	/PlantillaReporte	**RF-18**
EvaluadasFilaCompleta.html	/PlantillaReporte	**RF-18**
FacturacionMensual.html	/PlantillaReporte	**RF-25**
FacturacionFilaCompleta.html	/PlantillaReporte	**RF-25**
Solicitud.html	/PlantillaReporte	**RF-25**
TableBodyEvaluacion.html	/PlantillaReporte	**RF-18**
TableBodyFacturacion.html	/PlantillaReporte	**RF-25**
Index.cshtml	/PreRegistro	**RF-02**
_PreRegistroMasivo.cshtml	/PreRegistro	**RF-02**
Index.cshtml	/RegistroPostulante	**RF-10**
_DatosPatrimoniales.cshtml	/RegistroPostulante	**RF-13, RF-14**
_DocumentosRequeridos.cshtml	/RegistroPostulante	**RF-22**
_EnviarRP.cshtml	/RegistroPostulante	**RF-10**
_MovimientosRP.cshtml	/RegistroPostulante	**RF-13, RF-14**
_RegistroConyuge.cshtml	/RegistroPostulante	**RF-10**

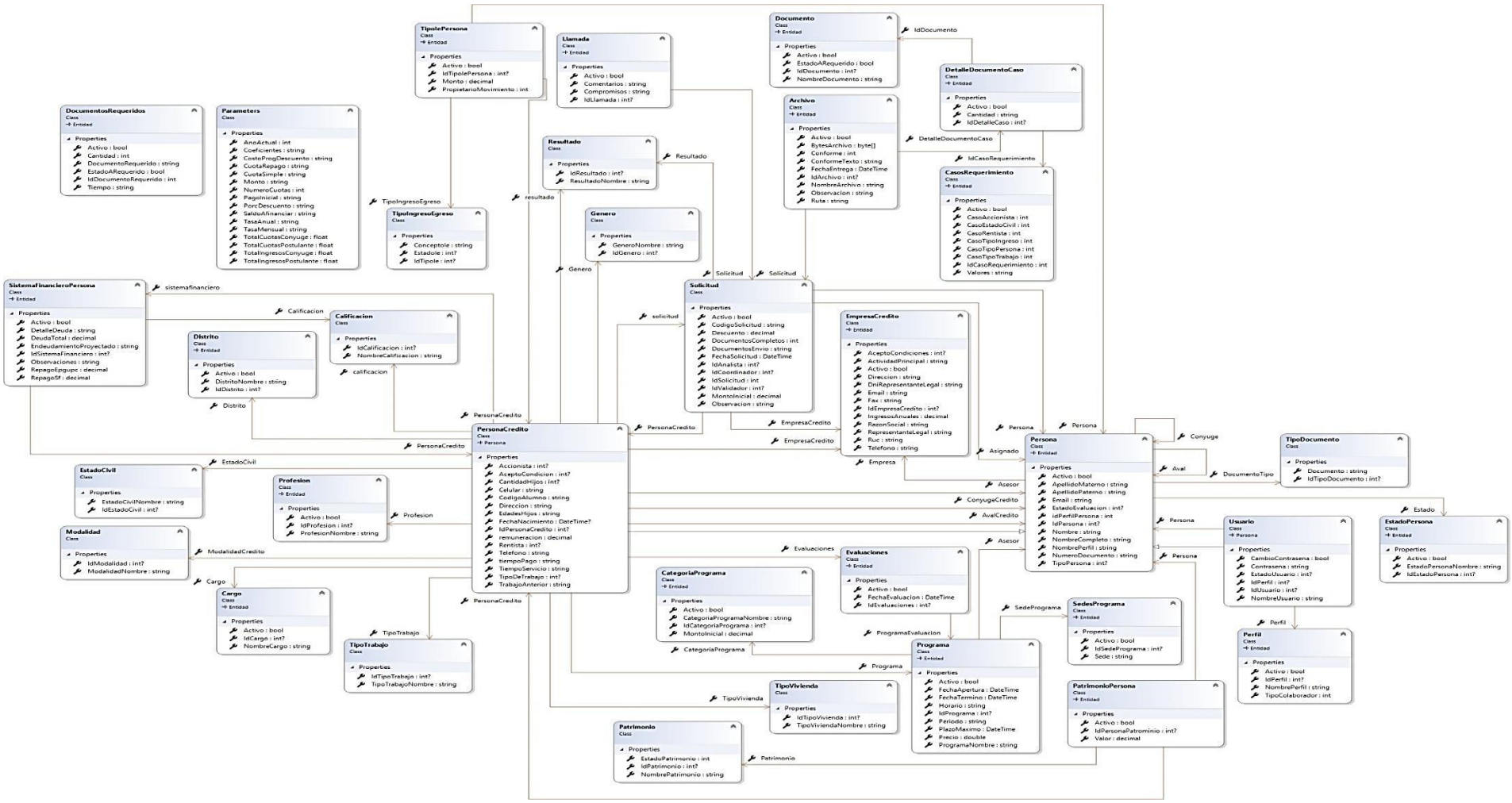
Index.cshtml	/SimuladorCuotas	**RF-19**
Index.cshtml	/Asignacion	**RF-11**
_Analizadas.cshtml	/Asignacion	**RF-11**
_Validadas.cshtml	/Asignacion	**RF-11**

Fuente: Elaboración propia

5.1.4. CLASES E INTERFACES

En esta sección se mostrarán algunas clases principales en el desarrollo del sistema. No todas las clases presentadas en esta sección se encuentran en el diagrama de clases que contempla los objetos con los que se trabajó para la representación de las entidades de la base de datos, el cual se muestra en la Figura 35. Sino que también se muestran algunas clases que realizan operaciones imprescindibles para el funcionamiento del sistema y que no representan entidades, tales como las de la comunicación con el Azure Storage, el envío de correo electrónico, entre otros. La lista de las clases implementadas, así como el detalle de las mismas, se puede encontrar en el anexo A.3. Clases e Interfaces.

Figura 35: Diagrama de clases completo



Fuente: Elaboración propia

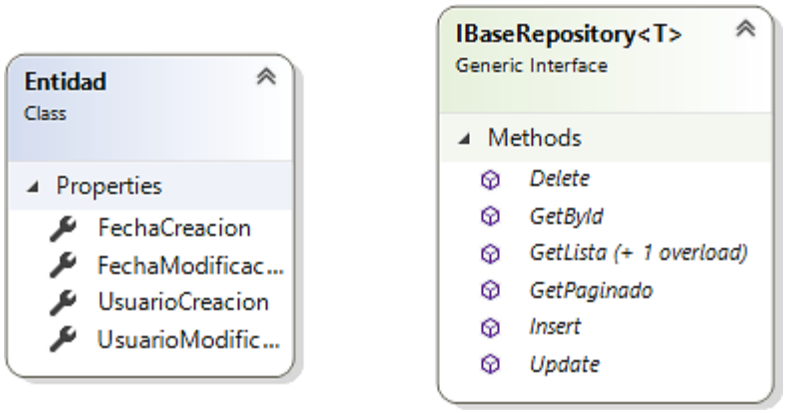
5.1.4.1. Entidades Del Sistema

Para el desarrollo de este sistema, se ha realizado un desacoplamiento fuerte para las clases, separando sus métodos de sus propiedades. Esto se realizó con el objetivo de poder aplicar adecuadamente principios de S.O.L.I.D.

Teniendo esto en mente, se muestran las clases más importantes que representan a las entidades del sistema:

Clase “Entidad” e interfaz IBaseRepository<T>. Es la clase de la que heredan todas las otras clases, debido a que cuenta con las propiedades destinadas a servir objetivos de auditoría interna. Asimismo, se ha creado una interfaz denominada IBaseRepository, la cual hereda sus métodos a todas las otras interfaces que representan una entidad dentro del diagrama físico de la base de datos y son métodos genéricos. La clase, con sus propiedades, y la interfaz, con sus métodos, se observan en la Figura 36.

Figura 36: Clase "Entidad" e interfaz "IBaseRepository", con sus propiedades y métodos



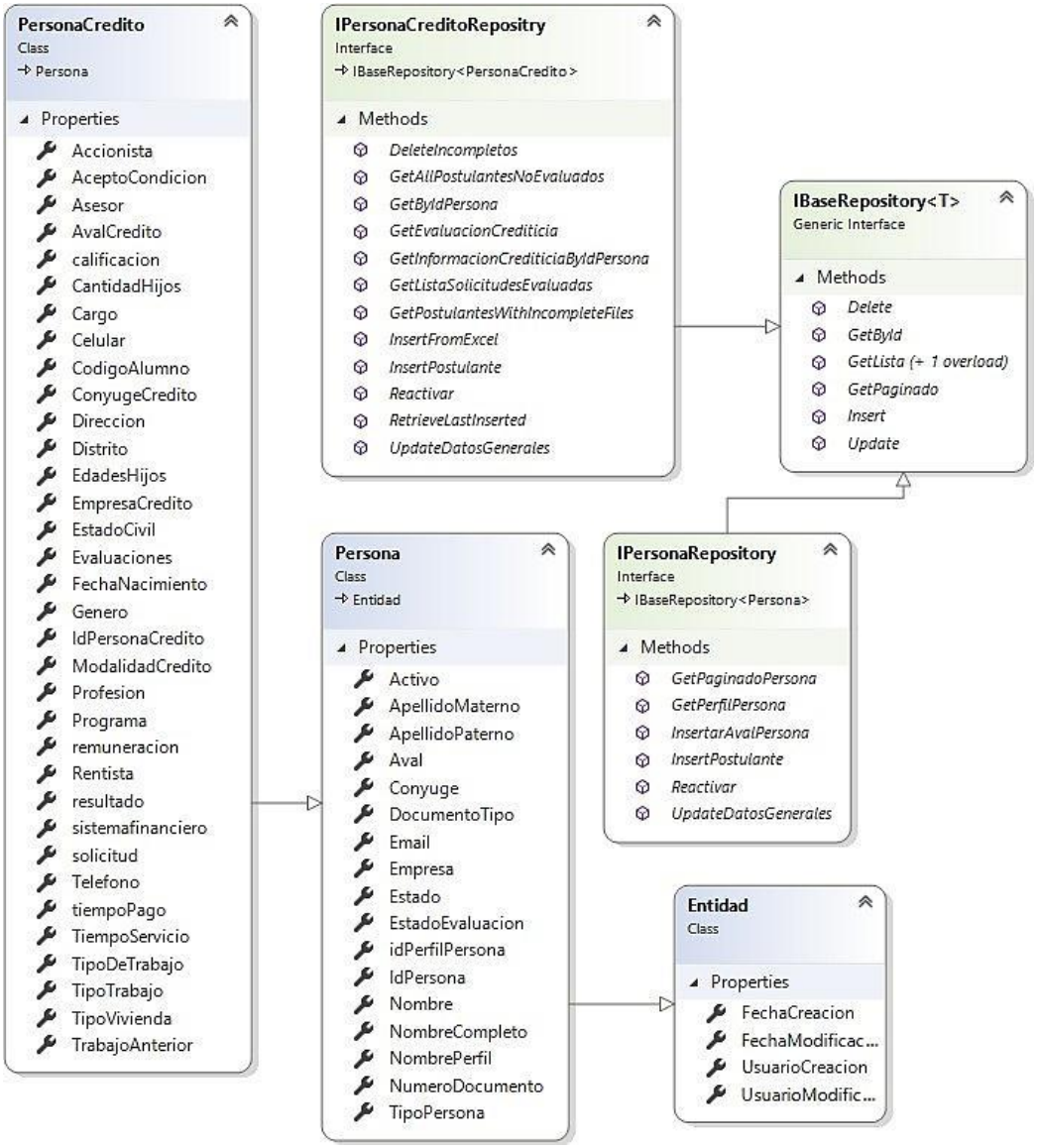
Fuente: Elaboración propia

Clases Persona y Persona Credito e interfaces IPersonaRepository y IPersonaCreditoRepository. La clase PersonaCredito hereda de Persona, y contiene todas las propiedades necesarias para relacionar una a una persona con una solicitud. Además, registra los parámetros adecuados

para establecer los documentos que los solicitantes necesitarán enviar. La clase Persona, además, almacena información concerniente a los colaboradores de la ONG.

La relación que se da entre ambas clases y la clase Entidad, es que Persona hereda de Entidad y, por consiguiente, lo hace Persona Credito al heredar de Persona. En el caso de las interfaces, la relación es diferente debido a las operaciones que son necesarias para cada una de estas clases, no siendo necesaria la herencia de métodos, pero sí de propiedades. Ambas interfaces heredan directamente de IBaseRepository. Estas relaciones se pueden ver en la Figura 37.

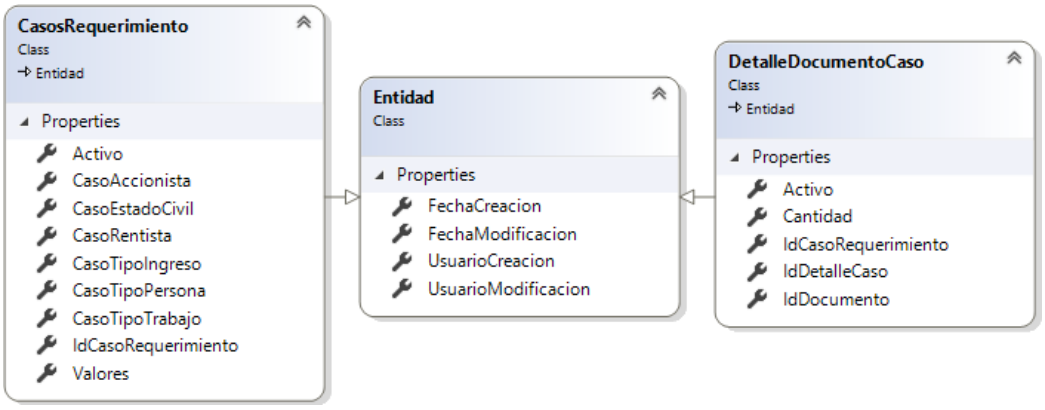
Figura 37: Clases Persona y PersonaCredito e interfaces PersonaRepository y PersonaCreditoRepository.



Fuente: Elaboración propia

Clases CasosRequerimiento y DetalleDocumentoCaso. Ambas clases heredan los métodos declarados en sus interfaces correspondientes, pero resultan vitales para el desarrollo del sistema debido a que brindan los datos necesarios para realizar la asignación de los documentos a los postulantes en base a los parámetros registrados por ellos. La Figura 38 nos muestra ambas clases, las cuales heredan de la clase base Entidad.

Figura 38: Clases CasosRequerimiento y DetalleDocumentoCaso

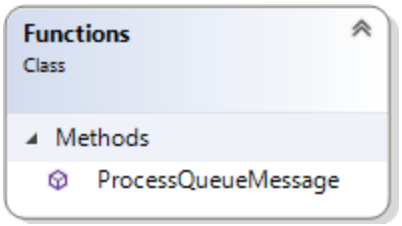


Fuente: Elaboración propia

5.1.4.2. Clases De Funciones Principales Del Servicio

Clase "Functions". El proyecto destinado a hacer uso de las funciones programadas en los servicios CronJob de Microsoft Azure, implementa una clase que se comunica con el servicio web. Esta clase posee un único método, el cual se encarga de hacer la limpieza de las solicitudes que se encuentra fuera del plazo, actualizando sus valores. Asimismo, realiza el envío de las alertas y notificaciones por mensajes de texto y correo electrónico. La Figura 39 muestra la clase Functions. Su implementación se podrá ver en la sección de Anexos (Anexo A.3. Clases e Interfaces).

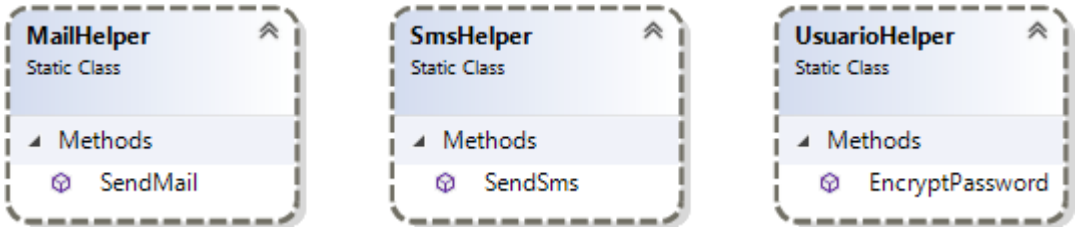
Figura 39: Clase Functions



Fuente: Elaboración propia

Clases "Helper". Las clases que se encuentran en esta definición son MailHelper, SmsHelper y UsuarioHelper. Estas clases brindan una única función específica. MailHelper realiza el envío de correos electrónicos por SMTP para todo el sistema, recibiendo parámetros que detallan el tipo de correo, el cuerpo del mensaje y los valores de la solicitud asociada. Por su parte, SmsHelper, recibe los datos del tipo de mensaje de texto y del postulante para realizar los envíos al comunicarse con las APIs de Twilio, utilizando los tokens de configuración especificados en el archivo Web.Config del servicio. La Figura 40 muestra las clases mencionadas. Finalmente, UsuarioHelper implementa un método encargado de encriptar las contraseñas antes de ser almacenadas en la base de datos. La imagen muestra las clases mencionadas.

Figura 40: Clases MailHelper, SmsHelper y UsuarioHelper



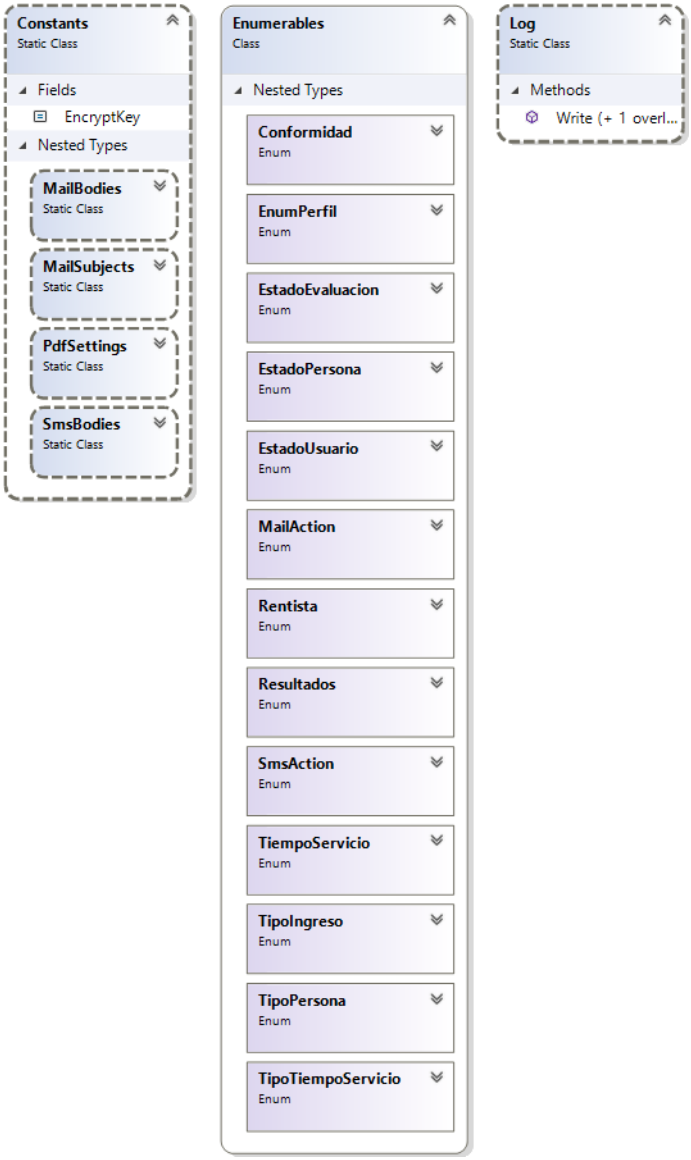
Fuente: Elaboración propia

Clases “Common”. Las clases que se encuentran en esta definición son Constants, Enumerables, y Log. La clase Constants almacena cuatro sub clases estáticas (MailBodies, MailSubjects, PdfSettings y SmsBodies), las cuales declaran valores que se utilizan dependiendo de las diferentes casuísticas para el envío de correos electrónicos, mensajes de texto y el formato de los documentos en PDF.

La clase Enumerables almacena un conjunto de tipos Enum con los valores que serán vinculados a sus clases correspondientes.

La clase Log implementa un único método, denominado Write, el cual se encarga de escribir la data de auditoría en la base de datos correspondiente cada vez que se realiza una acción. La Figura 41 muestra las clases mencionadas:

Figura 41: Clases Constants, Enumerables y Log



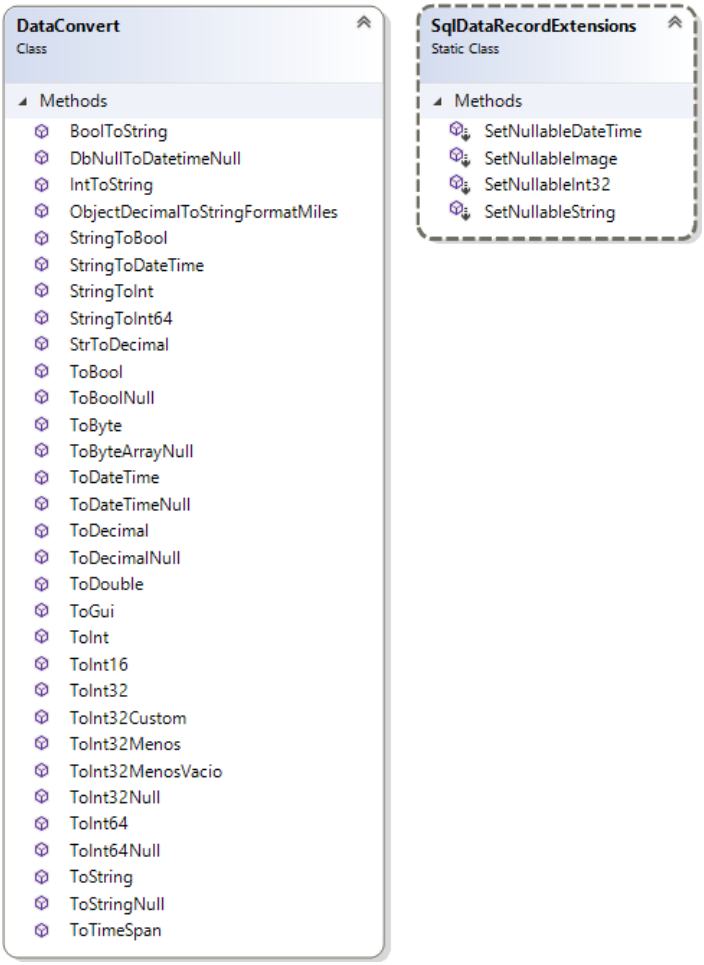
Fuente: Elaboración propia

Clases “Utils”. Dentro de estas clases se encuentran DataConvert y SqlDataRecordExtensions, las cuales funcionan como un paquete de utilizatarios para todo el sistema. DataConvert posee métodos que permiten la conversión de y hacia diversos tipos de datos, mientras que SqlDataRecordExtensions permite la devolución de valores nulos para ser

introducidos de forma masiva a los registros que se mandan en consultas de tipo bulk.

Las clases mencionadas en esta sección son mostradas en la Figura 42.

Figura 42: Clases DataConvert y SqlDataRecordExtensions

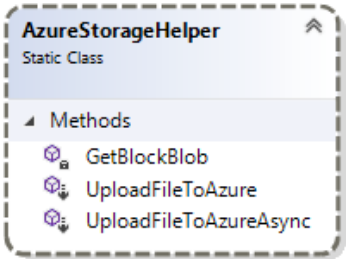


Fuente: Elaboración propia

Clase AzureStorageHelper. Es la clase encargada de realizar la carga y la descarga de los blobs al Azure Storage configurado en el tenant de la ONG. Esta clase contiene tres métodos estandarizados para tal fin.

La Figura 43 muestra el detalle de esta clase.

Figura 43: Clase AzureStorageHelper



Fuente: Elaboración propia

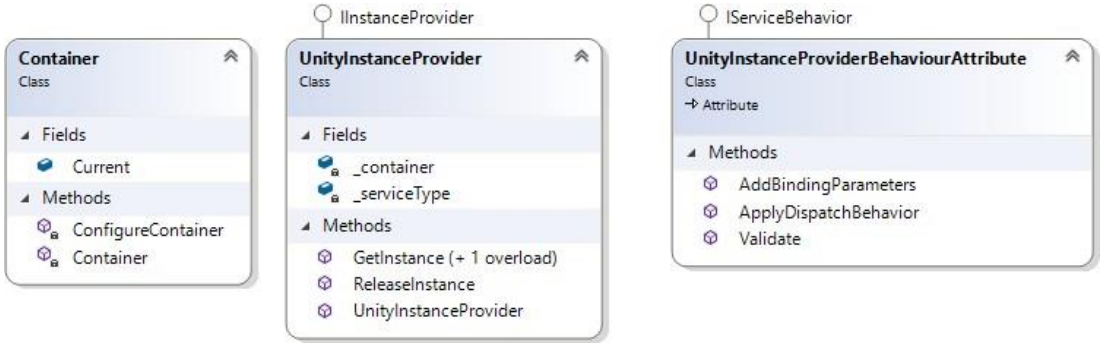
Clases para registrar “Instance Providers”. Dentro de esta categoría están:

Container, UnityInstanceProvider y UnityInstanceProviderBehaviourAttribute.

La clase Container realiza el trabajo de inyección de las dependencias de los objetos a las interfaces correspondientes. UnityInstanceProvider crea un InstanceProvider personalizada de proveer la instancia del servicio en lugar de utilizar el que posee WCF por defecto.

Por último, UnityInstanceProviderBehaviourAttribute registra el InstanceProvider creado, enlazando el mismo al container declarado en la clase Container. Las clases mencionadas se muestran en la Figura 44.

Figura 44: Clases Container, UnityInstanceProvider y UnityInstanceProviderBehaviourAttribute



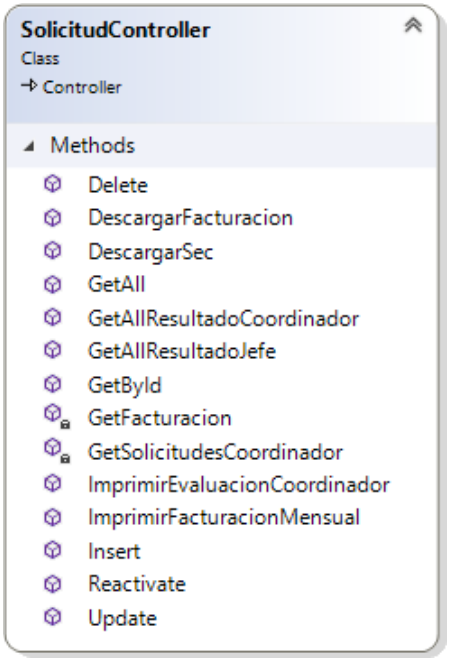
Fuente: Elaboración propia

5.1.4.3. Clases De Funciones Principales De La Aplicación Web

La aplicación web utiliza MVC, cambiando la funcionalidad principal de cada uno de los elementos del mismo, utilizando los controladores como los medios de consumo del servicio web. Asimismo, añaden funcionalidades que serán descritas en las clases que se mostrarán en las siguientes líneas, siendo las mismas las más importantes:

Clases para consumo del servicio. Dentro de esta categoría están todas las que implementan una instancia de la referencia del servicio como cliente para utilizar los métodos descritos en el contrato del mismo. Una de las clases principales es SolicitudController. Esta clase posee dos métodos privados, que recogen ciertos parámetros necesarios para generar documentos en PDF: GetFacturacion y GetSolicitudesCoordinador. La Figura 45 muestra la clase Solicitud Controller.

Figura 45: Clase SolicitudController



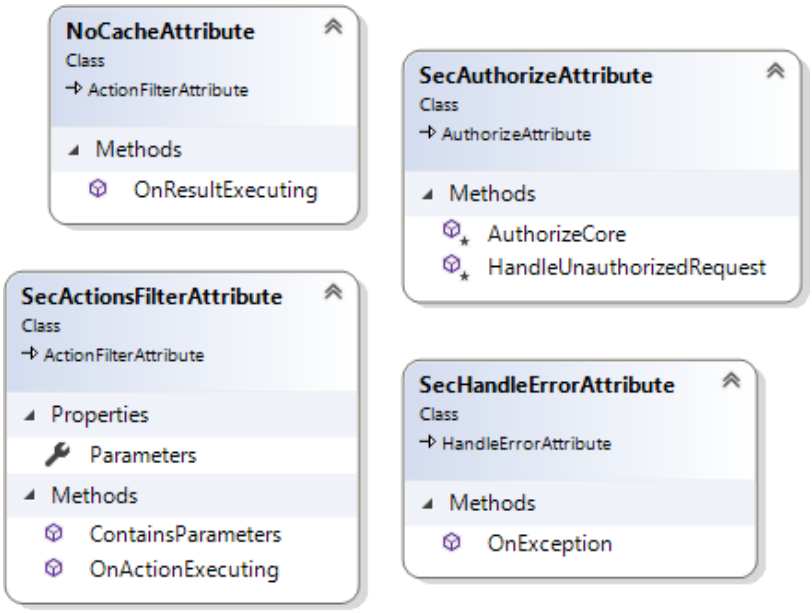
Fuente: Elaboración propia

Clases de filtro. Se incluyen NoCacheAttribute, SecActionsFilterAttribute, SecAuthorizeAttribute y SecHandleErrorAttribute como clases de este grupo.

NoCacheAttribute es una clase que funciona como filtro y atributo para los controladores y métodos que no deben guardar en caché las respuestas de salida. SecActionsFilterAttribute filtra la ejecución de las acciones en base a los parámetros que recibe de las mismas, devolviendo a la página principal en caso de no recibir parámetros. SecAuthorizeAttribute limita el acceso a los métodos en base a la verificación de si el usuario posee una sesión activa en base a la clase SessionHelper. SecHandleErrorAttribute maneja los errores que se presenten en la aplicación para registrarlos mediante un método expuesto por una interfaz en el contrato del servicio, encargada de realizar el registro de las excepciones producidas.

Estas clases se utilizan en diversos métodos y clases y, al ser atributos, ejecutan sus instrucciones antes de ejecutar las clases y métodos que los implementen. La Figura 46 muestra las clases mencionadas.

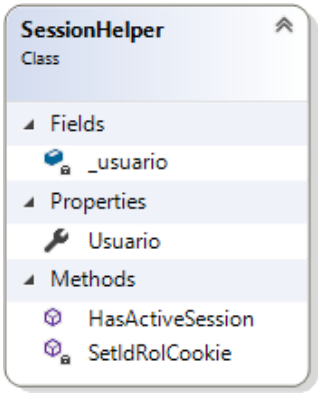
Figura 46: Clases NoCacheAttribute, SecAuthorizeAttribute, SecActionsFilterAttribute y SecHandleErrorAttribute



Fuente: Elaboración propia

Clase SessionHelper. La clase SessionHelper establece una cookie de sesión cuando el usuario inicia sesión en el sistema y verifica la validez de las cookies recibidas por la clase SecAuthorizeAttribute. La Figura 47 muestra las propiedades y los métodos de la clase descrita.

Figura 47: Clase SessionHelper



Fuente: Elaboración propia

5.1.5. CONFIGURACIÓN DEL ENTORNO DE NUBE Y DESPLIEGUE

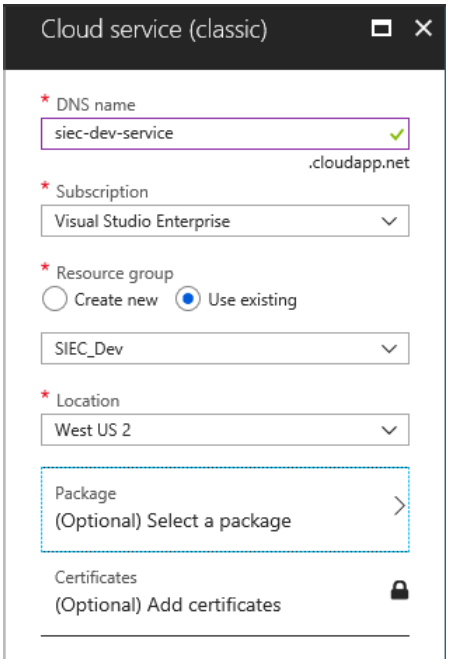
La configuración del grupo de recursos del cliente se describe en la siguiente sección, no considerándose la implementación de los Traffic Manager y detallando el despliegue de los recursos en un entorno de “Staging”. En este entorno se desplegó un Cloud

Service para el servicio web, un WebApp para el aplicativo web, un Azure Storage (classic) para el guardado de documentos y una base de datos Azure SQL DB. Cabe resaltar que los valores críticos, como nombres de recursos, usuarios, contraseñas y cadenas de conexión mostradas no son las instaladas para el cliente, siendo éstas creadas para este proyecto de tesis específicamente.

En el tenant del cliente se procedió con la creación del grupo de recursos SIEC_Dev, el cual contendrá todos los servicios contratados para este despliegue. La ubicación elegida para el grupo de recursos y sus servicios anidados será West US 2.Una vez creado el grupo de recursos, se procedió con la creación del Cloud Service, el cual se creó con los parámetros que se listan a continuación y se muestran en la Figura 48:

- **Nombre DNS:** siec-dev-service.cloudapp.net
- **Grupo de recursos:** SIEC_Dev
- **Ubicación:** West US 2

Figura 48: Microsoft Azure: Creación de Cloud Service



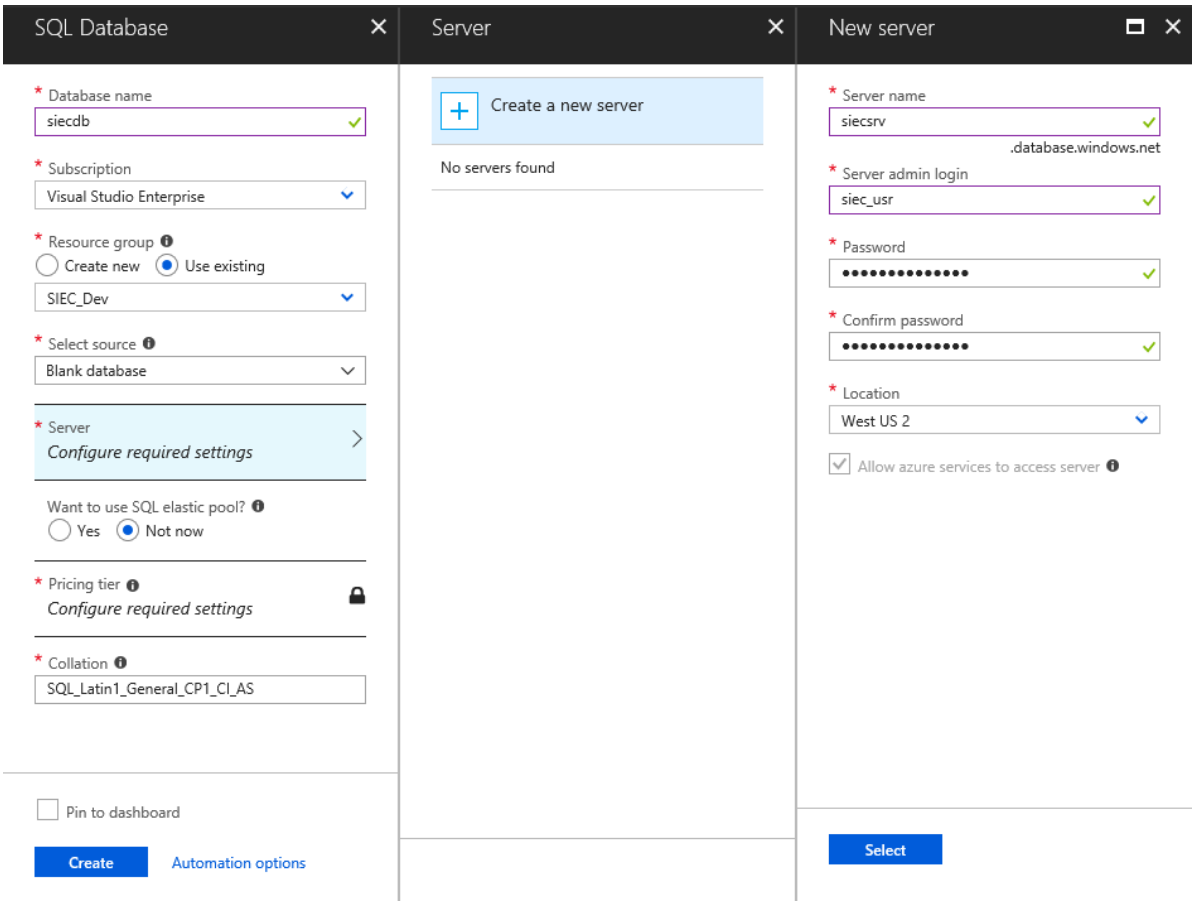
Fuente: Elaboración propia

El servicio web a desplegarse debe comunicarse con una base de datos en Azure SQL DB, por lo que se realizó la creación de la misma, y del servidor que la soportaría, con los siguientes parámetros (ver Figura 49):

- **Nombre de BD:** siecdb
- **Grupo de recursos:** SIEC_Dev
- **Fuente:** Base de datos en blanco
- Configuración del servidor:
- **Nombre:** siecsrv.database.windows.net
- **Usuario:** siec_usr
- **Contraseña:** devs$$16lho5lj
- **Ubicación:** West US 2

- **Ubicación:** West US 2
- Capa de precios: **S1**

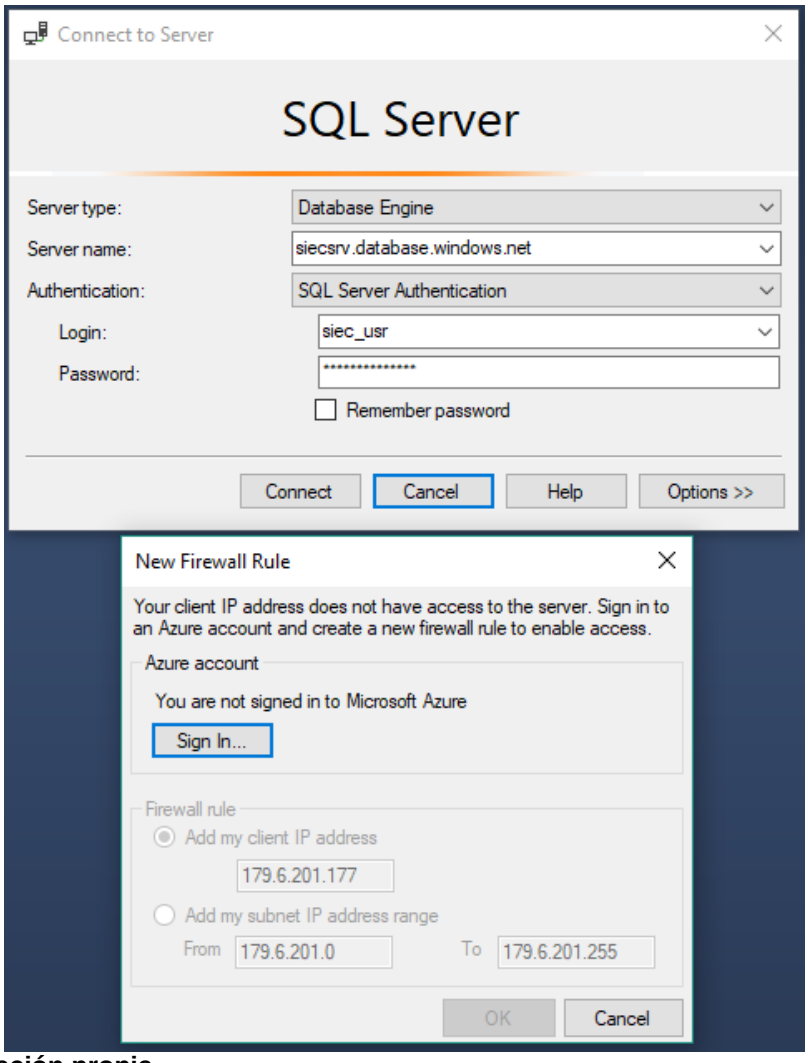
Figura 49: Microsoft Azure: Creación de SQL Database



Fuente: Elaboración propia

Tras su creación, se realizó la conexión con el servidor de la base de datos para realizar la ejecución del script de creación de tablas, procedimientos almacenados y valores por defecto, proporcionando el usuario y la contraseña configuradas anteriormente. Para la primera conexión, es necesario el registro de la dirección IP del dispositivo que está intentando acceder, lo que se realiza de forma automática con las credenciales de acceso a la cuenta de Azure donde se poseen los recursos o alguna forma de acceso IMO, tal y como se ve en la Figura 50:

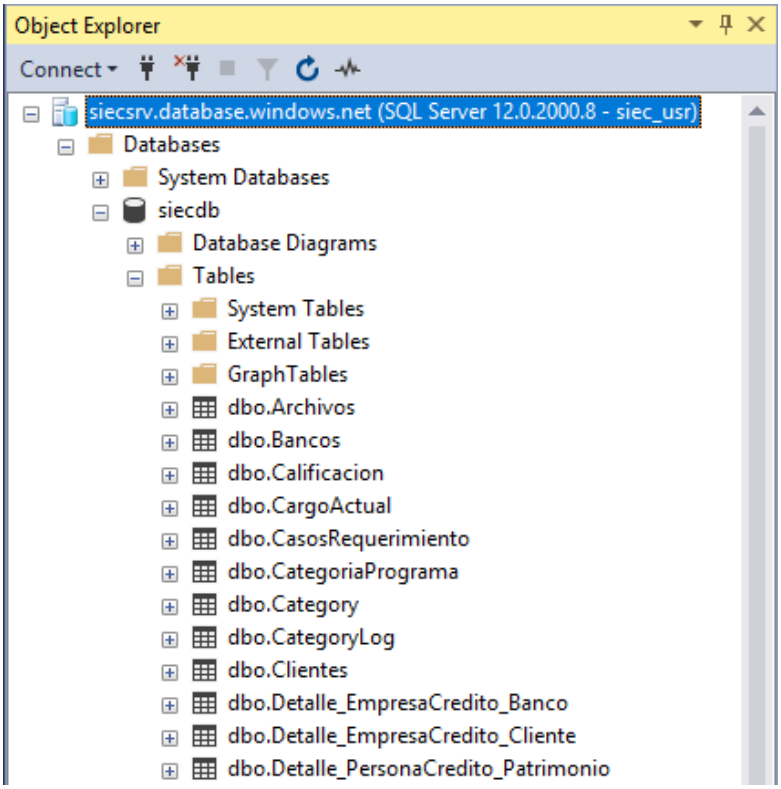
Figura 50: Pantalla de conexión de SQL Server Management Studio: Acceso y registro de permisos de firewall para Azure SQL DB



Fuente: Elaboración propia

La conexión exitosa muestra la base de datos desplegada, vacía, en la lista de bases de datos del servidor, en el panel Object Explorer. La Figura 51 muestra la base de datos tras la ejecución exitosa del script de creación de tablas, procedimientos almacenados y valores por defecto:

Figura 51: Microsoft Azure: Azure SQL DB desplegada

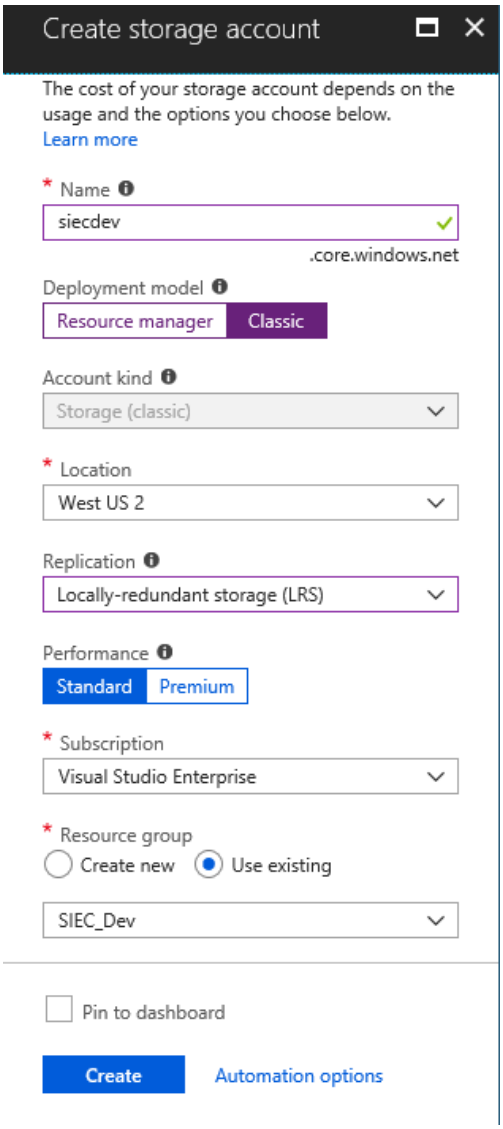


Fuente: Elaboración propia

Con la base de datos y el Cloud Service desplegados, se requiere de la creación del Azure Storage para el manejo y el guardado de blobs, el cual será consumido por el servicio que se desplegará en el Cloud Service. El despliegue de este servicio cuenta con los siguientes parámetros (ver Figura 52):

- Nombre: siecdev.core.windows.net
- Modelo de despliegue: Clásico
- Grupo de recursos: SIEC_Dev
- Ubicación: West US 2
- Replicación: Almacenamiento redundante localmente (LRS)
- Desempeño: Standard

Figura 52: Microsoft Azure: Creación de Azure Storage (Classic)



Fuente: Elaboración propia

El servicio web desarrollado en WCF sólo requiere del uso de la base de datos, el espacio de almacenamiento y la plataforma en la que será desplegado para su correcto funcionamiento, por lo que se procedió con la configuración del servicio para su despliegue. Lo primero fue registrar los valores de las cadenas de conexión para los servicios de Azure SQL DB, Azure Storage, Twilio y el servicio de correo, que serán consumidos por el servicio web. Esta configuración se realiza dentro del

archivo Web.Config, la cual se puede ver en Código Fuente 1 y Código Fuente 2. Cabe resaltar que algunos valores fueron modificados para evitar el filtrado de información, otros (como los valores para el servicio de Twilio), viciados con caracteres X, esto con el fin de evitar el uso inapropiado de la información.

Código Fuente 1. Registro de conexión a Azure SQL DB en Web.Config

```
<connectionStrings>

<add name="SIEC" connectionString="Server=tcp:siecsrv.database.windows.net,1433;Initial
Catalog=siecdb;Persist Security Info=False;User
ID=siec_usr;Password=devs$$16lho5lj;MultipleActiveResultSets=False;Encrypt=T
rue;TrustServerCertificate=False;Connection Timeout=30;"
providerName="System.Data.SqlClient" />

</connectionStrings>
```

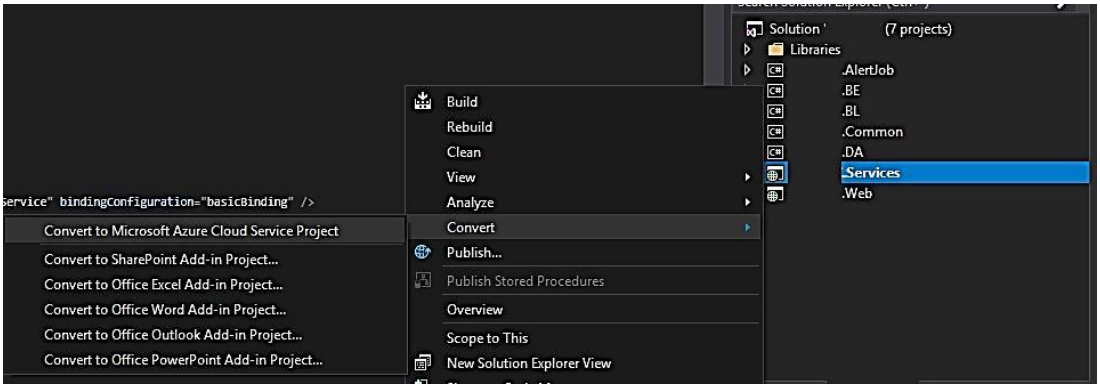
ódigo Fuente 2. Registro de parámetros de configuración para Azure Storage, servicios de Twilio,

```
<appSettings>

    <add key="aspnet:UseTaskFriendlySynchronizationContext" value="true" />

    <add key="mailAccount" value="pruebas@siectest.com" />

    <add key="mailPassword" value="73dj8m2m" />

    <add key="smtp" value="smtp.office365.com" />

    <add key="mailDisplayName" value="SIEC" />

    <add key="systemUrl" value="http://siec-dev-service.azurewebsites.net" />

    <add key="smsAccount" value="XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX" />

    <add key="smsAuthToken" value="XXXXXXXXXXXXXXXXXXXXXXXXXXXXXXXX" />
```

Servicio de envío de correos y dirección de WebApp Fuente: Elaboración propia

Una vez realizadas dichas configuraciones, dentro del IDE de Visual Studio, se realiza la conversión del proyecto del servicio a un proyecto “Microsoft Azure Cloud Service Project”, tal y como se ve en la Figura 53.

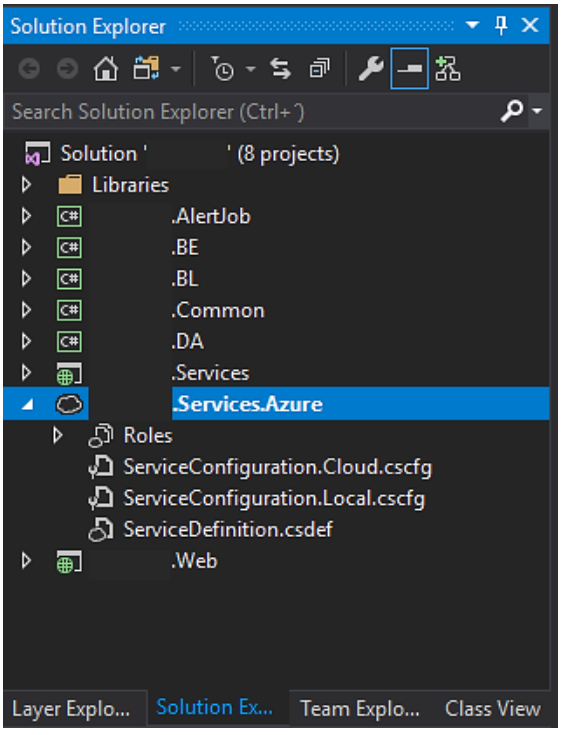
Figura 53: Conversión del proyecto WCF a Microsoft Azure Cloud Service Project



Fuente: Elaboración propia

Como resultado, se obtiene un proyecto que es el que será publicado al servicio Cloud Service, el cual contiene un perfil a ser configurado para futuros despliegues. El proyecto es mostrado en la Figura 54.

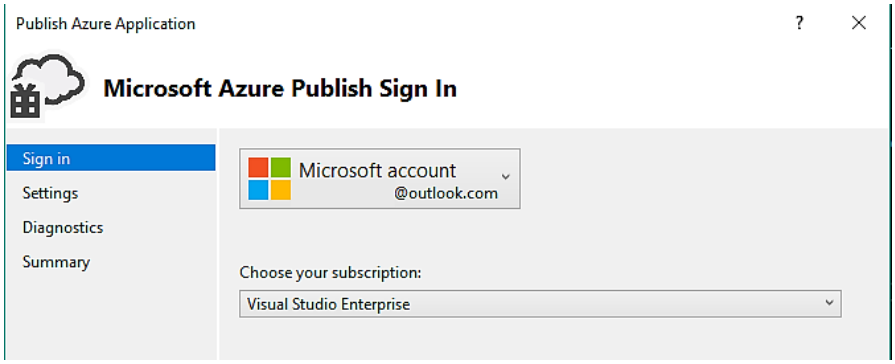
Figura 54: Captura de Solution Explorer de Visual Studio: Microsoft Azure Cloud Service Project creado



Fuente: Elaboración propia

Se le da clic derecho a este proyecto, seleccionando la opción “Publicar”. Antes de proceder con la publicación del servicio, se debe aplicar una configuración para establecer la plataforma de destino en la que se realizará el despliegue de la aplicación. Al presionar el botón “Publicar”, se muestra un cuadro de diálogo que solicitará el acceso a la cuenta y a la suscripción de Azure que posee los accesos al tenant en el que está configurado el Cloud Service. En el caso del proyecto de esta tesis, el correo electrónico elegido es el correo personal, motivo por el que se oculta dicha información, y el tipo de suscripción es “Visual Studio Enterprise”. La Figura 55 muestra el cuadro de diálogo mencionado.

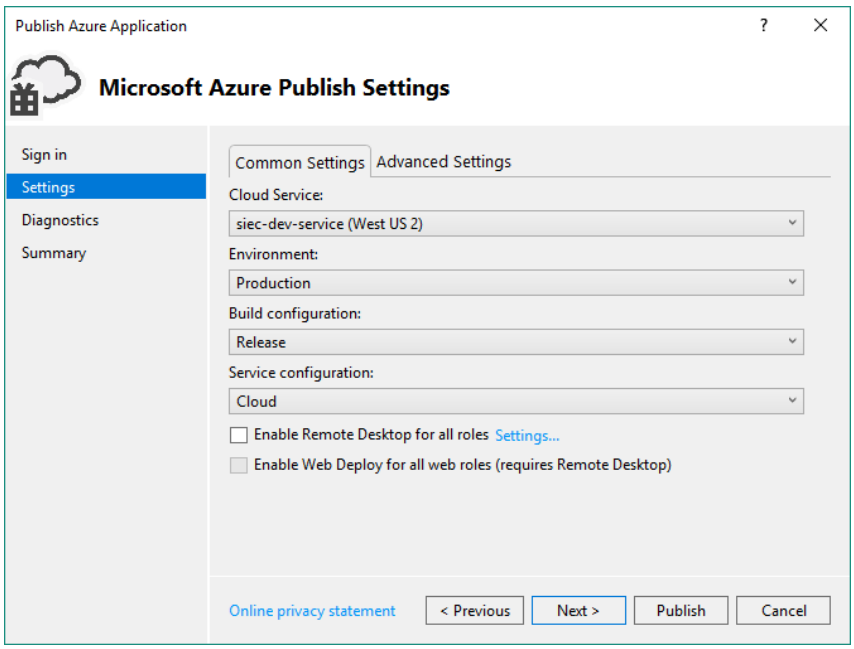
Figura 55: Cuadro de diálogo de Configuración para publicación - Inicio de sesión



Fuente: Elaboración propia

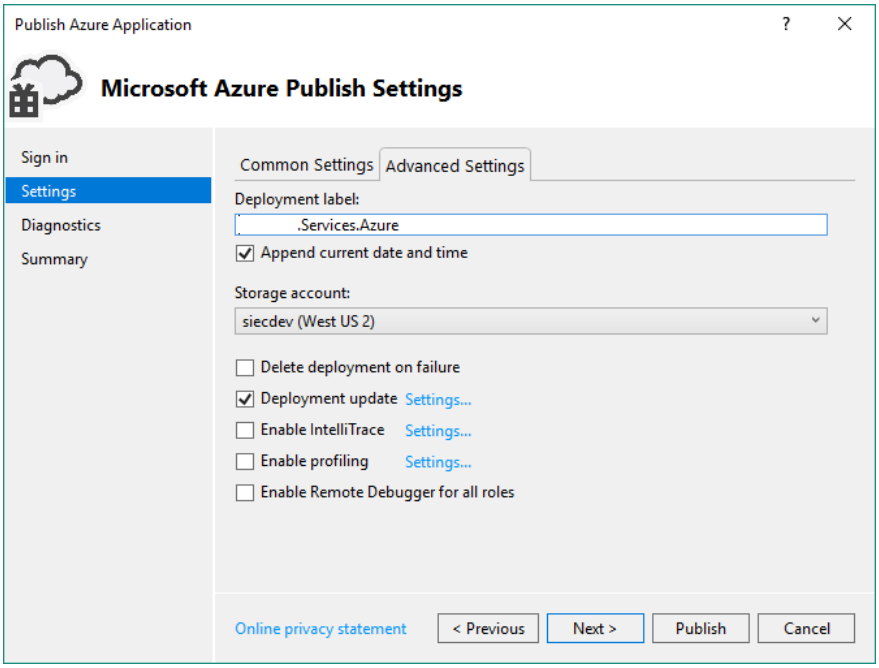
La configuración común incluye la selección del Cloud Service desplegado previamente, la indicación del entorno (de producción para el caso de estudio), el tipo de compilación y el tipo de configuración para el servicio. En el cuadro de configuración avanzada, se añade la etiqueta de despliegue y se selecciona el servicio de Azure Storage configurado previamente. Luego, se marca la opción Deployment Update, lo cual le indicará al perfil de configuración de despliegue que se realizará una publicación incremental, evitando tiempos largos tras cada cambio en el proceso de desarrollo. La Figura 56 y la Figura 57 muestran las configuraciones aplicadas en este proyecto.

Figura 56: Cuadro de diálogo de Configuración para publicación - Configuración común



Fuente: Elaboración propia

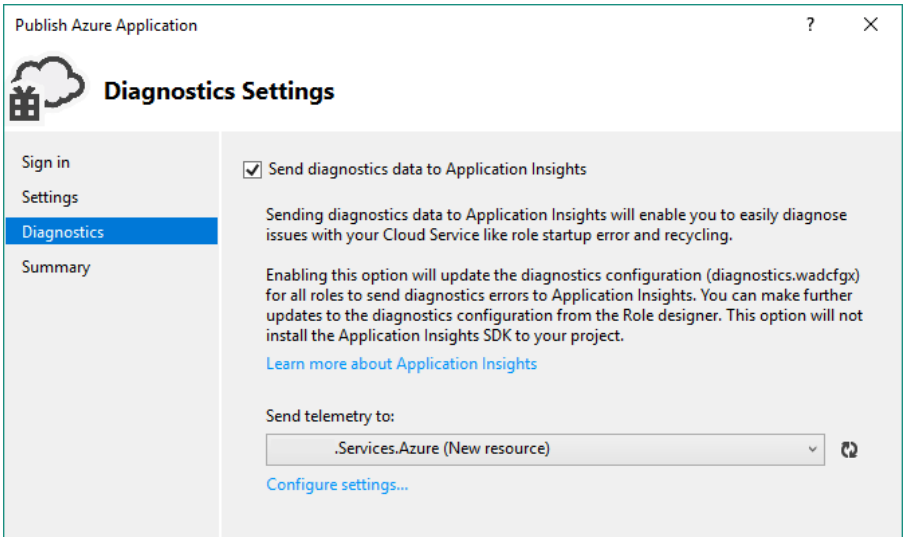
Figura 57: Cuadro de diálogo de Configuración para publicación - Configuración avanzada



Fuente: Elaboración propia

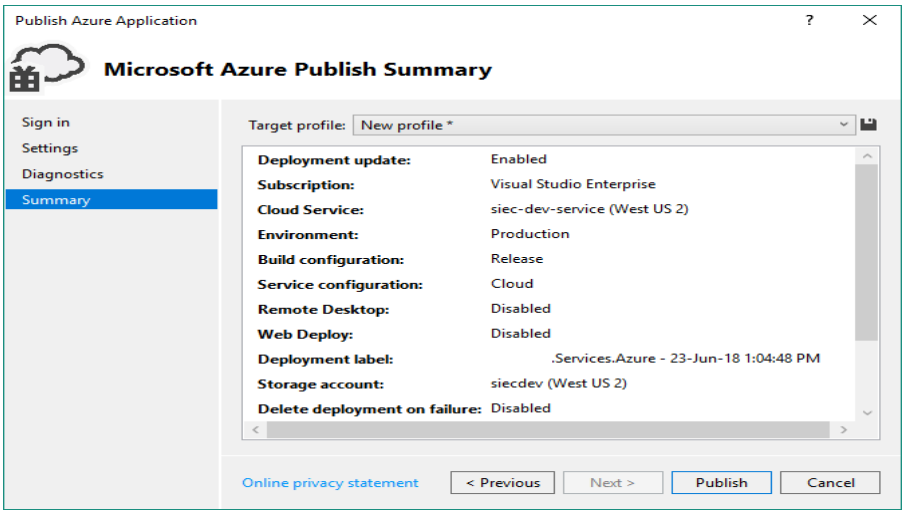
Luego, se seleccionó la opción de envío de información de diagnóstico al servicio de Application Insights, el cual nos brindará datos de telemetría referidos al uso y estado de salud del servicio. Para esto, se selecciona el destino de guardado de la información de telemetría, como lo muestra la Figura 58. Finalmente, se puede revisar un sumario del perfil configurado y listo para realizar la publicación, el mostrado en la Figura 59, dando inicio a la misma luego de pulsar “Publicar”.

Figura 58: Cuadro de diálogo de Configuración para publicación – Diagnóstico



Fuente: COlaboración propia

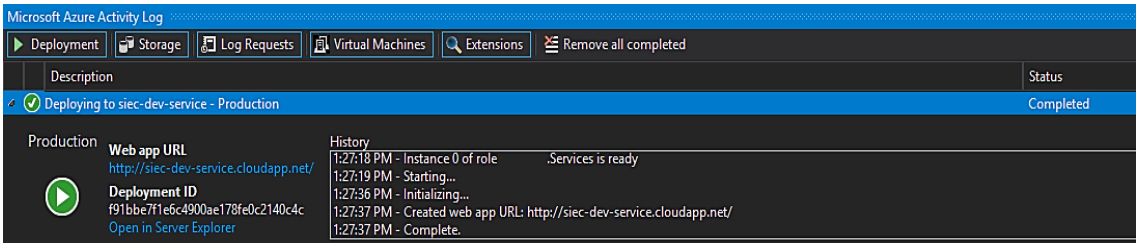
Figura 59: .Cuadro de diálogo de Configuración para publicación- Sumario y publicación



Fuente: Elaboración propia

Visual Studio comienza con el proceso de publicación, brindando información detallada sobre los sucesos que se van registrando desde el Cloud Service. Una vez finalizado el proceso de publicación, se puede ver el cuadro mostrado en la Figura 60.

Figura 60: Finalización de despliegue de servicio



Fuente: Elaboración propia

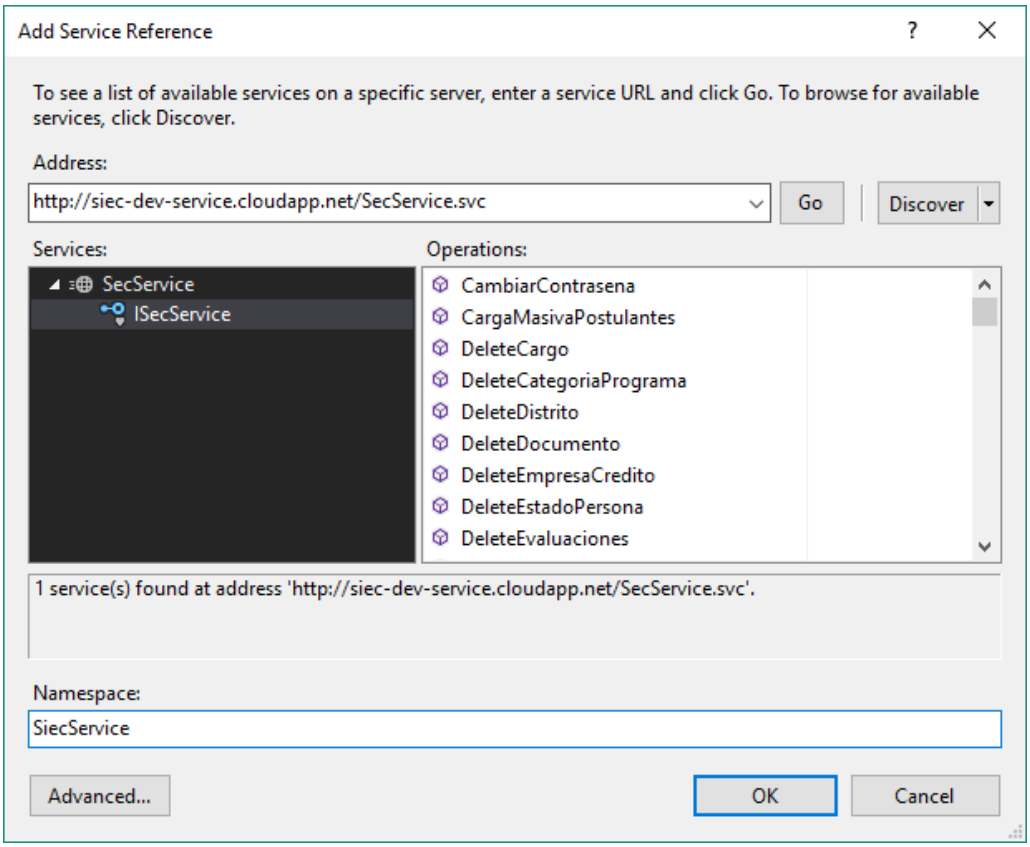
Con el servicio configurado y desplegado en el servicio de nube, Cloud Service, se realizó el registro de la referencia del servicio en la aplicación web, para proceder con su despliegue. Para realizar esta configuración, hay que desglosar el proyecto web MVC, dar clic derecho a “Service References” y seleccionar la opción “Add Service Reference…”. Hecho esto, se muestra un cuadro de diálogo en el que se debe dar información sobre el servicio publicado.

Hay que recordar que el servicio fue montado en la dirección http://siec-dev-service.cloudapp.net/. Sin embargo, las interfaces del servicio creado se encuentran dentro de un archivo con extensión SVC. Realizar la referencia a la dirección antes mencionada no mostraría el servicio desplegado como tal, por lo que se ingresó la ruta completa: http://siec-dev-service.cloudapp.net/SecService.svc. Una vez hecho esto, se mostró el servicio y la interfaz con los métodos desarrollados. Sólo fue necesario añadir el Namespace para la referencia al servicio. Esto se muestra reflejado en la Figura 61.

Es menester mencionar que el proyecto web se desarrolló utilizando el servicio original del cliente, por lo que, para motivos de desarrollo del presente trabajo, se realizó el despliegue en un tenant personal, siendo necesario únicamente actualizar el Service Reference para apuntar al servicio desplegado en el nuevo Cloud Service. Sin embargo, este procedimiento se realiza durante el desarrollo de la aplicación, ya

que se necesita tener acceso a los métodos que expone el servicio para ser llamados desde los controladores mediante una instancia del Service Reference que actúe como cliente del mismo.

Figura 61: Configuración del Service Reference en el proyecto web



Fuente: Elaboración propia

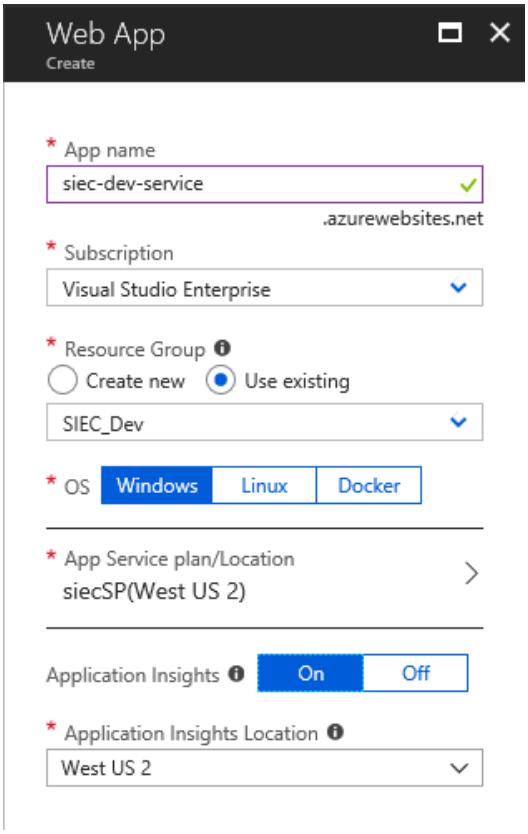
Al haber realizado las configuraciones antes mencionadas en el proyecto web, se procedió con su despliegue. Para poder realizarlo, fue necesario configurar la plataforma

de despliegue, el Web App. En el portal de Azure se creó uno nuevo, aplicando la siguiente configuración (ver Figura 62):

- Nombre:siec-dev-service.azurewebsites.net

- Grupo de recursos: SIEC_Dev
- Sistema Operativo: Windows
- Application Insights: Sí
- Ubicación: West US 2

Figura 62:. Microsoft Azure: Creación de Web App

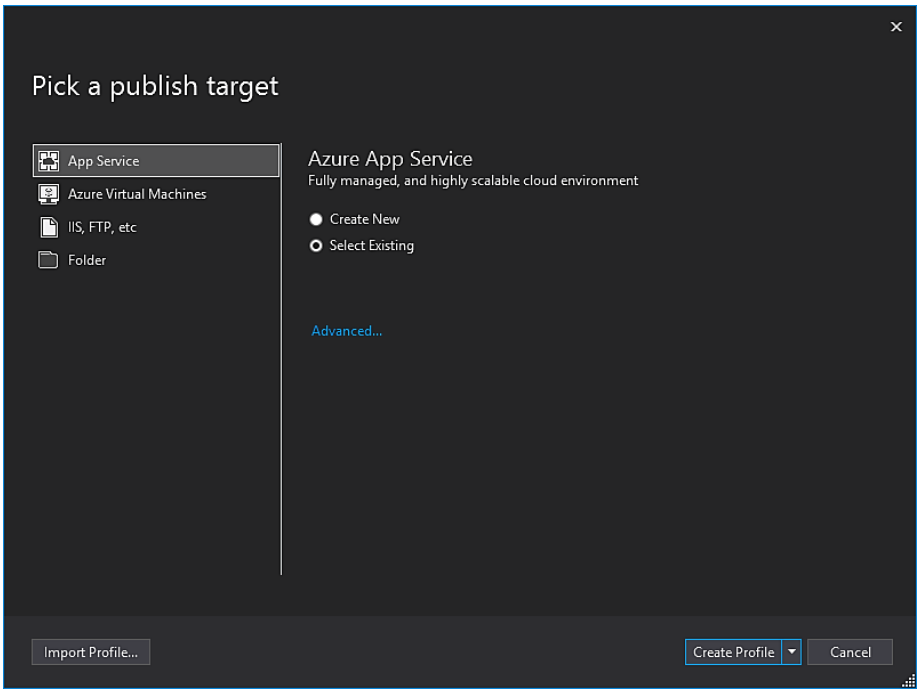


Fuente: Elaboración propia

Teniendo el Web App configurado, se procedió con la publicación de la aplicación web. Para realizarla, se hizo clic derecho en el proyecto web de la solución y se seleccionó la opción “Publish”. Se mostró un cuadro de diálogo, en el que se daba la opción de elegir entre crear un nuevo App Service de Microsoft Azure o seleccionar uno existente. Se marcó la segunda opción y se presionó el enlace “Advanced…”.

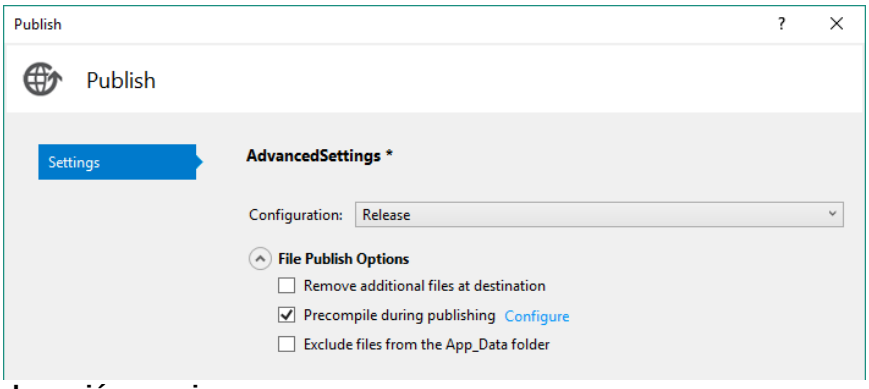
Dentro de la configuración avanzada que se muestra en la Figura 64, se seleccionó la configuración Release para no permitir la integración de las herramientas de depuración de código al momento de ejecutar la solución. Asimismo, se marcó la casilla “Precompile during publishing”, garantizando que la publicación se realice únicamente tras una compilación exitosa de la solución. Al finalizar, se presionó el botón “Create Profile”, para crear un perfil de publicación, como se hizo con el servicio WCF antes de ser desplegado al Cloud Service, tal y como se ve en la Figura 63.

Figura 63: Selección de plataforma de destino de publicación



Fuente: Elaboración propia

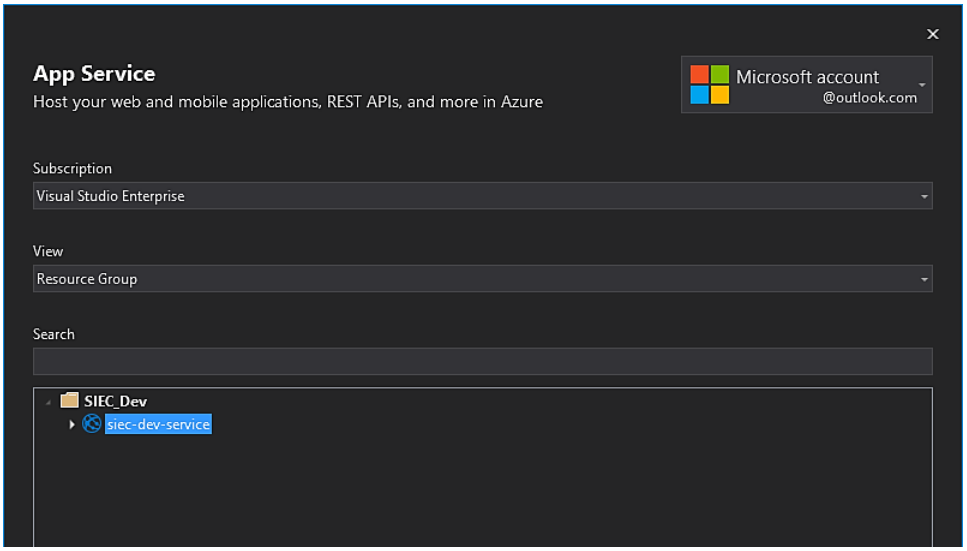
Figura 64: Configuración avanzada del perfil de publicación



Fuente: Elaboración propia

Al presionar el botón “Create Profile” en el cuadro anterior, el formulario solicita los datos de la suscripción en la que se encuentra el grupo de recursos. Una vez seleccionado el grupo de recursos, se desglosó la lista de los Web Apps creados, en nuestro caso fue el Web App siec-dev-service, creado anteriormente. La Figura 65 muestra el cuadro el recurso mencionado en la suscripción.

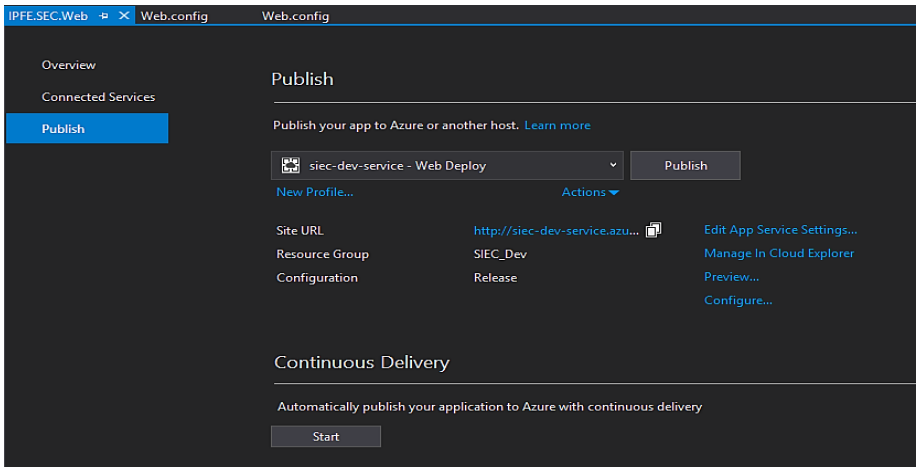
Figura 65: Confirmación de selección de App Service existente.



Fuente: Elaboración propia

Tras este proceso de configuración, se muestra una vista donde se encuentra registrado el perfil configurado, el cual se ve en la Figura 66.

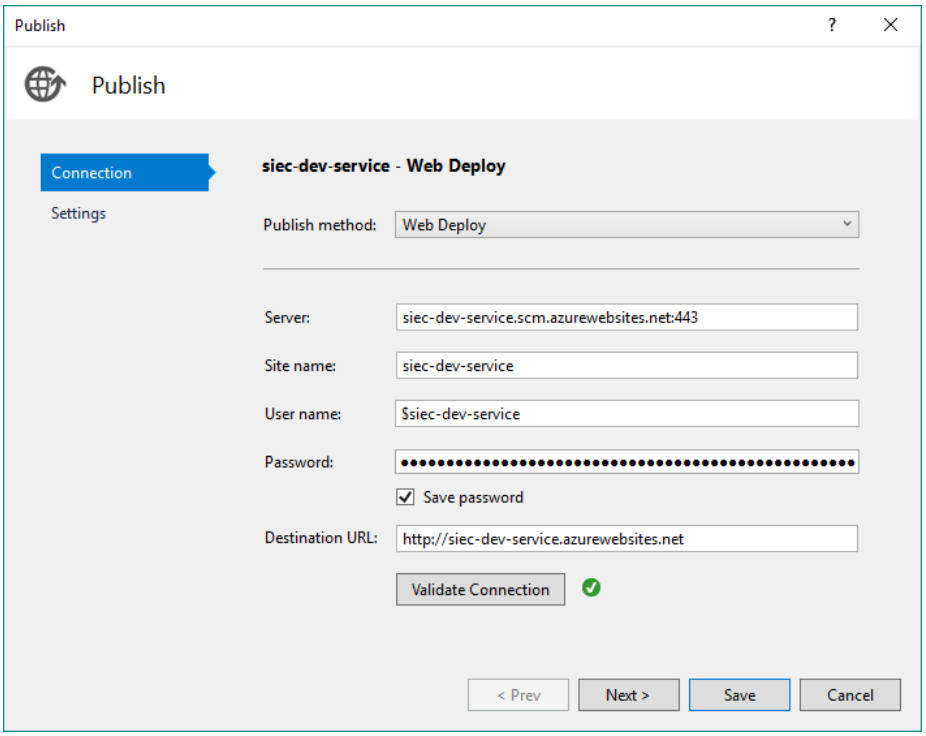
Figura 66: Confirmación de selección de perfil



Fuente: Elaboración propia

Antes de finalizar con la publicación del proyecto web, se realizó una verificación de la conexión con el Web App. Para realizar esto, se dio clic a la opción “Configure…”, abriéndose el cuadro de diálogo mostrado en la Figura 67. En este cuadro de diálogo, se selecciona la opción Web Deploy, que es el método por defecto, se verifican los parámetros mostrados, como el servidor y la dirección de destino. El nombre de usuario y la contraseña son extraídos automáticamente por las herramientas de Visual Studio. Una vez hecho esto, se presionó el botón “Validate Connection”, procediendo con la publicación del sistema desde la vista mostrada en la Figura 66.

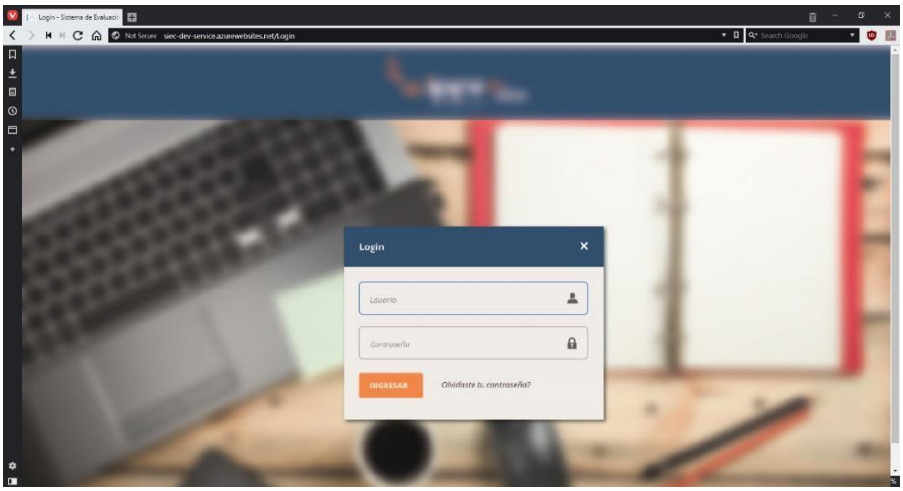
Figura 67: Verificación de conexión con Web App



Fuente: Elaboración propia

Finalmente, una vez concluida la publicación de la aplicación, se tiene el sistema desplegado en el Web App de destino, logrando acceder a él mediante la dirección configurada previamente: http://siec-dev-service.azurewebsites.net.

Figura 68: Web App desplegado con la aplicación web



Fuente: Elaboración propia

5.2. PRUEBAS Y RESULTADOS

Las pruebas se llevaron a cabo desde el 25 de julio del 2016 hasta el 02 de agosto del 2016 con el Sistema implementado, ya que los servicios de Azure permiten recolectar la información de resultados que se requería en las pruebas de escalabilidad.

5.2.1. PRUEBAS DE USABILIDAD

Para poder realizar las pruebas de usabilidad del sistema desarrollado en este trabajo, se elaboraron un conjunto de encuestas a diferentes actores del sistema, llegando a sumar un total de 19 usuarios encuestados. El formato de la encuesta se encuentra en la sección de Anexos (Anexo A.7. Modelo de Encuesta Empleada en las Pruebas de Usabilidad).

La elección del modelo de encuesta se dio debido a que se desea obtener parámetros uniformes y un análisis completo a las características de un sistema con usabilidad. Este cuestionario tenía una escala de opciones para cada uno de los enunciados puestos en él, que varían del 1 al 5 de la siguiente manera: 1 – En total desacuerdo, 2 – No tan de acuerdo, 3 – Parcialmente de acuerdo, 4 – Muy de

acuerdo, 5 – Totalmente de acuerdo. De esto se puede derivar que 1 es la opción menos favorable y 5 es la opción más favorable. Además, las preguntas se dividieron en 5 grupos, lo cual determinaría las características de usabilidad consideradas por los usuarios:

- **Navegación.** Referida a la experiencia de uso a nivel de apariencia visual, control y manejo de los elementos, así como la experiencia de fluidez.
- **Accesibilidad.** Se refiere a la facilidad de acceso y uso delsistema.
- **Funcionalidad.** Referido al cumplimiento de las funciones esperadas y actuales del sistema.
- **Ayuda.** El cual se refiere a la documentación otorgada y el contenido de la misma para el uso del sistema.
- **Prevención de errores.** Se refiere a la capacidad del sistema de prevenir el uso inadecuado del sistema y evitar resultados no esperados en el comportamiento del mismo, a nivel de usuario común. Una vez realizadas las encuestas, se contabilizaron la cantidad de veces que se repetía una respuesta para cada pregunta, en las 19 encuestas realizadas. De esto, se obtuvo lo mostrado en la siguiente Tabla 40:

Tabla 40: Conteo de respuestas por pregunta

SECCIÓN / PREGUNTA	EN DESACUERDO	NO TAN DE ACUERDO	PARCIALMENTE	MUY DE ACUERDO	TOTALMENTE DE ACUERDO
TOTAL DE NAVEGACIÓN	**0**	**5**	**17**	**39**	**34**
La aplicación carga rápidamente	0	0	4	9	6

El sistema presenta la información de forma clara y comprensiva	0	0	5	5	9
El tipo y tamaño de letra empleados en el sitio permiten su manejo sin esfuerzo visual	0	5	3	4	7
Las páginas existentes en el sistema muestran contenido bien escrito, sin errores ortográficos	0	0	0	12	7
La apariencia de la aplicación web no se distorsiona en la pantalla, de escritorio o portátil, que utilizo.	0	0	5	9	5
TOTAL DE ACCESIBILIDAD	**0**	**0**	**7**	**22**	**28**
Existe coherencia en la distribución de los elementos presentes en todas las páginas de la aplicación	0	0	0	7	12
La navegación con el botón TAB es posible, facilitando la navegación	0	0	0	9	10
Existe claridad sobre la función de los botones, las listas y los controles proporcionados por el sistema	0	0	7	6	6
TOTAL DE FUNCIONALIDAD	**7**	**6**	**7**	**8**	**10**
El sistema me permite realizar todas las funciones que esperaba	7	6	3	2	1
El funcionamiento del sistema es sencillo de entender y manejar	0	0	4	6	9
TOTAL DE AYUDA	**0**	**0**	**5**	**16**	**17**
Los documentos que me otorgaron, a modo de manuales de ayuda, son claros	0	0	0	9	10

Pude replicar el funcionamiento detallado en los manuales de ayuda sin dificultad	0	0	5	7	7
TOTAL DE PREVENCIÓN DE ERRORES	**0**	**2**	**5**	**13**	**18**
El sistema realiza la validación de la información ingresada para evitar errores	0	2	5	8	4
El sistema solicita confirmación antes de realizar acciones que serán registradas	0	0	0	5	14

Fuente: Elaboración propia

Las filas de "total" se obtuvieron sumando la cantidad de respuestas por cada indicador, con el fin de calcular los porcentajes de cada criterio de usabilidad.

Con los datos de la Tabla 40, se realizó el cálculo de los porcentajes para cada grupo de preguntas en base a la cantidad de usuarios encuestados. Esto se hizo mediante la división de la sumatoria para cada respuesta, entre la cantidad de encuestas. El resultado de dicha operación, que se expresa en porcentajes, puede verse en la Tabla 41:

Tabla 41: Valores porcentuales por grupo de preguntas

	EN DESACUERDO	NO TAN DE ACUERDO	PARCIALMENTE	MUY DE ACUERDO	TOTALMENTE DE ACUERDO
NAVEGACIÓN	0.00	5.26	17.89	41.05	35.79
ACCESIBILIDAD	0.00	0.00	12.28	38.60	49.12
FUNCIONALIDAD	18.42	15.79	18.42	21.05	26.32
AYUDA	0.00	0.00	13.16	42.11	44.74

PREVENCIÓN DE ERRORES	0.00	5.26	13.16	34.21	47.37

Fuente: Elaboración propia

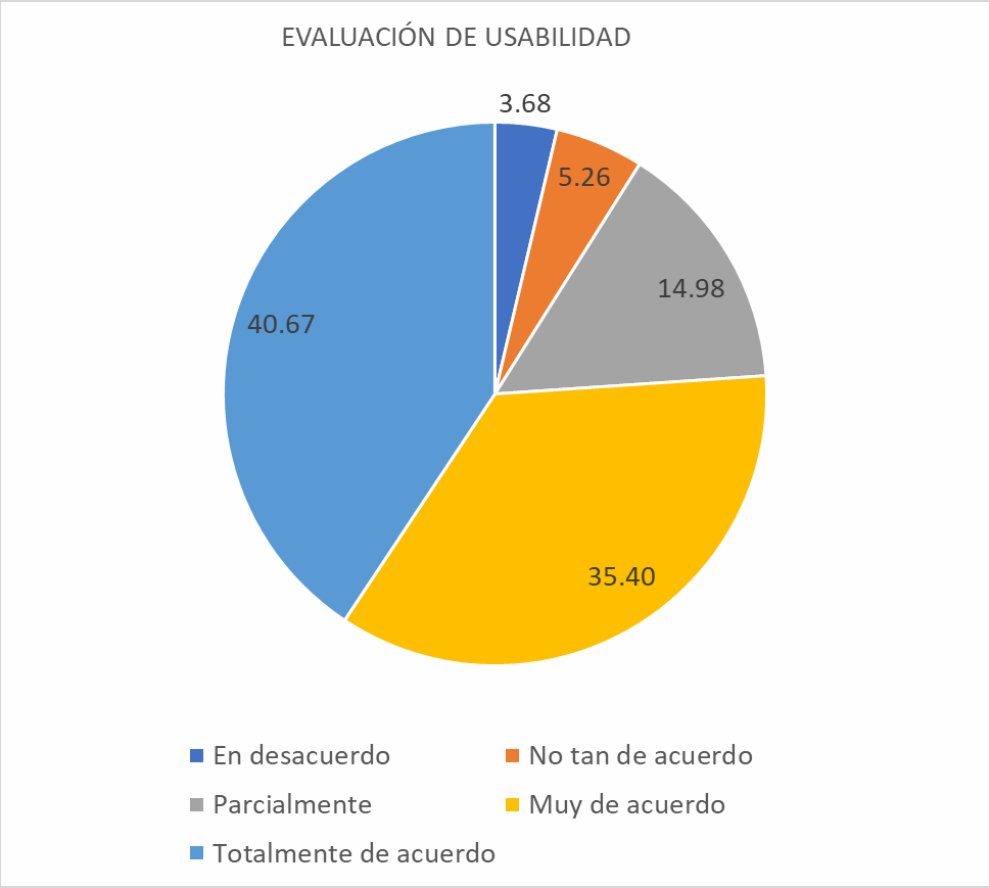
Una vez finalizado este procedimiento, y para resumir los resultados de los criterios de usabilidad, se promediaron los resultados de la Tabla 41 considerando cada una de las columnas. Es decir, el promedio se calculó considerando “En desacuerdo”, “No tan de acuerdo” y así sucesivamente, obteniéndose los resultados mostrados en la Tabla 42 y en la Error! Reference source not found.:

Tabla 42: Resultados porcentuales de las encuestas

EN DESACUERDO	NO TAN DE ACUERDO	PARCIALMENTE	MUY DE ACUERDO	TOTALMENTE DE ACUERDO
3.68	5.26	14.98	35.40	40.67

Fuente: Elaboración propia

Figura 69: Resultados porcentuales de las encuestas



Fuente: Elaboración propia

La lista de los casos de prueba empleados en esta etapa se muestra en el Anexo A.2. Casos de Prueba

CONCLUSIÓN

Como conclusión de los resultados de usabilidad, tenemos:

- En desacuerdo: 3.6%
- No tan de acuerdo: 5.3%
- Parcialmente: 14.9%
- Muy de acuerdo: 35.4%
- Totalmente de acuerdo: 40.7%

De estos resultados podemos interpretar que los criterios empleados para medir la usabilidad del sistema fueron cumplidos de forma satisfactoria. Esto se deduce al observar un porcentaje de 76.1% de respuestas positivas a los enunciados establecidos en la encuesta (opciones 4 y 5, “Muy de acuerdo” y “Totalmente de acuerdo”, respectivamente). La razón de elegir ambas opciones como resultados favorables, fue que ambas expresan un grado positivo de satisfacción con respecto al criterio al que hacen referencia.

Además, se ha obtenido un 14.8% de respuestas “Parcialmente”, que no resultan desfavorables, sino que marcan ciertos puntos en los que poner énfasis en futuras entregas para mejorar la experiencia de uso de la aplicación. El resultado desfavorable para esta encuesta suma un total de 8.94%, estando más enfocados al grupo de preguntas de funcionalidad. Esto se explica debido a que muchas funcionalidades solicitadas por ellos fueron desestimadas por el equipo de TI de la ONG, debido a que no cumplían estrictamente un papel en el flujo del proceso para el que se deseaba crear el sistema. Probablemente la falta de comunicación interna sobre la desestimación de sus solicitudes haya llevado a estos resultados, ya que los requerimientos aceptados han sido cumplidos eficazmente.

5.2.2. PRUEBAS DE ESCALABILIDAD Y RENDIMIENTO

Para las pruebas de escalabilidad y rendimiento se tuvo el siguiente criterio: el sistema ha sido pensado para la evaluación de entre 500 y 800 solicitudes mensuales; además, existen 14 colaboradores que desempeñan actualmente el papel de los actores del sistema, entre los pertenecientes a la ONG y los de la universidad con la que funcionó inicialmente el sistema. De esto se calcula un promedio de 30 solicitudes diarias. Suponiendo que deba atenderse a los 30 postulantes de forma

concurrida, daría un total de 44 usuarios concurrentes en el sistema en un escenario común, atendiendo la cantidad de solicitudes previstas y con los usuarios de la ONG y universidad realizando su trabajo de forma simultánea.

Tabla 43: Tabla de concurrencia de usuarios

HORA	LUNES	MARTES	MIÉRCOLES	JUEVES	VIERNES	SÁBADO
9:00 - 9:30	18	16	17	8	9	23
9:30 - 10:00	24	8	16	16	8	29
10:00 - 10:30	15	6	17	16	6	20
10:30 - 11:00	13	12	13	16	16	22
11:00 - 11:30	10	10	15	24	18	16
11:30 - 12:00	11	7	31	21	15	10
12:00 - 12:30	8	20	18	21	14	16
12:30 - 13:00	12	28	23	4	20	18
13:00 - 13:30	6	9	3	5	6	2
13:30 - 14:00	6	3	0	6	7	6
14:00 - 14:30	19	10	21	21	8	13
14:30 - 15:00	15	24	14	23	16	16
15:00 - 15:30	11	13	9	16	10	20
15:30 - 16:00	20	14	21	17	8	18
16:00 - 16:30	12	14	5	21	22	17
16:30 - 17:00	23	11	19	15	10	19
17:00 - 17:30	17	16	9	13	14	4
17:30 - 18:00	11	19	16	10	15	9

Fuente: Área de TI de la ONG

Sin embargo, este sistema ha sido diseñado pensando en que sería la primera fase de un proceso de sistematización de los procesos globales de la organización. Por lo

que la tasa de usuarios subirá en futuras implementaciones. Además, hay que considerar que la tasa de postulantes es variable en el tiempo.

Considerando todo lo anterior, se estableció el siguiente criterio para la configuración de escalabilidad de los servicios web configurados en la nube de Microsoft Azure (ver Figura 70):

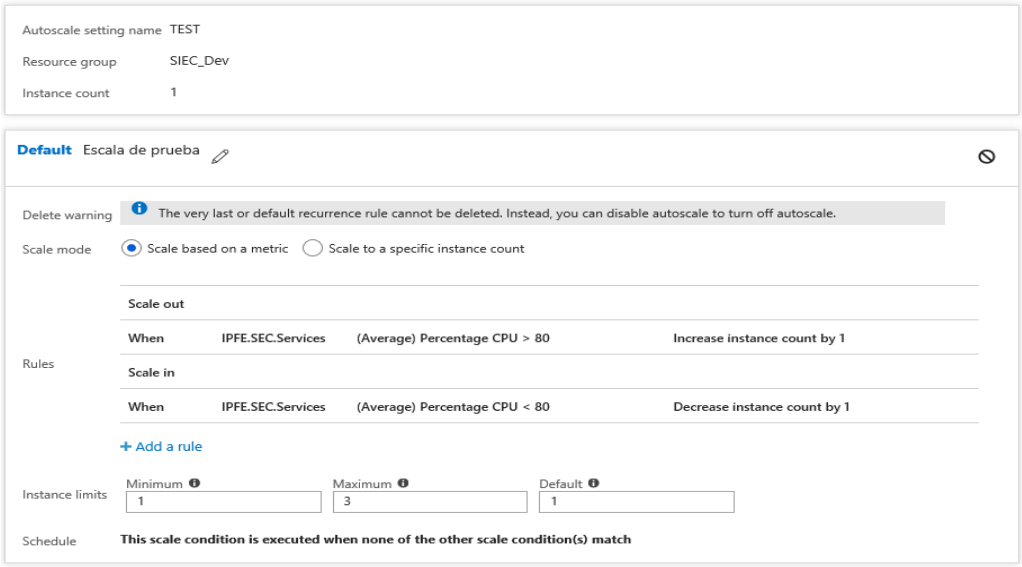
Scale out:

- Uso de procesador mayor a 80%
- Duración: 10 minutos de cumplimiento de condición
- Incremento de instancias: 1
- Escala basada en métricas
- Tiempo de enfriamiento: 5 minutos

Scale In:

- Uso del procesador menor a 80%
- Duración: 10 minutos de cumplimiento de condición
- Reducción de instancias: 1
- Escala basada en métricas
- Tiempo de enfriamiento: 5 minutos

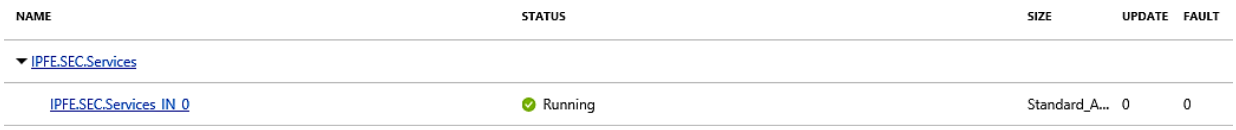
Figura 70: Captura del panel de configuración de escalabilidad de Microsoft Azure



Fuente: Elaboración propia

Luego de aplicar esta configuración, se revisó la cantidad de instancias corriendo sin ningún tipo de carga de usuarios, siendo una única instancia la que se ejecutaba en ese momento. El cuadro mostrado por Microsoft Azure se visualiza en la Figura 71.

Figura 71: Captura de sección de instancias en panel principal del recurso "Cloud Service"



Fuente: Elaboración propia

Con este escenario inicial, se procedió con la elaboración de las pruebas. Las pruebas se elaboraron empleando JMeter 4.0, la cual es una herramienta que permite las pruebas de carga de servicios web de forma modular y ofrece flexibilidad para las mismas; InfluxDB 1.5.4, que es un almacén de datos para casos de uso que impliquen grandes cantidades de datos con marcas de tiempo; y Grafana 5.2, el cual es un panel de visualización y monitoreo de data que maneja reportes en tiempo real.

Los valores ingresados en el módulo Concurrency Thread Group de Jmeter fueron:

- **Tasa de concurrencia:** 500 usuarios.
- **Tiempo de rampa (total en minutos):** 50 minutos.
- Número de pasos: **12.**

- **Tiempo de conexión de cada usuario (en minutos):** 40 minutos.

Las pruebas se realizaron durante un período de 50 minutos, considerando un total de 500 usuarios concurrentes. Estos usuarios fueron agregados a razón de 41 usuarios cada 4 minutos, en promedio, los cuales tuvieron un tiempo de conexión de 40 minutos. Este cálculo se obtuvo de la siguiente manera:

- 50 minutos totales de prueba, divididos entre 12 pasos, dan un promedio de 4 minutos por cada paso.
- 500 usuarios totales, divididos entre 12 pasos, dan un promedio de 41 usuarios por cada paso

Además, cada hilo generado por un usuario realizó un bucle de 5 consultas HTTP, las cuales siguieron realizándose mientras la sesión del mismo se mantuvo activa (durante 40 minutos cada uno). Como resultado del proceso de pruebas se obtuvo que, en un lapso de 20 minutos de pruebas, se realizaron 338,136 consultas al servicio, de las cuales sólo 2606 resultaron fallidas, dando una proporción de 99.23% de solicitudes completadas exitosamente, tal como se muestra en la Figura 72.

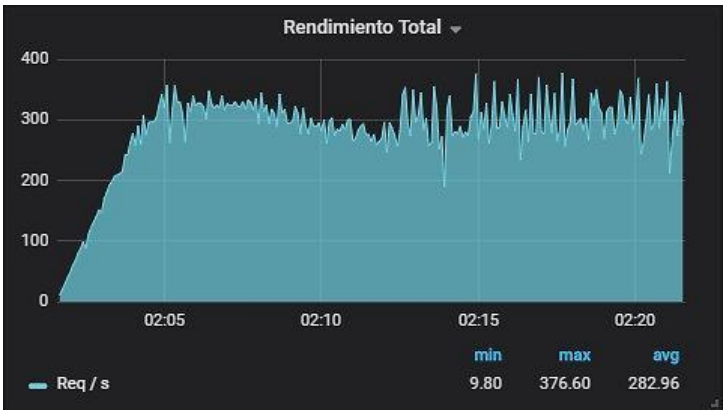
Figura 72: Captura de panel de Grafana: Métricas de errores vs consultas

Fuente: Elaboración propia

Durante los 20 primeros minutos, se realizaron solicitudes mediante 201 hilos (usuarios) conectados de forma concurrente, los cuales lanzaron un promedio de 282.96 solicitudes por segundo (con pico máximo de 376.60 y mínimo de 9.80). Asimismo, los errores que se registraron tuvieron un pico máximo de 67 errores por

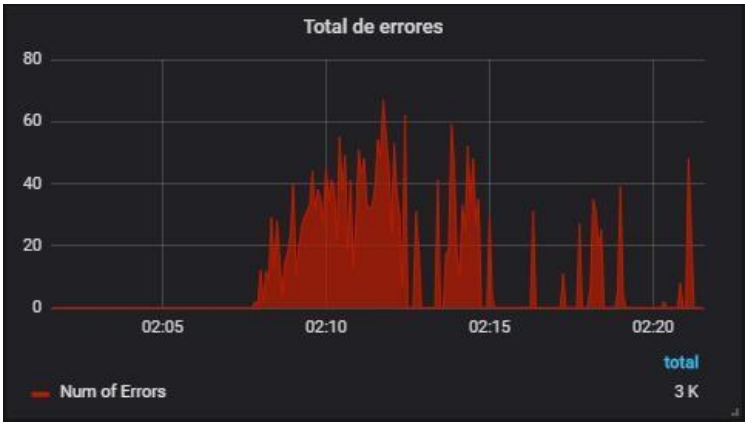
segundo y pico mínimo de 0 errores. Esta información se muestra en las Figura 73, Figura 74 y Figura 75:

Figura 73: Captura de panel de Grafana: Rendimiento total



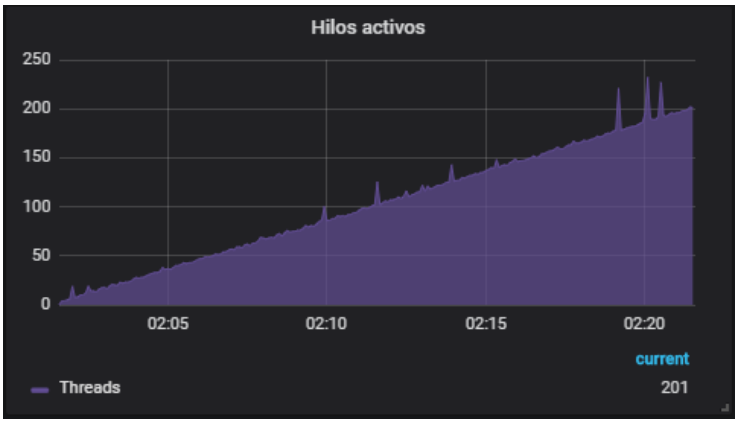
Fuente: Elaboración propia

Figura 74: Captura de panel de Grafana: Total de errores



Fuente: Elaboración propia

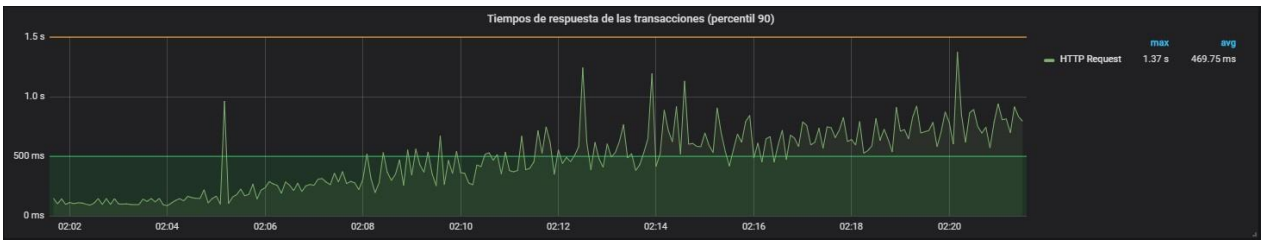
Figura 75: Captura de panel de Grafana: Hilos activos



Fuente: Elaboración propia

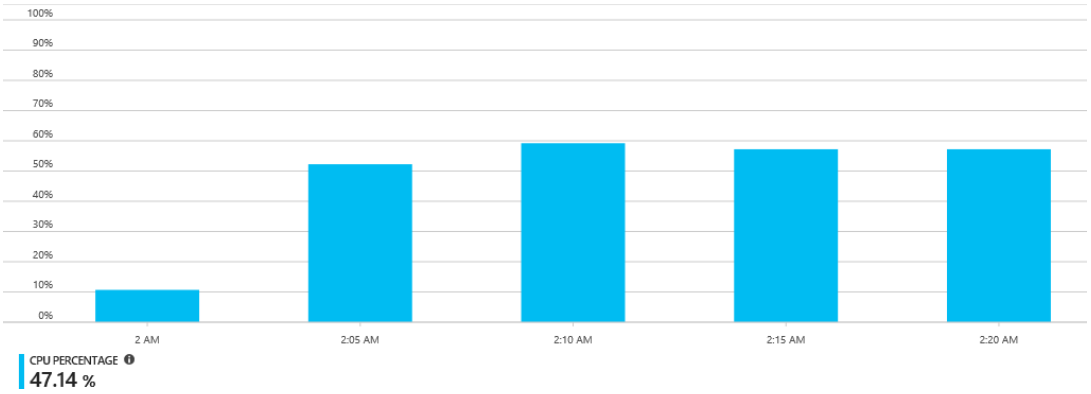
Además, se pudo extraer un gráfico de los tiempos de respuesta de las transacciones, considerando el percentil 90. De este gráfico se puede observar que ninguna transacción pasa el tiempo máximo aceptable de 1.5 segundos. El promedio de tiempo de respuesta por transacción ha sido calculado en 469.75, siendo el pico máximo el de 1.37 segundos por transacción (véase la Figura 76). Al mismo tiempo, se muestra en el gráfico del porcentaje de uso de CPU producido por Microsoft Azure en la Figura 77:

Figura 76: Captura de panel de Grafana: Tiempos de respuesta de transacciones



Fuente: Elaboración propia

Figura 77: Captura de Microsoft Azure: Porcentaje de uso del CPU del servicio



Fuente: Elaboración propia

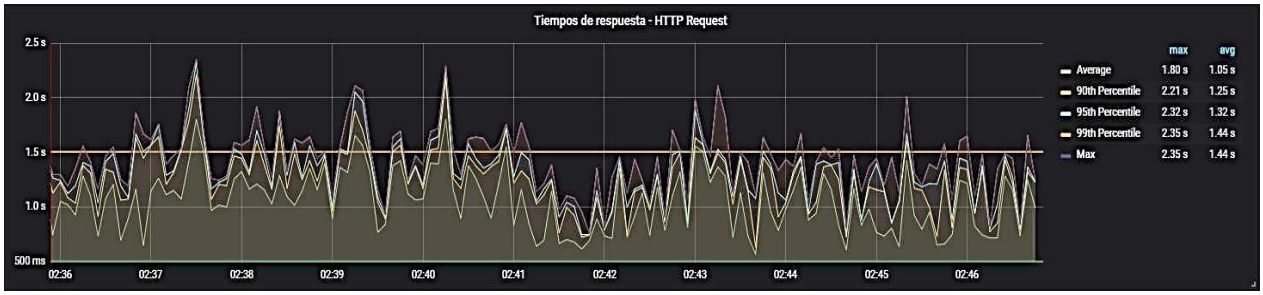
Se continuaron las pruebas, aumentando la cantidad de usuarios (como estaba previsto) e incrementando el uso de CPU del servidor destinado al servicio web. Entre los 350 y 450 usuarios concurrentes consumiendo el servicio, la tasa de errores tenía una razón de 42.32%, con 150,528 errores registrados de un total de 355,732 peticiones, todo en ese rango de usuarios (ver Figura 78). Además, los tiempos de

respuesta a las solicitudes dirigidas al servicio, se vieron incrementados drásticamente, alcanzando tasas de hasta 2.35 segundos de atención de una solicitud (ver Figura 79).

Figura 78: Captura de panel de Grafana: Estado de solicitudes e hilos con alta carga de usuarios

Fuente: Elaboración propia

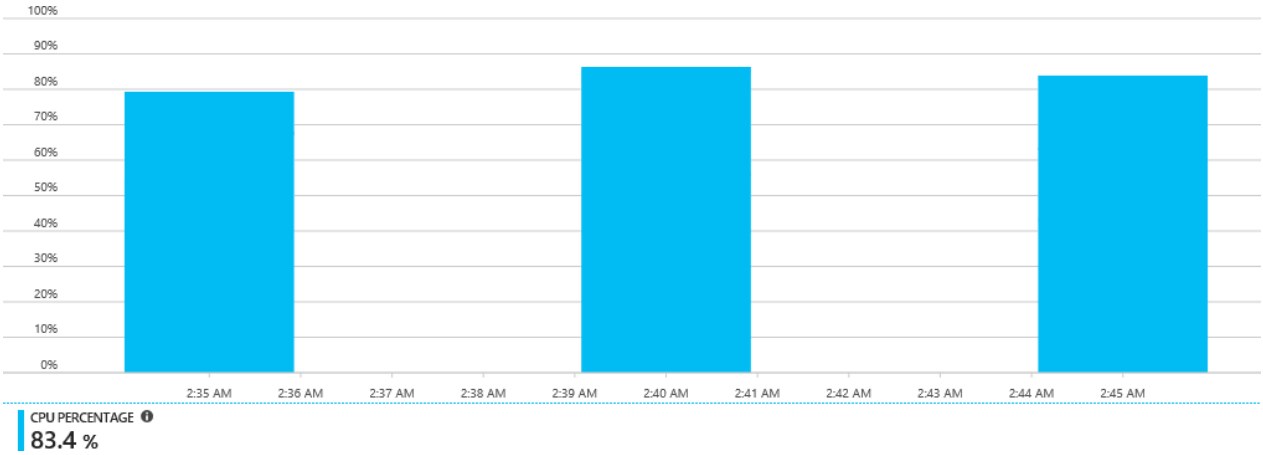
Figura 79: Captura de panel de Grafana: Tiempos de respuesta de consultas HTTP



Fuente: Elaboración propia

Asimismo, el uso del CPU se elevó a más del 80% (ver Figura 80), iniciando el escalamiento de la instancia del servicio configurado en Microsoft Azure. Esto obedeció a la primera regla de escalamiento creada, destinando un recurso de máquina virtual para hospedar la instancia con las mismas características que la instancia inicial. En este tipo de escalamientos, el balanceo de carga se da de forma automática:

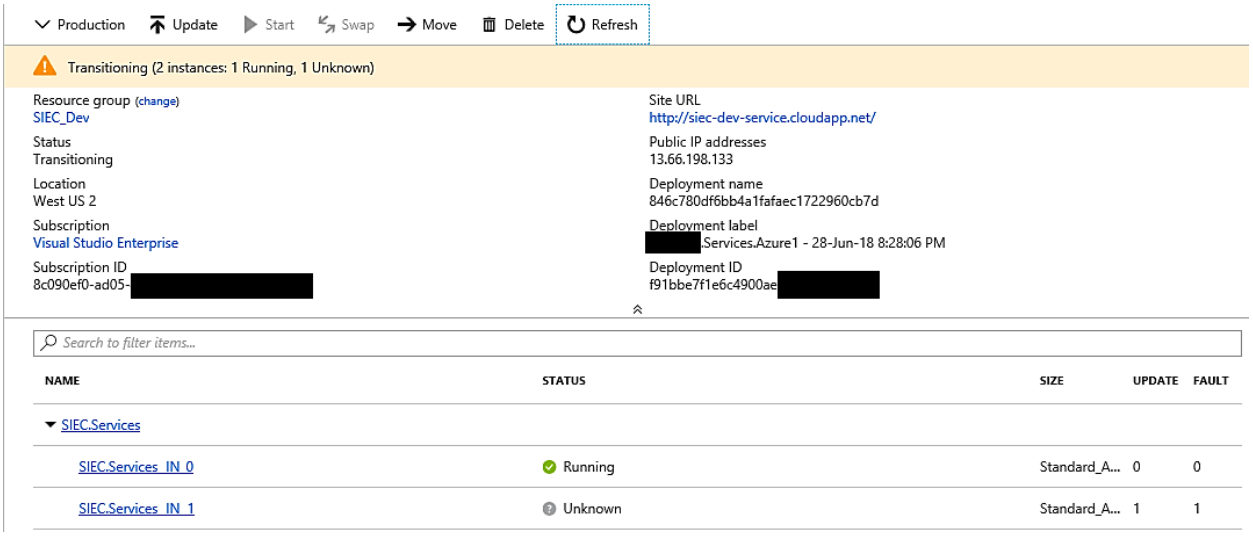
Figura 80: Captura de Microsoft Azure: Uso de CPU con una concurrencia de entre 350 y 450 usuarios



Fuente: Elaboración propia

El escalamiento del servicio produce una segunda instancia que se configura con los mismos parámetros con los que fue configurada la primera instancia, tomando un tiempo para finalizar su despliegue. Durante este proceso, la primera instancia sigue soportando el peso de las pruebas ejecutadas contra el servicio (ver Figura 81):

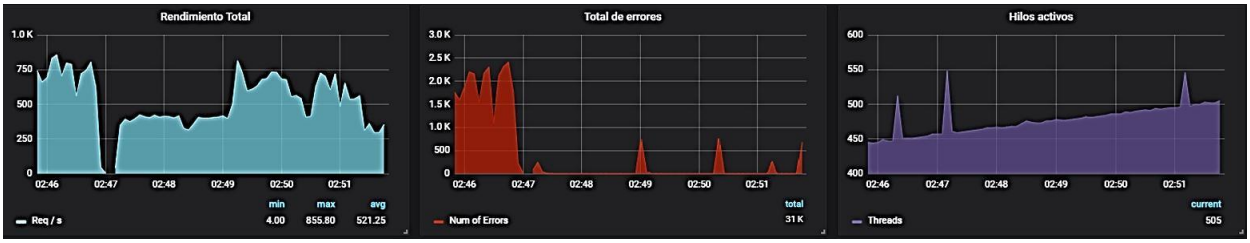
Figura 81: Captura de Microsoft Azure: Autoescalamiento de servicio web



Fuente: Elaboración propia

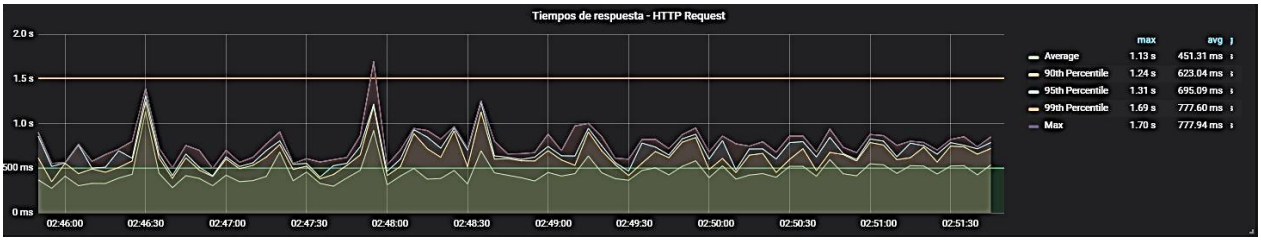
Tras producirse el escalamiento de forma exitosa, se bloqueó toda solicitud por unos milisegundos para luego volver a tomar las solicitudes que se estaban lanzando en las pruebas de forma exitosa. Durante los últimos cuatro minutos que duró la prueba, y en los que el escalamiento había sido efectivo, se lanzaron un total de 137,146 solicitudes contra las dos instancias del servicio, obteniendo un margen de error de 2.71%, con un total de 3712 solicitudes fallidas. La Figura 82 nos muestra que se enviaban entre 40.80 y 814.80 solicitudes por segundo con una cantidad de hilos concurrentes que oscilaba entre los 450 y 505. Asimismo, la Figura 83 nos indica que los tiempos de respuesta para las solicitudes disminuyeron considerablemente, ubicándose en una media de 451.31 milisegundos de respuesta y manteniendo estos valores estables hasta el final de las pruebas.

Figura 82: Captura de panel de Grafana: Estado de solicitudes e hilos tras escalamiento



Fuente: Elaboración propia

Figura 83: Captura de panel de Grafana: Tiempos de respuesta HTTP tras escalamiento



Fuente: Elaboración propia

CONCLUSIÓN

Como conclusión de estas pruebas de escalabilidad y rendimiento, podremos decir que el sistema cumple el objetivo de atender la demanda de usuarios para la cual fue pensado inicialmente con creces. Además, contempla escenarios de mayor demanda, llegando a soportar una concurrencia cinco veces mayor que la inicial. Esto se concluye debido a que soportó sin problemas una concurrencia de hasta 500 usuarios.

De los gráficos se extrae:

- Con una instancia se puede atender a la demanda pensada de 44 usuarios concurrentes a un promedio de 180ms por solicitud, a 320 solicitudes por segundo.
- Una sola instancia puede soportar hasta un máximo de 300 usuarios en promedio con una velocidad de casi 1000ms por solicitud, a una razón de casi 470 solicitudes concurrentes. En este punto es necesario el escalamiento, ya que la carga de la máquina que soporta el sistema es alta.

Tras realizar el escalamiento, la tasa de solicitudes simultáneas atendidas es realmente alta, llegando a más de 800 por segundo, con una cantidad de, en promedio, 460 usuarios concurrentes. El tiempo de respuesta para dichas solicitudes es 451.31ms en promedio.

Los tiempos de respuesta de las solicitudes cumplen con los estándares mínimos de 1 segundo, considerados para que el usuario considere a un sistema “fluido”. Además, cabe destacar que, en un escenario común, no debería presentarse una cantidad tan abismal de solicitudes por segundo ni de hilos concurrentes.

Por otro lado, es menester resaltar el cumplimiento del requisito de escalabilidad, el cual ha sido simplificado gracias al uso de la nube de Microsoft Azure, pero ha demostrado otorgar beneficios inmediatos y palpables. En este punto, es importante señalar que el uso de los archivos JSON de despliegue de recursos sirven también para realizar un escalamiento horizontal de las aplicaciones, pero de forma manual. Este archivo se muestra en el anexo A.6. Script de Automatización de despliegue.

CONCLUSIONES

Las conclusiones que se lograron obtener con el desarrollo del proyecto expuesto en este trabajo tesis fueron las siguientes:

1. Se ha logrado cubrir satisfactoriamente la característica de usabilidad del sistema, debido a que se ha cumplido con los criterios empleados para medir la usabilidad del sistema. Esto se respalda en los resultados: para navegación se obtuvo 41.05% de respuestas para “Muy de acuerdo” y 35.79% de respuestas para “Totalmente de acuerdo”; en el caso de accesibilidad, se obtuvo 38.6% de respuestas para “Muy de acuerdo” y 49.12% de respuestas para “Totalmente de acuerdo”; en el caso de funcionalidad, se obtuvo 21.05% de respuestas para “Muy de acuerdo” y 26.32% de respuestas para “Totalmente de acuerdo”; en el caso de ayuda, se obtuvo 42.11% de respuestas para “Muy de acuerdo” y 44.74% de respuestas para “Totalmente de acuerdo”; en el caso de prevención de errores, se obtuvo 34.21% de respuestas para “Muy de acuerdo” y 47.37% de respuestas para “Totalmente de acuerdo”.
2. Además, el objetivo de dotar al sistema de escalabilidad, rendimiento y funcionalidad se logró, ya que los resultados de las pruebas respaldan dicha afirmación: El sistema escaló duplicó la instancia que tenía inicialmente, pasando a soportar de 300 a 460 usuarios concurrentes, con una velocidad que ascendió de 1000ms a 470ms de respuesta para las más de 800 solicitudes concurrentes. Asimismo, se obtuvo un total de 96.94% de métodos con resultados esperados en el servicio desplegado.
3. Por lo tanto, al haber logrado los específicos, concluimos que el objetivo general, de analizar, diseñar e implementar un sistema de evaluación crediticia estudiantil en la nube para una ONG de la ciudad de Lima con características orientadas a la aceptación y variabilidad de demanda, también se ha logrado.

TRABAJOS FUTUROS

Como trabajos a futuro en este campo, queda pendiente:

1. La creación de los nuevos módulos de procesos internos, integrando las actividades manejadas por el sistema integral que poseen actualmente para así poder relegarlo de los flujos actuales. Esto, debido a que la idea principal era la de desarrollar un sistema que pudiera manejar y gestionar toda información generada por la ONG, objetivo que no lograron con la implementación de ese sistema integral.

2. Con respecto al servicio, se recomienda ampliar las posibilidades del postulante para que pueda conocer el estado actual de sus solicitudes conforme estas sigan dentro del proceso de evaluación. Además, es necesario tener en cuenta la ampliación de la cantidad de información solicitada, ya que se encuentran habilitadas en la base de datos las tablas para el guardado de información correspondiente a las empresas que deseen brindar acceso a los diplomados y cursos de maestría para sus colaboradores en las universidades con las que posee convenio la ONG, pero durante el análisis del sistema, y de acuerdo a la comunicación con el equipo técnico de la ONG, no estaría contemplado en esta primera etapa.

3. Asimismo, se recomienda la creación de un API a nivel del servicio, en un nuevo contrato del mismo, para el acceso a la información crediticia de los postulantes por parte de las entidades bancarias con las que se estén trabajando.

4. Considerando que la ONG trabaja con la familia de soluciones Microsoft, incluidos los servicios de Microsoft Office y las plataformas Windows, se recomienda comunicar el servicio de Active Directory, manejado por el servidor On-Premise que poseen, con el servicio de Azure Active Directory para la gestión de las identidades de usuario, y extenderla a los dispositivos y sistemas para un mejor seguimiento de las actividades y asignación de responsabilidades de los colaboradores, siendo posible integrar a la aplicación desarrollada, así como a futuras implementaciones de nuevos sistemas y módulos, un canal de comunicación directo con ese servicio de Active Directory para establecer un método de logueo de SSO (Single Sign-On). De esta forma, se gestionaría con mayor eficacia y eficiencia los protocolos de seguridad, los permisos y los usuarios admitidos a nivel interno y de cara a los agentes externos involucrados en sus procesos.

5. Como se observó en los resultados de las encuestas y entrevistas, el control interno en las MYPES comerciales de Huancayo se encuentran en un estado crítico y casi inexistente, motivo por el que requieren de ayuda para mejorarlo o implementarlo.

6. Acorde a los resultados obtenidos en las encuestas y entrevistas brindadas a las MYPES comerciales de Huancayo, se concluye que existe una cantidad significativa de MYPES (77.72%) que tienen problemas organizativos a causa de no implementar adecuadamente sus controles.

7. La estrategia de ambiente de control en las MYPES comerciales de Huancayo está aplicada ineficientemente, se encuentran en un estado crítico y requiere de atención para mejorar la calidad de sus controles en el 79.75% de ellos. El microempresario huancaíno desarrolla sus operaciones a la deriva, no tiene una planificación, ni se proyecta a futuro.

8. Basado en la información recopilada por la encuestas, los riesgos más comunes en las MYPES comerciales de Huancayo son: no saber identificar sus riesgos relevantes (78.51%), no tener buenos controles administrativos (57.03%) y tomar malas decisiones (16.45%).

9. Fundamentado en la información recopilada por la encuestas y entrevistas, se concluye que los instrumentos de información y comunicación que usan con frecuencia las MYPES comerciales de Huancayo son la libreta de apuntes (Figura n° 26) y el diálogo (80.37%).

10. Luego de desarrollar e implementar un instrumento a raíz de las problemáticas que tienen las MYPES comerciales huancaínas, se concluye que los cuadros sinópticos creados del control interno en base al COSO 92 permiten mejorar y facilitar las funciones organizacionales.

REFERENCIAS BIBLIOGRÁFICAS

1. **Sanabria Romero, Alejandro Arturo y Tascón Millán, Jessica Mercedes.** *Diseño e implementación de la arquitectura orientada a servicios (SOA), en el desarrollo de un software para la empresa CONCIVIN LTDA.* Bogotá : Universidad San Buenaventura, 2008.

2. **Arévalo Navarro, José Manuel.** *Cloud Computing: fundamentos, diseño y arquitectura aplicados a un caso de estudio.* Madrid : Universidad Rey Juan Carlos, 2011.

3. **Cedillo Crisosto, Franco Eduardo.** *Análisis, diseño, implementación e integración de un sistema de gestión de casos y un SoftPhone Web para un centro de contacto virtual con múltiples casos de comunicación.* Lima : Pontificia Universidad Católica del Perú, 2008.

4. **Lapiedra Alcamí, Rafael, Devece Carañana, Carlos y Guiral Herrando, Joaquín.** *Introducción a la gestión de sistemas de información en la empresa.* 1ra. Castellón de la Plana : Publicaciones de la Universidad Jaime I, 2011. págs. 13-26. 978-84-693-9894-4.

5. **Cohen Karen, Daniel y Asín Lares, Enrique.** *Sistemas de información para los negocios. Un enfoque de toma de decisiones.* México D.F. : McGraw - Hill/Interamericana Editores S.A., 2000. págs. 9-13. ISBN: 970-10-2658-6.

6. **Plancarte Sanchez, Federico.** Sistemas de información estratégica para la competitividad. *GestioPolis.* [En línea] 1 de 10 de 2007. [Citado el: 15 de 03 de 2018.] https://www.gestiopolis.com/sistemas-de-informacion-estrategica-para-la-competitividad/.

7. **Mell, Peter y Grance, Timothy.** *NIST Special Publication 800-145: The NIST Definition of Cloud.* Gaithersburg : National Institute of Standards and Technology, 2011.

8. *Cloud Computing Architecture & Services.* **Shah, Manan y Shiman, Charusmita.** 11, Bhopal : International Journal of Computer Science and Mobile Computing, 2015, Vol. 4. ISSN 2320–088X.

9. *The Cloud Computing: A Systematic Review.* **Oqail Ahmad, Mohammad y Zaman Khan, Rafiqul.** 5, Aligarh : International Journal of Innovative Research in Computer and Communication Engineering, 2015, Vol. 3. ISSN: 2320-9798.

10. **Hamdaqa, Mohammad y Tahvildari, Ladan.** Cloud Computing Uncovered: A Research Landscape. [aut. libro] Ali Hurson. *Advances in Computers, Volume 86.* Waterloo : University of Waterloo, 2012.

11. *Auto Scaling in Cloud Computing: An Overview.* **Kriushanth, M., Justy Mirobi, G. y Arockiam, L.** 7, Bhopal : International Journal of Advanced Research in Computer and Communication Engineering, 2013, Vol. II. ISSN: 2319-5940.

12. **Dean, John y Dean, Raymond.** *Introducción a la programación con Java.* 1ra. México D.F. : McGraw-Hill/Interamericana Editores, S.A. de C.V., 2009. ISBN: 978-970-10-7278-3.

13. **Velarde de Barraza, Olinda, y otros.** *Introducción a la programación orientada a objetos.* 1ra. Naucalpan de Juárez : Pearson Educación de México, S.A., 2006. págs. 8-10. ISBN: 970-26-0887-2.

14. **Krafzig, Dirk, Banke, Karl y Slama, Dirk.** *Enterprise SOA: Service-Oriented Architecture Best Practices.* Hagerstown, Maryland : Prentice Hall PTR, 2005. ISBN: 0-13-146575-9.

15. **Microsoft Corporation.** *Enterprise Solution Patterns Using Microsoft .NET.* s.l. : Microsoft Press, 2003.

págs. 267-268. Vol. II. ISBN: 978-0735618398.

16. **Rodríguez Sala, Jesús Javier, y otros.** *Introducción a la programación. Teoría y práctica.* Alicante : Editorial Club Universitario, 2003. ISBN: 84-8454-274-2.

17. **Microsoft Corporation.** Introduction to object storage in Azure. *Microsoft Docs.* [En línea] 27 de 03 de 2018. [Citado el: 16 de 05 de 2018.] https://docs.microsoft.com/en-us/azure/storage/blobs/storage-blobs-introduction.

18. **Quin, Liam.** Extensible Markup Language (XML). *W3C.* [En línea] W3C, 11 de 10 de 2016. [Citado el: 12 de 05 de 2018.] https://www.w3.org/XML/.

19. **Ambler, Scott.** The Agile Unified Process (AUP). *AmbySoft.* [En línea] AmbySoft. Scott Ambler + Associates, 2005. [Citado el: 17 de 04 de 2018.] http://www.ambysoft.com/unifiedprocess/agileUP.html.

20. **Microsoft Corporation.** .NET Framework Guide. *Microsoft Docs.* [En línea] Microsoft Corporation, 10 de 04 de 2018. [Citado el: 10 de 04 de 2018.] https://docs.microsoft.com/en- us/dotnet/framework/.

21. —. Chapter 1: Introduction to .NET . *TechNet.* [En línea] Microsoft Coporation, 31 de Mayo de 2006. [Citado el: 04 de 15 de 2018.] https://technet.microsoft.com/en-us/library/bb496996.aspx.

22. **Rodríguez Sala, Jesús Javier, y otros.** *Introducción a la programación. Teoría y práctica.* Alicante : Editorial Club Universitario, 2003. ISBN: 84-8454-274-2.

23. **Microsoft Corporation.** Introduction to object storage in Azure. *Microsoft Docs.* [En línea] 27 de 03 de 2018. [Citado el: 16 de 05 de 2018.] https://docs.microsoft.com/en-us/azure/storage/blobs/storage-blobs-introduction.

24. **Quin, Liam.** Extensible Markup Language (XML). *W3C.* [En línea] W3C, 11 de 10 de 2016. [Citado el: 12 de 05 de 2018.] https://www.w3.org/XML/.

25. **Ambler, Scott.** The Agile Unified Process (AUP). *AmbySoft.* [En línea] AmbySoft. Scott Ambler + Associates, 2005. [Citado el: 17 de 04 de 2018.] http://www.ambysoft.com/unifiedprocess/agileUP.html.

26. **Microsoft Corporation.** .NET Framework Guide. *Microsoft Docs.* [En línea] Microsoft Corporation, 10 de 04 de 2018. [Citado el: 10 de 04 de 2018.] https://docs.microsoft.com/en- us/dotnet/framework/.

27. —. Chapter 1: Introduction to .NET . *TechNet.* [En línea] Microsoft Coporation, 31 de Mayo de 2006. [Citado el: 04 de 15 de 2018.] https://technet.microsoft.com/en-us/library/bb496996.aspx.

ANEXOS

ANEXO 1: ESTRUCTURA DE LA SOLUCIÓN

El desarrollo de la solución para la ONG, presenta como entregable un directorio de proyectos, las bibliotecas externas necesarias y los scripts de creación de base de datos.

A continuación, se listan los proyectos contenidos en la solución:

Tabla A.1: Estructura de repositorios finales

NOMBRE DEL PROYECTO	DESCRIPCIÓN
SIEC.AlertJob	Proyecto para el envío de notificaciones automáticas que se emiten diariamente.
SIEC.BE	Proyecto transversal, que contiene las entidades utilizadas por todas las capas del servicio.
SIEC.BL	Capa de lógica de negocio, desde donde se hacen llamadas a las funciones del sistema.
SIEC.Common	Proyecto transversal, que contiene los valores constantes para el proyecto y la lista de enumerables.
SIEC.DA	Capa de acceso a datos, configurada con Enterprise Library.
SIEC.Services	Proyecto central del servicio. Contiene el contrato del servicio y los valores necesarios para la generación de documentos y envío de correo electrónico y mensajes de texto.
SIEC.Services.Azure	Proyecto que contiene el perfil de publicación del servicio web a la nube de Microsoft Azure.
SIEC.Web	Proyecto MVC, que contiene la capa de presentación y de aplicación de la presentación, que se comunica con el servicio desarrollado.

Fuente: Elaboración propia

ANEXO 2: CASOS DE PRUEBA

Se mostrarán dos casos de pruebas de aceptación por módulo, para los dos módulos mostrados a continuación:

- Módulo de Pre-Registro
- Módulo de Registro

Módulo de Pre-Registro.

Para realizar las siguientes pruebas, es necesario que el usuario que ingrese cuente con el Perfil de “Asesor”, o que su perfil asociado tenga acceso al módulo de Pre-Registro. Una vez ingresado al sistema se procederá a seleccionar la opción “Pre-Registro”, ello nos direccionará a una nueva pantalla donde se visualizarán dos opciones: Registro Individual y masivo.

ADMINISTRACIÓN DE REGISTRO INDIVIDUAL

Tabla A.2: Caso de Prueba: Administración de registro individual

NOMBRE DEL CASO DE PRUEBA:	**CP001 – ADMINISTRACIÓN DE REGISTRO INDIVIDUAL (Acciones de registrar postulante)**		
FLUJO DE PASOS DE LA PRUEBA:			
Nro.	**Pasos**	**Resultado Esperado**	**Resultado obtenido**
1	Seleccionar la opción “Registro Individual”	El sistema deberá mostrar en pantalla el formulario con los datos solicitados para hacer el registro.	
2	Llenar los campos solicitados de acuerdo al formulario.	Si no se rellenó los campos, el sistema deberá mostrar alertas de “Campo Obligatorio”.	
3	Hacer clic sobre el botón “Registrar”.	Al realizar dicha acción se mostrará un mensaje de confirmación.	
Decisión de Aprobación del Caso de Prueba: Aprobó:___Falló: ____			

Fuente: Elaboración propia

ADMINISTRACIÓN DE REGISTRO MASIVO

Tabla A.2: Caso de Prueba: Administración de registro individual

NOMBRE DEL CASO DE PRUEBA:	**CP002 – ADMINISTRACIÓN DE REGISTRO MASIVO (Acciones de carga masiva)**		
FLUJO DE PASOS DE LA PRUEBA:			
Nro.	**Pasos**	**Resultado Esperado**	**Resultado obtenido**
1	Seleccionar la opción “Registro Masivo”	El sistema deberá mostrar en pantalla el formulario con los datos solicitados para hacer el registro.	
2	Hacer clic sobre el botón “Cargar”.	Al realizar dicha acción se mostrará un examinador para seleccionar el archivo (Excel), para la carga masiva de postulantes. **Nota:** El archivo Excel tendrá un formato establecido de acuerdo a los datos que se desea obtener de los postulantes. Ya cargado el archivo se mostrará un mensaje de confirmación.	
3	Formato de Excel	Antes de llenar los datos de los postulantes, se procederá a hacer el llenado de las listas desplegables en el Excel, tales como: Tipo de Documento, Estado Civil, Programa, Modalidad de Crédito. **(Las cuales se llenarán en la hoja número 2)** **Campos para llenar directamente:** Nombres, Apellido Materno, Apellido Paterno, Número de Documento, Email, Teléfono. **Nota:** El formato se adjuntará con la documentación de pruebas.	
Decisión de Aprobación del Caso de Prueba: Aprobó:___Falló: ____			

Fuente: Elaboración propia

Módulo de Registro.

Para realizar las siguientes pruebas, es necesario que el usuario que ingrese cuente con el Perfil de “Postulante”, o que su perfil asociado tenga acceso al módulo de Registro. Una vez ingresado al sistema se procederá a seleccionar la opción “Registro”, ello nos direccionará a una nueva pantalla donde se visualizarán dos opciones: Registro Individual y masivo.

ADMINISTRACIÓN DE INFORMACIÓN PERSONAL

Tabla A.3: Caso de Prueba: Administración de información personal

NOMBRE DEL CASO DE PRUEBA:	**CP003 – ADMINISTRACIÓN DE INFORMACIÓN PERSONAL (Acciones de actualizar y registro)**		
FLUJO DE PASOS DE LA PRUEBA:			
Nro.	**Pasos**	**Resultado Esperado**	**Resultado obtenido**
1	Seleccionar la opción “Información Personal”	El sistema deberá mostrar en pantalla dos formularios: • Primero, mostrará los datos del postulante que se hicieron en el Pre-Registro. • Segundo, mostrará un formulario con los datos solicitados para que se proceda con la evaluación.	
2	Hacer clic sobre el icono con forma de lápiz para editar los datos del Pre-Registro del Postulante.	El sistema deberá mostrar un formulario con los datos correspondientes al Pre-Registro. Dentro de este formulario el usuario podrá realizar las modificaciones que cree convenientes.	
3	Llenar los campos con los datos solicitados en el segundo formulario. y presionar el botón guardar.	Al realizar dicha acción se mostrará un mensaje de confirmación.	
Decisión de Aprobación del Caso de Prueba: Aprobó:___Falló: ____			

Fuente: Elaboración propia

ADMINISTRACIÓN DE MOVIMIENTOS

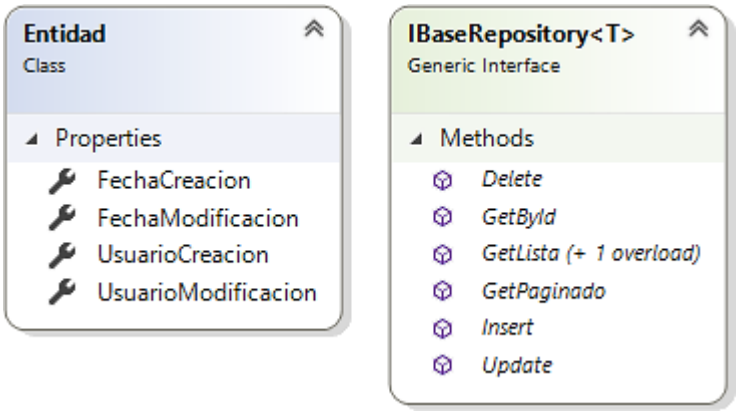
Tabla A.4: Caso de Prueba: Administración de movimientos

NOMBRE DEL CASO DE PRUEBA:	**CP003 – ADMINISTRACIÓN DE MOVIMIENTOS (Acciones de Registro y eliminación)**		
FLUJO DE PASOS DE LA PRUEBA:			
Nro.	**Pasos**	**Resultado Esperado**	**Resultado obtenido**
1	Seleccionar la opción “Movimientos”.	El sistema deberá mostrar en pantalla dos secciones: • Ingresos: Mostrará una lista desplegable con los tipos de ingresos permitidos, además de una caja de texto para el ingreso de monto. • Listado de ingresos en caso se haya registrado anteriormente. • Egresos: Mostrará una lista desplegable con los tipos de egresos permitidos, además de una caja de texto para el ingreso de monto. • Listado de egresos en caso se haya registrado anteriormente.	
2	Hacer clic sobre el botón “Guardar”. **Nota:** Sección de Ingresos.	Al realizar dicha acción por defecto se actualizará el listado de ingresos.	
3	Hacer clic sobre el botón “Guardar”. **Nota:** Sección de Egresos.	Al realizar dicha acción por defecto se actualizará el listado de egresos.	
4	Eliminación de Registros.	Tanto para ambas secciones, primero se debe seleccionar con el radio buton para así poder realizar la eliminación de forma correcta.	
Decisión de Aprobación del Caso de Prueba: Aprobó:___Falló: ____			

Fuente: Elaboración propia

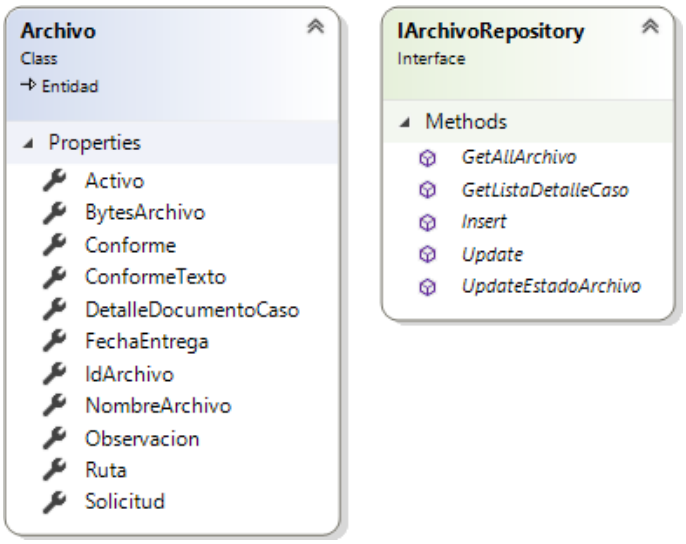
CLASES E INTERFACES

Las clases presentadas a continuación, acompañadas de las interfaces que describen sus métodos implementados, corresponden al sistema de evaluación crediticia, desarrollado en C#, siendo extraídas desde la aplicación Visual Studio.Clase Entidad e interfaz IBaseRepository



Fuente: Elaboración propia

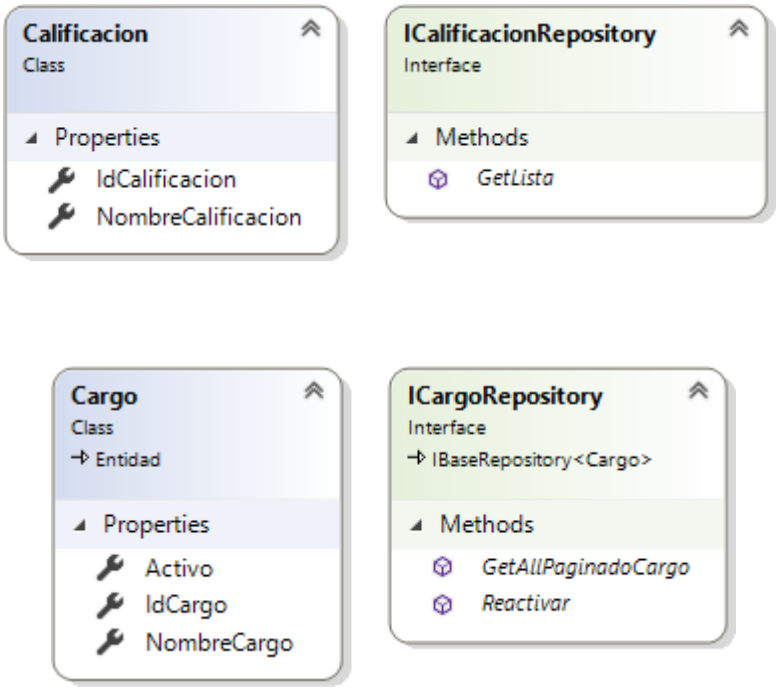
Clase Archivo e interfaz IArchivoRepository



Fuente: Elaboración propia

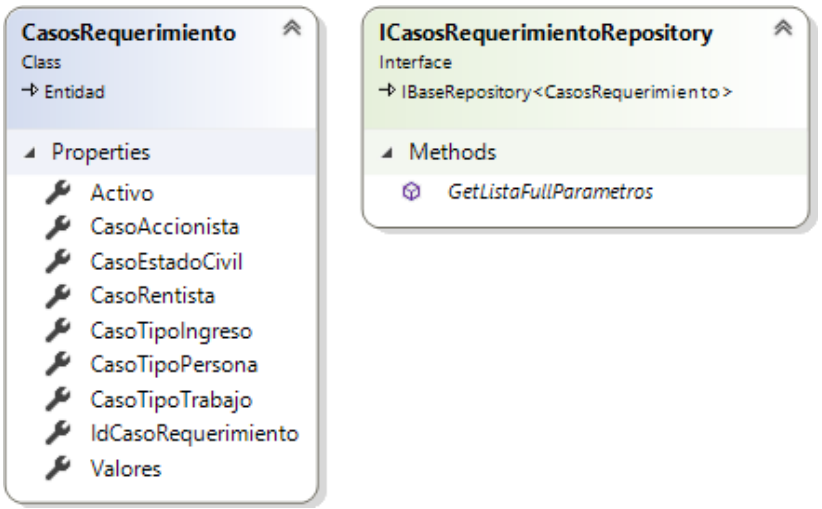
CLASE CALIFICACIÓN E INTERFAZ ICALIFICACIONREPOSITORY

Clase Cargo e interfaz ICargoRepository



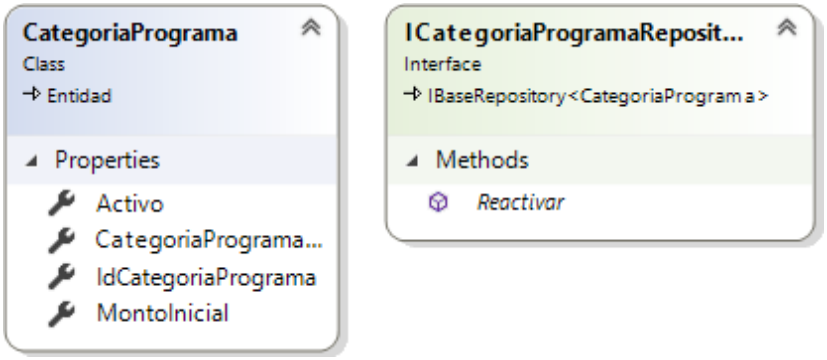
Fuente: Elaboración propia

CLASE CASOSREQUERIMIENTO E INTERFAZ ICASOSREQUERIMIENTOREPOSITORY

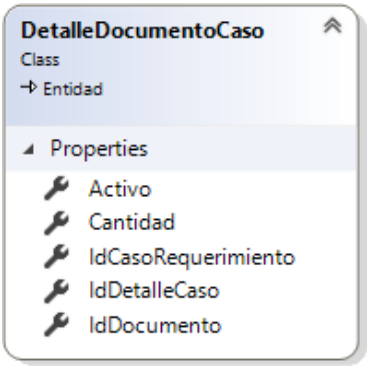


Fuente: Elaboración propia

CLASE CATEGORIAPROGRAMA E INTERFAZ ICATEGORIAPROGRAMAREPOSITORY

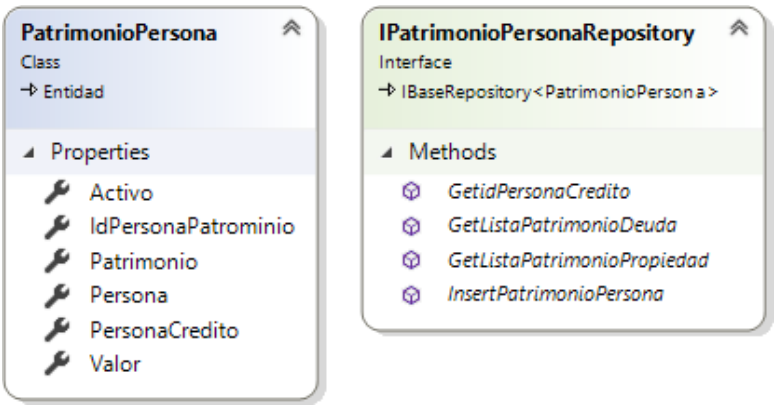


Clase DetalleDocumentoCaso



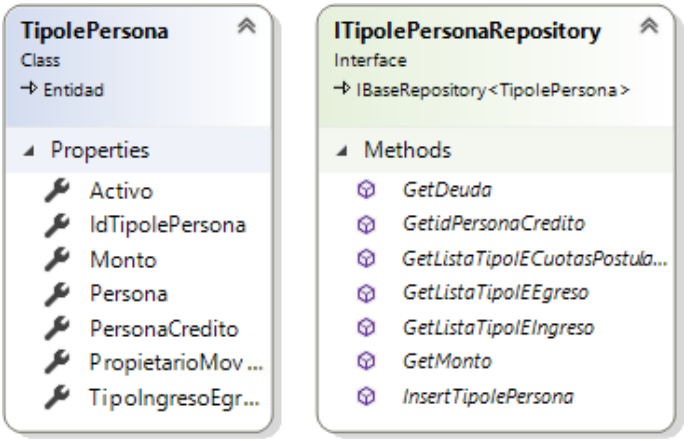
Fuente: Elaboración propia

CLASE PATRIMONIOPERSONA E INTERFAZ IPATRIMONIOPERSONAREPOSITORY

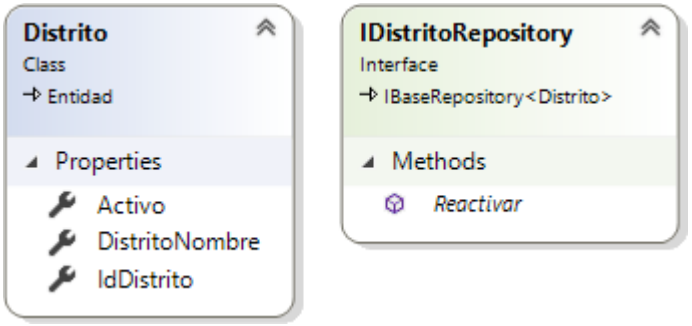


Fuente: Elaboración propia

CLASE TIPOIEPERSONA E INTERFAZ ITIPOIEPERSONAREPOSITORY

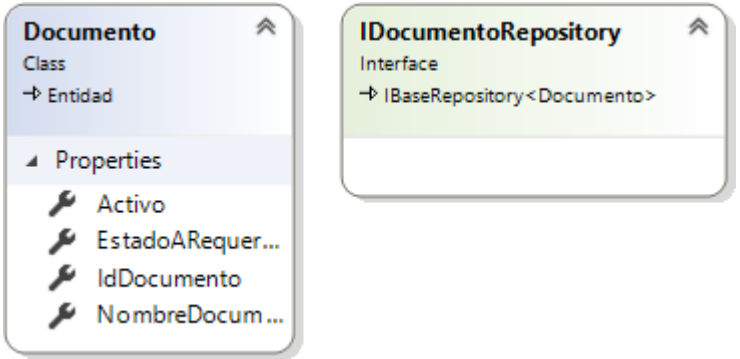


Clase Distrito e interfaz IDistritoRepository



Fuente: Elaboración propia

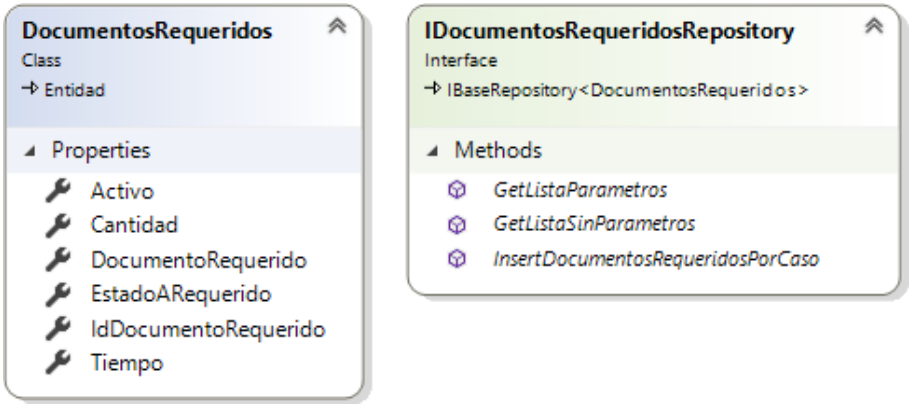
CLASE DOCUMENTO E INTERFAZ IDOCUMENTOREPOSITORY



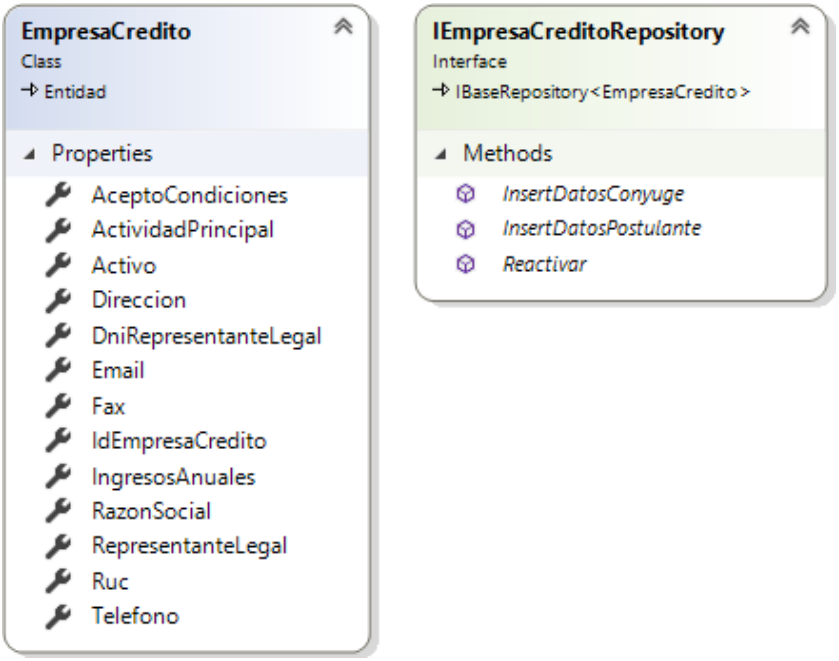
Fuente: Elaboración propia

CLASE DOCUMENTOSREQUERIDOS E INTERFAZ IDOCUMENTOSREQUERIDOSREPOSITORY

Fuente: Elaboración propia

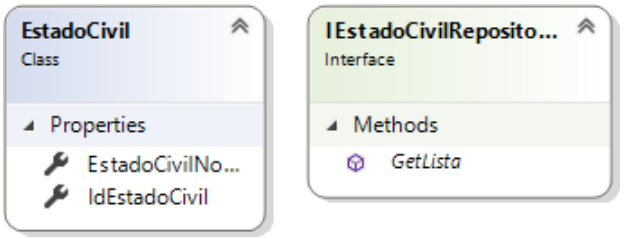


CLASE EMPRESACREDITO E INTERFAZ IEMPRESACREDITOREPOSITORY



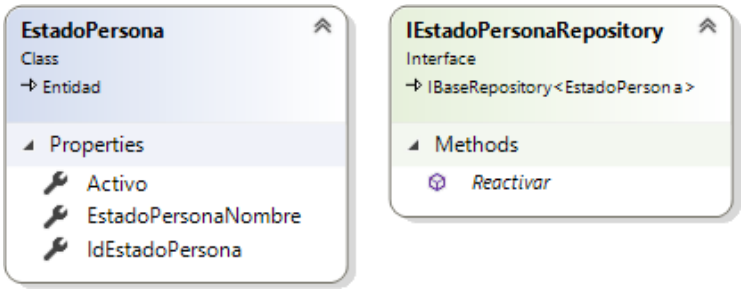
Fuente: Elaboración propia

CLASE ESTADOCIVIL E INTERFAZ IESTADOCIVILREPOSITORY



Fuente: Elaboración propia

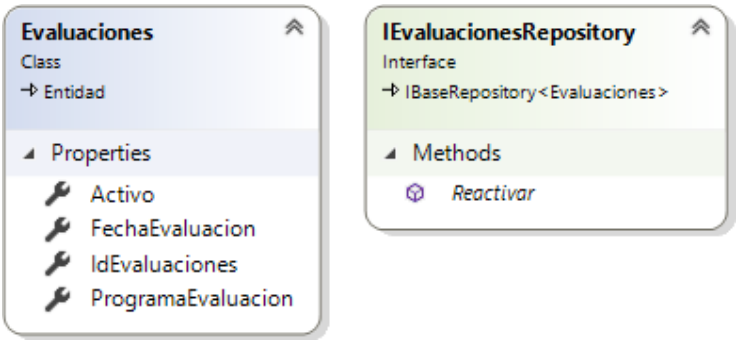
CLASE ESTADOPERSONA E INTERFAZ IESTADOPERSONAREPOSITORY



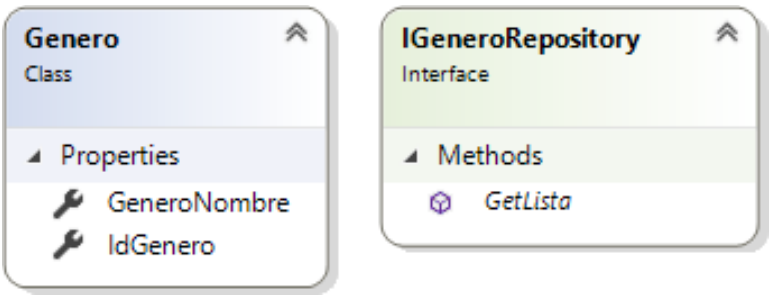
Fuente: Elaboración propia

CLASE EVALUACIONES E INTERFAZ IEVALUACIONESREPOSITORY

Fuente: Elaboración propia

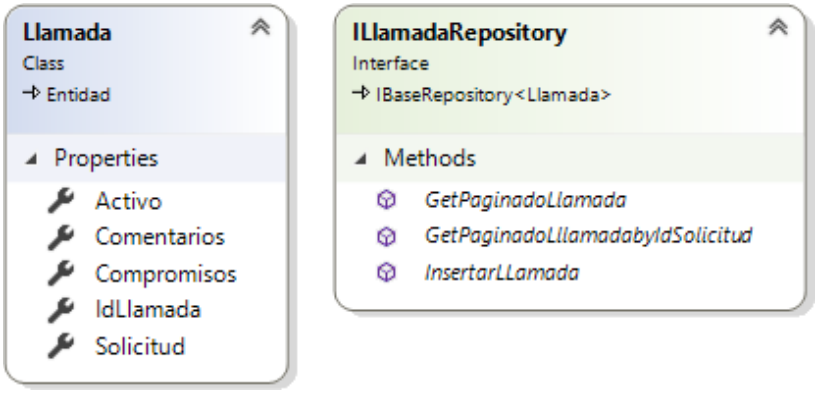


CLASE GENERO E INTERFAZ IGENEROREPOSITORY



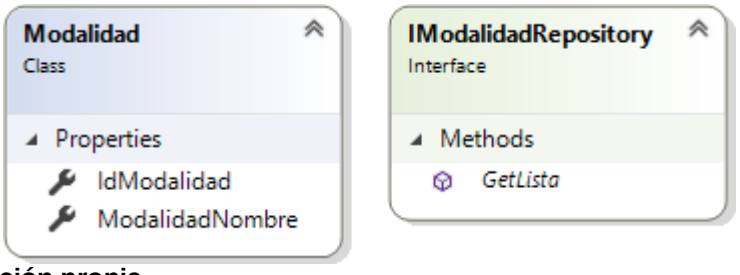
Fuente: Elaboración propia

CLASE LLAMADA E INTERFAZ ILLAMADAREPOSITORY



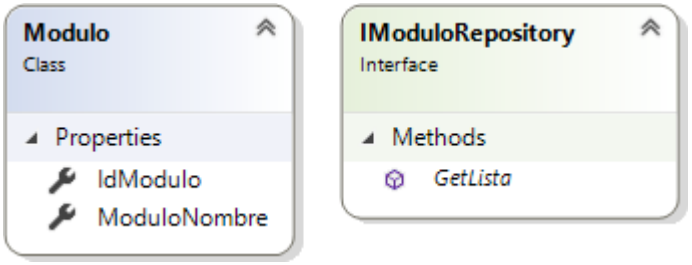
Fuente: Elaboración propia

CLASE MODALIDAD E INTERFAZ IMODALIDADREPOSITORY



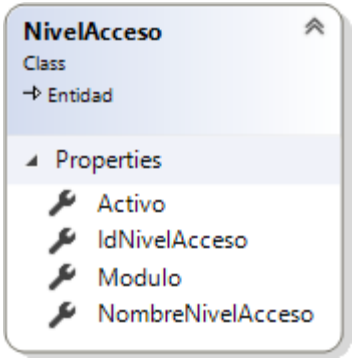
Fuente: Elaboración propia

CLASE MODULO E INTERFAZ IMODULOREPOSITORY



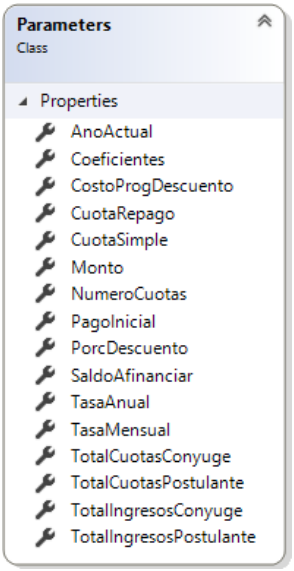
Fuente: Elaboración propia

CLASE NIVELACCESO



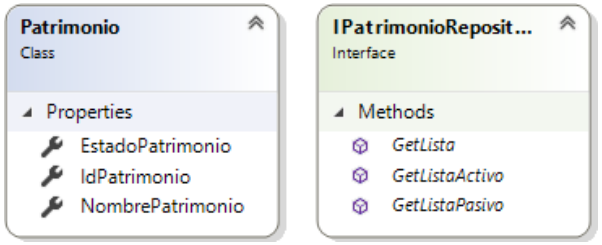
Fuente: Elaboración propia

CLASE PARAMETERS



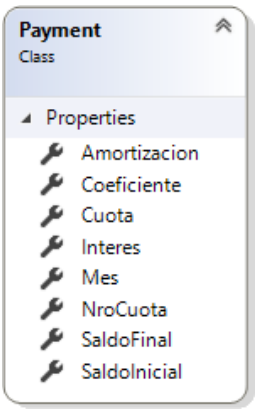
Fuente: Elaboración propia

CLASE PATRIMONIO E INTERFAZ IPATRIMONIOREPOSITORY



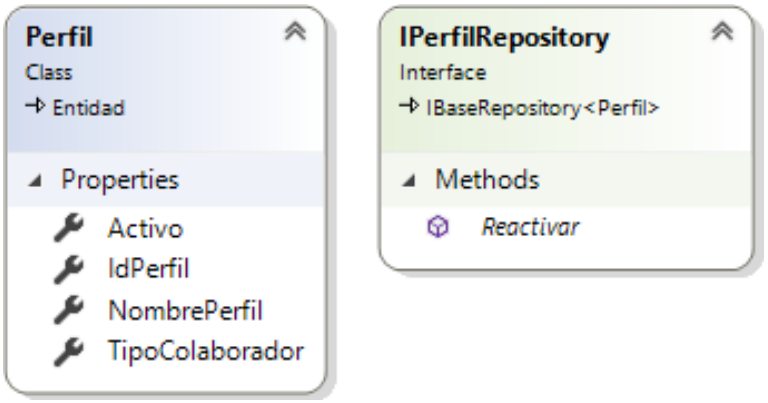
Fuente: Elaboración propia

CLASE PAYMENT



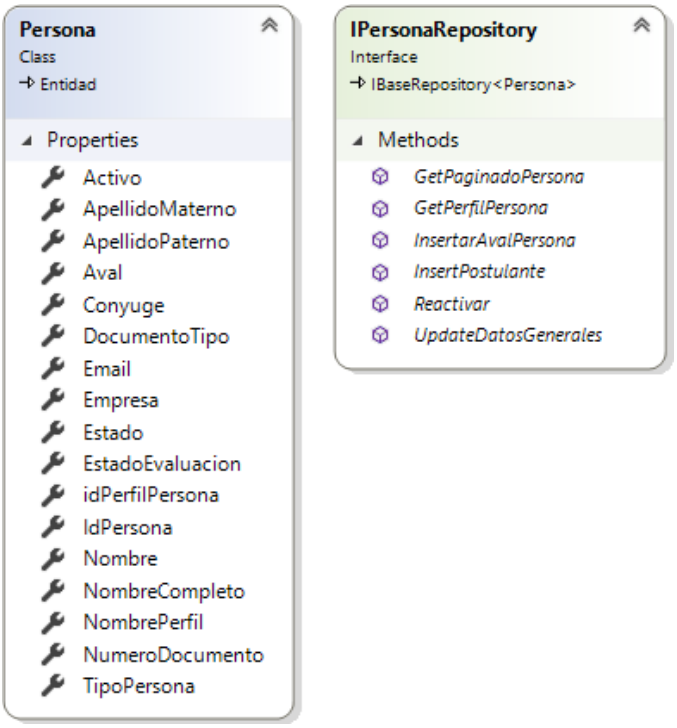
Fuente: Elaboración propia

CLASE PERFIL E INTERFAZ IPERFILREPOSITORY



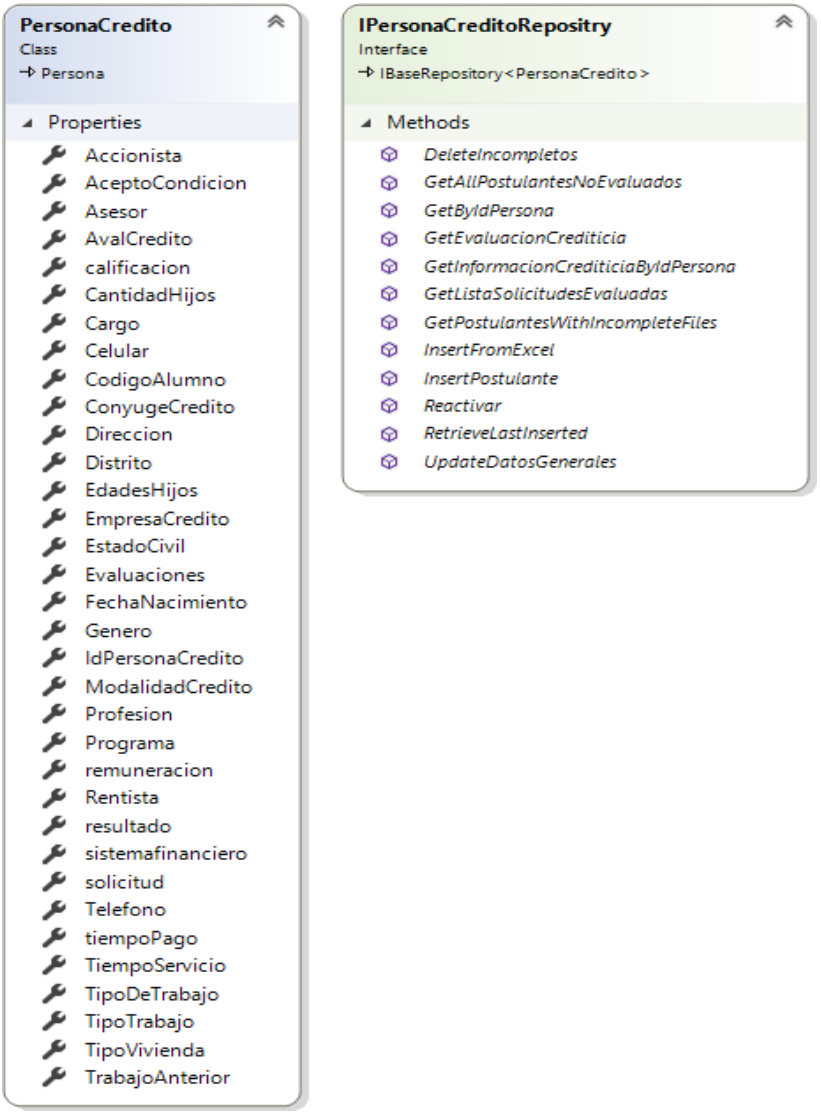
Fuente: Elaboración propia

CLASE PERSONA E INTERFAZ IPERSONAREPOSITORY



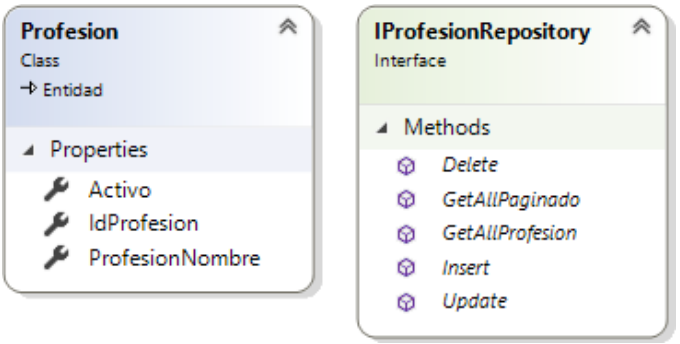
Fuente: Elaboración propia

CLASE PERSONACREDITO E INTERFAZ IPERSONACREDITOREPOSITORY



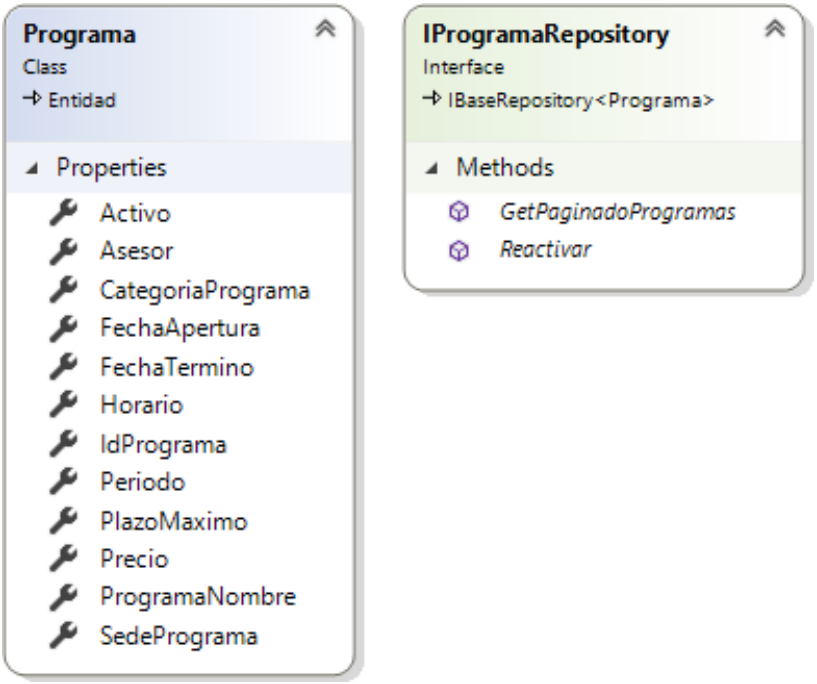
Fuente: Elaboración propia

CLASE PROFESION E INTERFAZ IPROFESIONREPOSITORY



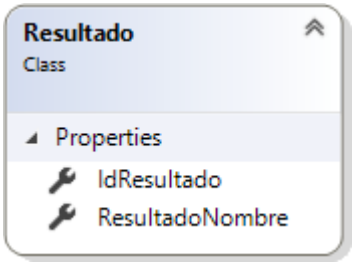
Fuente: Elaboración propia

CLASE PROGRAMA E INTERFAZ IPROGRAMAREPOSITORY



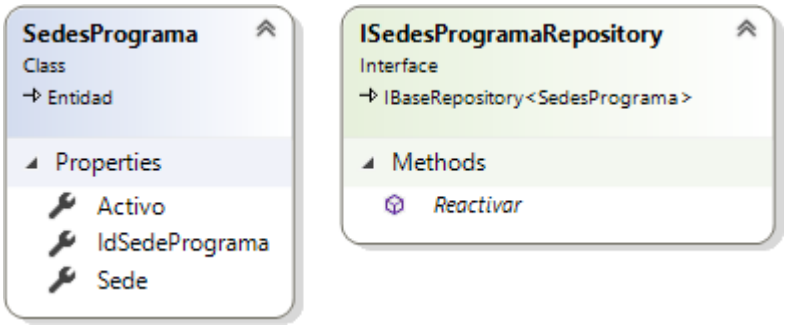
Fuente: Elaboración propia

CLASE RESULTADO



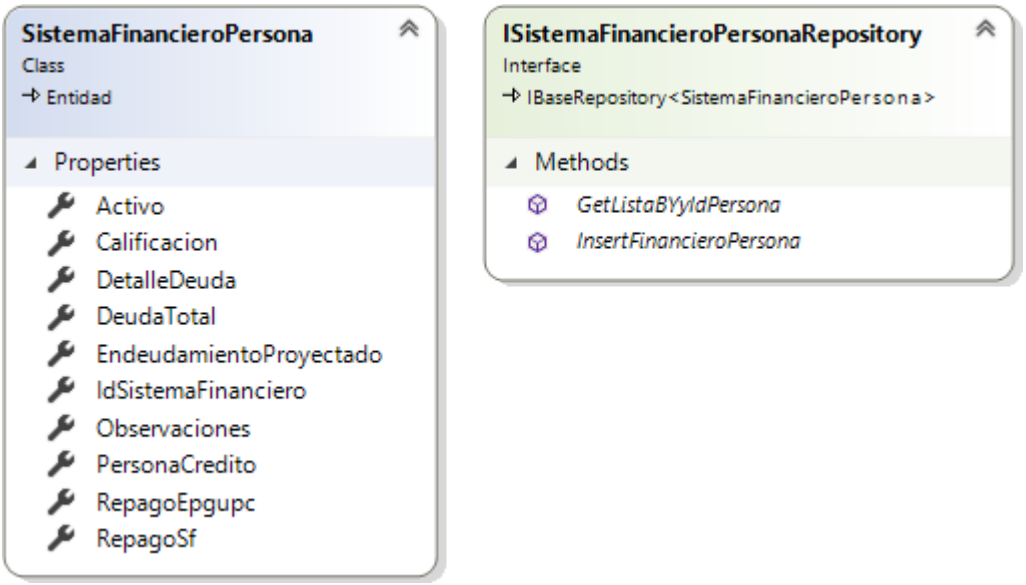
Fuente: Elaboración propia

CLASE SEDESPROGRAMA E INTERFAZ ISEDESPROGRAMAREPOSITORY



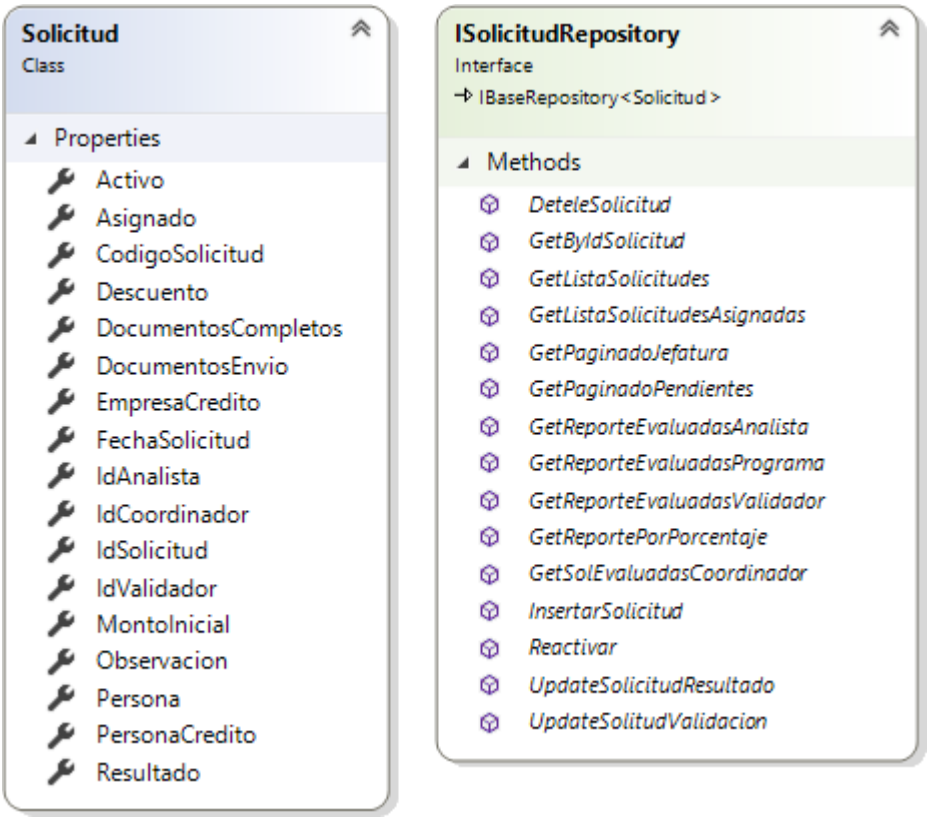
Fuente: Elaboración propia

CLASE SISTEMAFINANCIEROPERSONA E INTERFAZ ISISTEMAFINANCIEROPERSONAREPOSITORY



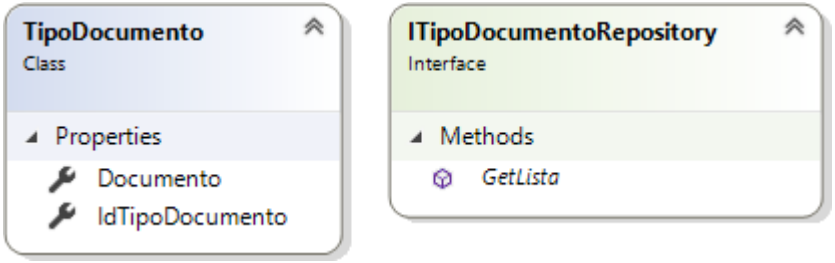
Fuente: Elaboración propia

CLASE SOLICITUD E INTERFAZ ISOLICITUDREPOSITORY



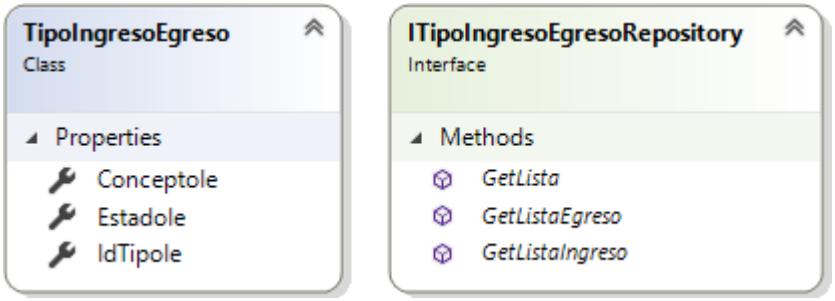
Fuente: Elaboración propia

CLASE TIPODOCUMENTO E INTERFAZ ITIPODOCUMENTOREPOSITORY



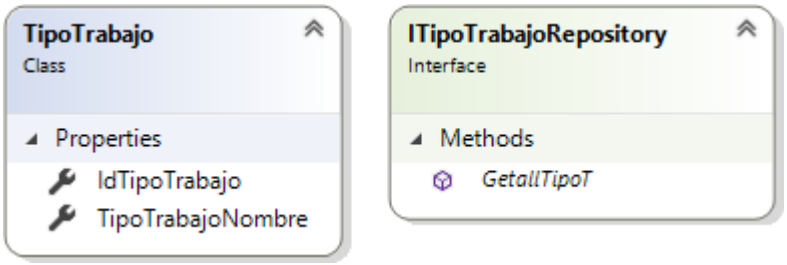
Fuente: Elaboración propia

CLASE TIPOINGRESOEGRESO E INTERFAZ ITIPOINGRESOEGRESOREPOSITORY



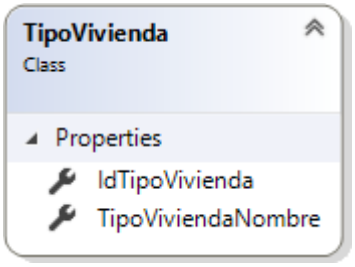
Fuente: Elaboración propia

CLASE TIPOTRABAJO E INTERFAZ ITIPOTRABAJOREPOSITORY



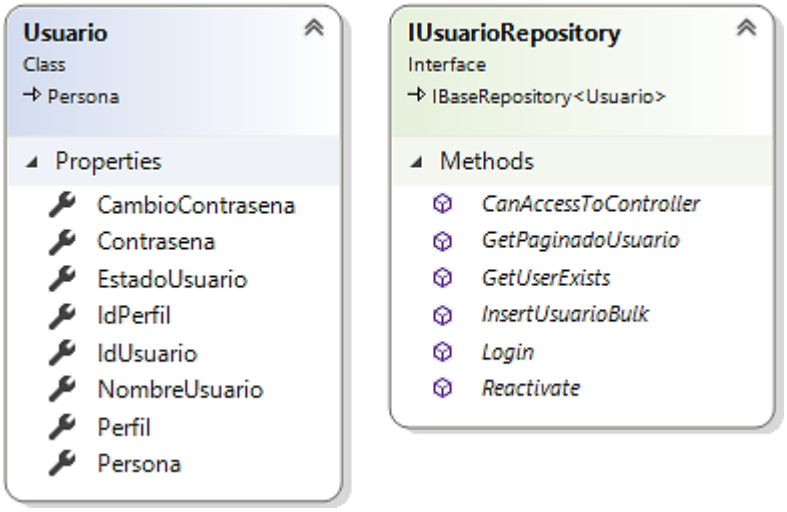
Fuente: Elaboración propia

CLASE TIPOVIVIENDA



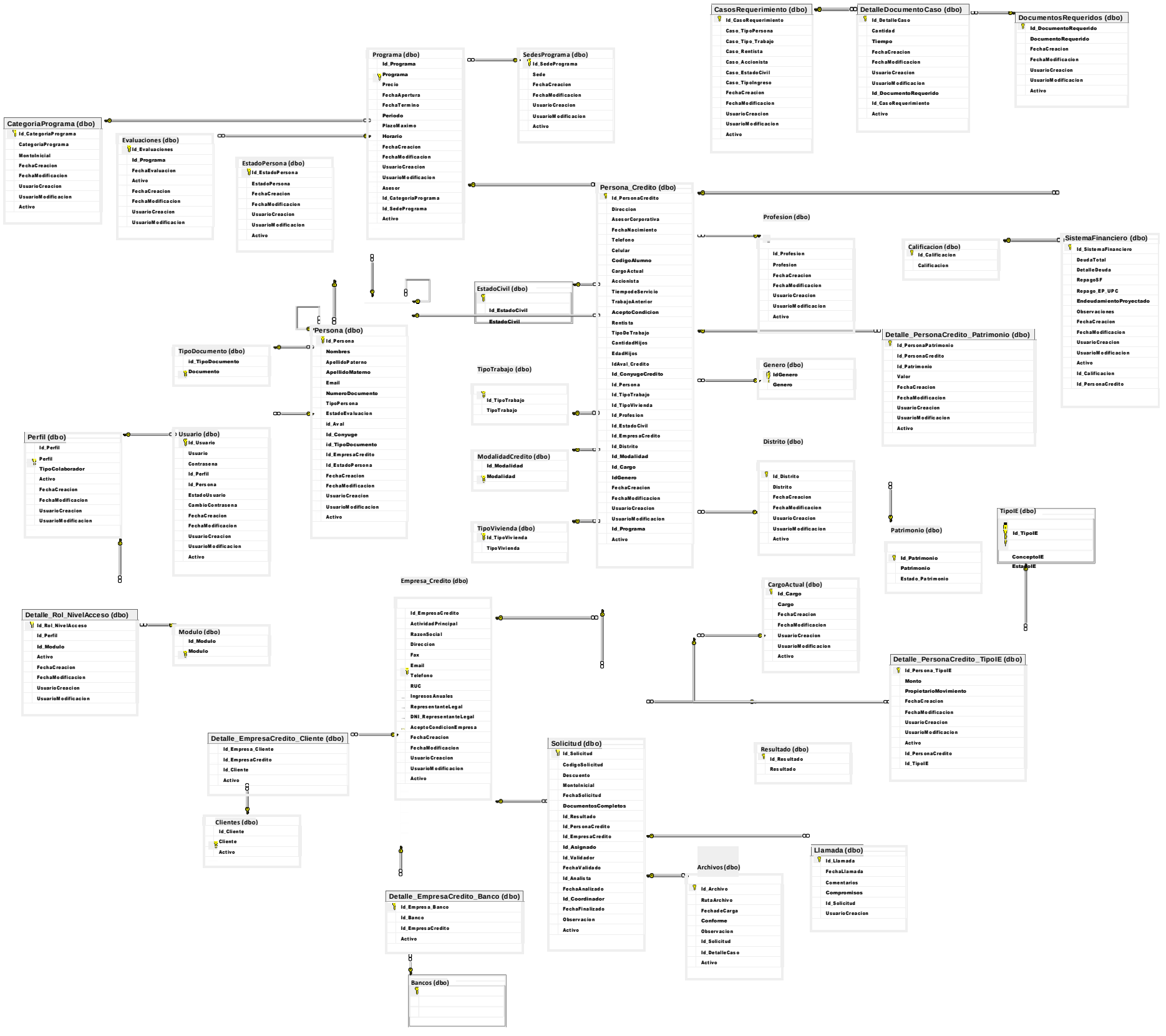
Fuente: Elaboración propia

CLASE USUARIO E INTERFAZ IUSUARIOREPOSITORY



Fuente: Elaboración propia

ANEXO 3: ASE DE DATOS



Fuente: Elaboración propia

Al momento de realizar el despliegue de la base de datos para el proyecto desplegado en este trabajo de tesis, se eligió el nombre siecdb.

El diseño de las tablas y del diagrama físico de base de datos, que se muestra abajo, ha sido trabajado con el software SQL Server Management Studio.

A continuación, se muestran las tablas que se manejan en este diseño de base de datos.

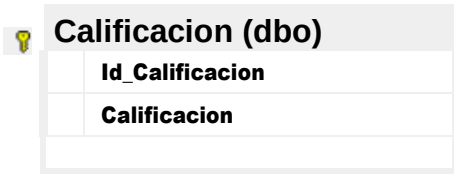
TABLA ARCHIVOS

Fuente: Elaboración propia

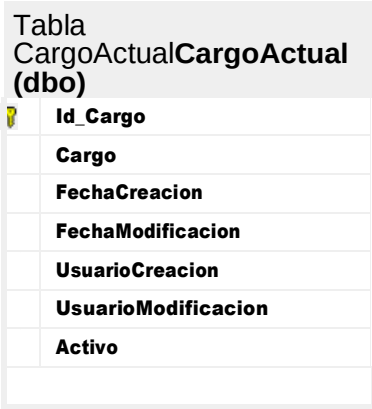
TABLA BANCOS

Fuente: Elaboración propia

TABLA CALIFICACIÓN



Fuente: Elaboración propia

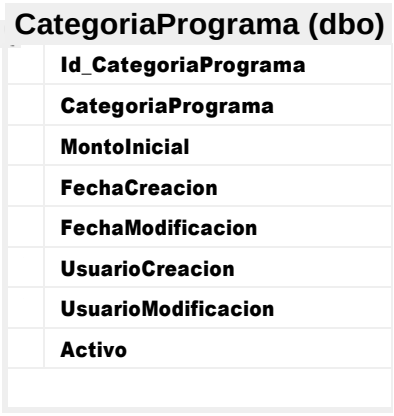


Fuente: Elaboración propia

TABLA CASOSREQUERIMIENTO

Fuente: Elaboración propia

TABLA CATEGORIAPROGRAMA



Fuente: Elaboración propia

TABLA CLIENTES

Fuente: Elaboración propia

TABLA DETALLE_EMPRESACREDITO_BANCO

Fuente: Elaboración propia

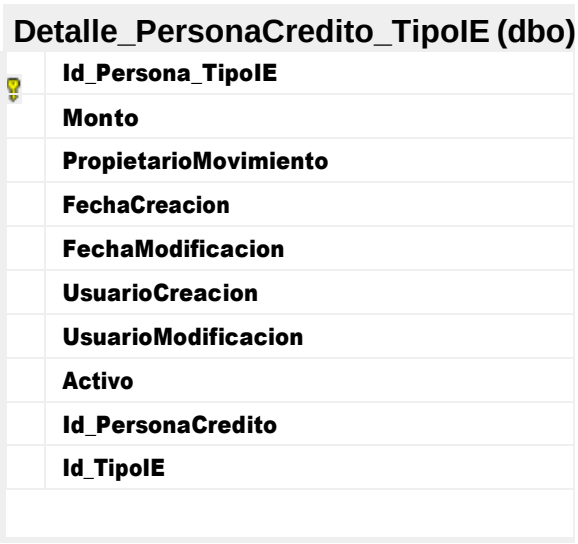
TABLA DETALLE_EMPRESACREDITO_CLIENTE

Fuente: Elaboración propia

TABLA DETALLE_PERSONACREDITO_PATRIMONIO

Fuente: Elaboración propia

TABLA DETALLE_PERSONACREDITO_TIPOIE

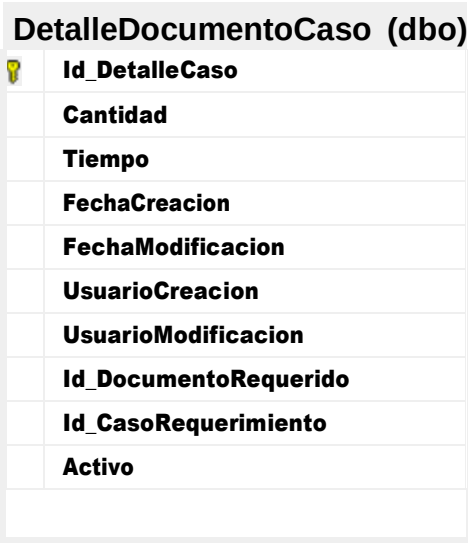


Fuente: Elaboración propia

TABLA DETALLE_ROL_NIVELACCESO

Fuente: Elaboración propia

TABLA DETALLEDOCUMENTOCASO

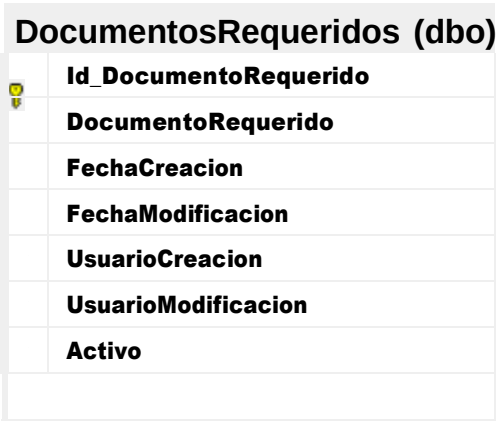


Fuente: Elaboración propia

TABLA DISTRITO

Fuente: Elaboración propia

TABLA DOCUMENTOSREQUERIDOS



Fuente: Elaboración propia

TABLA EMPRESA_CREDITO

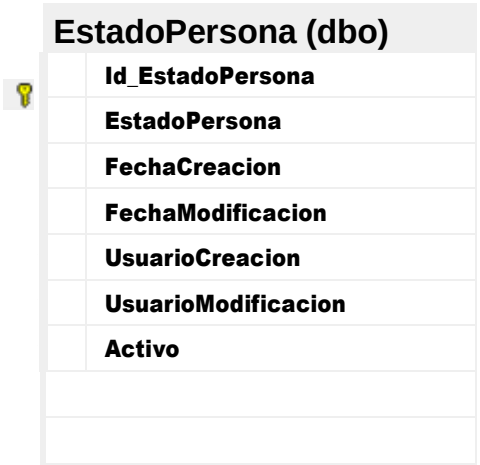
AceptoCondicionEmpresa
FechaCreacion
FechaModificacion
UsuarioCreacion
UsuarioModificacion
Activo

Fuente: Elaboración propia

TABLA ESTADOCIVIL

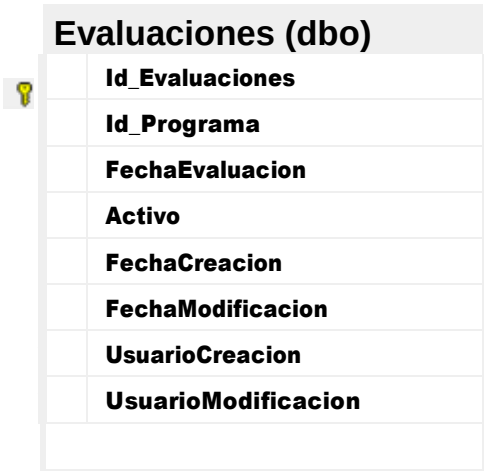
Fuente: Elaboración propia

TABLA ESTADOPERSONA



Fuente: Elaboración propia

TABLA EVALUACIONES



Fuente: Elaboración propia

TABLA GÉNERO

Fuente: Elaboración propia

TABLA LLAMADA

Fuente: Elaboración propia

TABLA MODALIDADCREDITO

Fuente: Elaboración propia

TABLA MODULO

Fuente: Elaboración propia

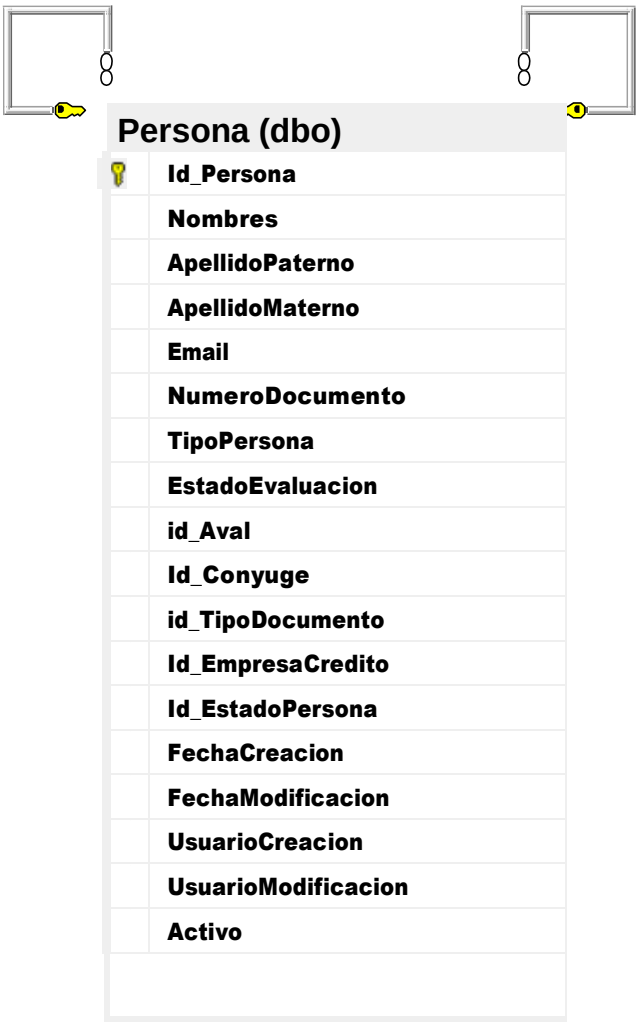
TABLA PATRIMONIO

Fuente: Elaboración propia

TABLA PERFIL

Fuente: Elaboración propia

TABLA PERSONA



Fuente: Elaboración propia

TABLA PERSONA_CRÉDITO

Persona_Credito (dbo)
Id_PersonaCredito
Direccion
AsesorCorporativa
FechaNacimiento
Telefono
Celular
CodigoAlumno
CargoActual
Accionista
TiempodeServicio
TrabajoAnterior
AceptoCondicion
Rentista
TipoDeTrabajo
CantidadHijos
EdadHijos
IdAval_Credito
Id_ConyugeCredito
Id_Persona
Id_TipoTrabajo
Id_TipoVivienda
Id_Profesion
Id_EstadoCivil
Id_EmpresaCredito
Id_Distrito
Id_Modalidad
Id_Cargo
IdGenero
FechaCreacion
FechaModificacion
UsuarioCreacion
UsuarioModificacion
Id_Programa
Activo

Fuente: Elaboración propia

TABLA PROFESION

Fuente: Elaboración propia

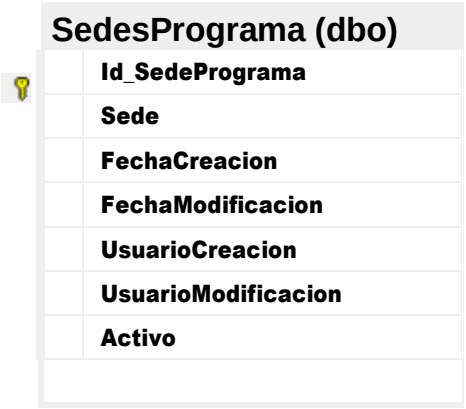
TABLA PROGRAMA

Fuente: Elaboración propia

TABLA RESULTADO

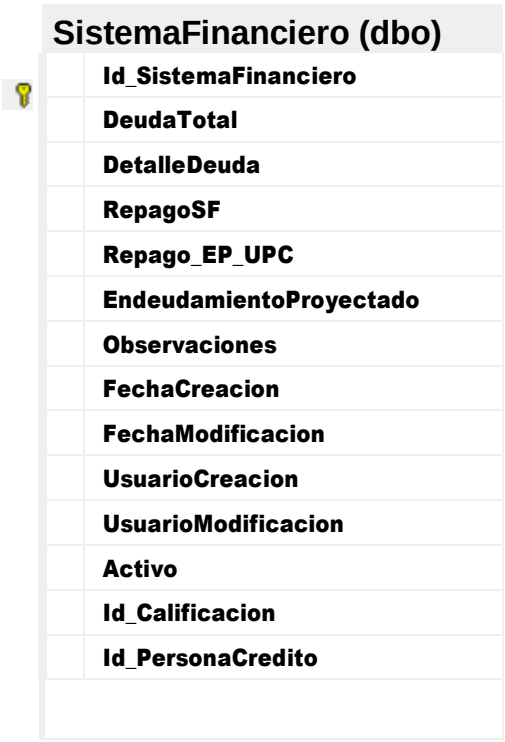
Fuente: Elaboración propia

TABLA SEDESPROGRAMA



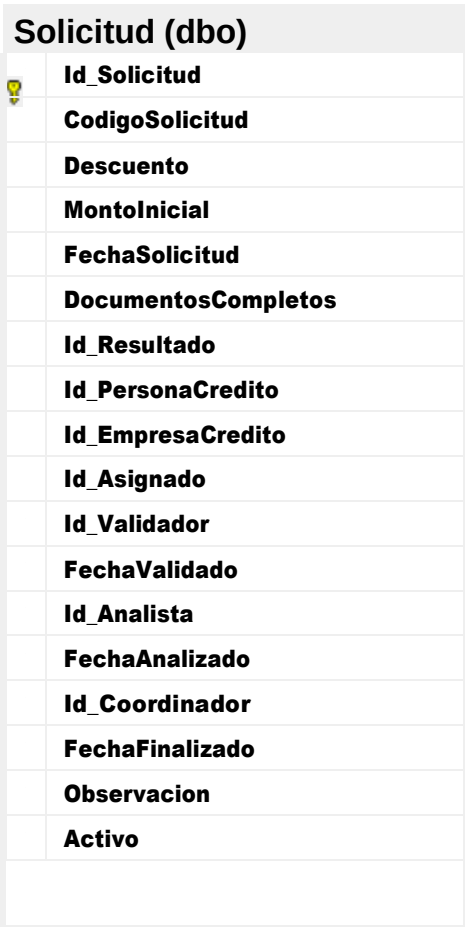
Fuente: Elaboración propia

TABLA SISTEMAFINANCIERO



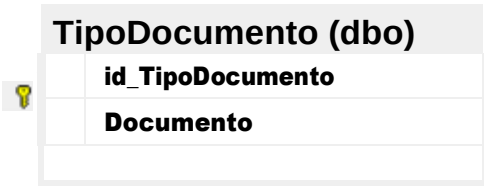
Fuente: Elaboración propia

TABLA SOLICITUD



Fuente: Elaboración propia

TABLA TIPODOCUMENTO



Fuente: Elaboración propia

TABLA TIPOIE

Fuente: Elaboración propia

TABLA TIPOTRABAJO

Fuente: Elaboración propia

TABLA TIPOVIVIENDA

Fuente: Elaboración propia

TABLA USUARIO

Fuente: Elaboración propia

DOCUMENTO DE SOLICITUD GENERADO

Una vez finalizado el registro del postulante en el sistema, éste devuelve un documento en formato PDF, el cual contiene un resumen de la solicitud emitida por el postulante y será utilizado por los actores involucrados en el proceso de evaluación.

Por razones de privacidad, se han tapado zonas en las que aparecen las marcas o nombres de las empresas de la ONG o información sensible de la misma.

Programa de Administración de Crédito de Estudios - Escuela de Post Grado
Universidad

Solicitud de Credito

Asesora Corporativo **Adolfo Ibarra Landeo**

Requisitos

- NO TENER INFORMACIÓN ADVERSA, NI MOROSIDAD EN EL SISTEMA FINANCIERO.
- NO TENER DEUDA PENDIENTE EN UPC.
- CAPACIDAD DE PAGO SUSTENTABLE.

Documentación Requerida (POSTULANTE)

- NO TENER INFORMACIÓN ADVERSA, NI MOROSIDAD EN EL SISTEMA FINANCIERO.
- NO TENER DEUDA PENDIENTE EN UPC.
- CAPACIDAD DE PAGO SUSTENTABLE.

Constancia de Ingreso

a.
b.
c.

Información Personal

Nombres	Martín					
Apellido Paterno	Silva					
Apellido Materno	Peralta					
DNI / CI / CE	11223344					
Dirección	Urb. Virgen de Chapi 185					
Distrito	La Molina					
Teléfono	2661899	**Celular**	9599	**E-mail**	@outlook.com	
Vivienda	Propia					
Estado Civil	Soltero	**Nº de hijos**		**Edades**		
Fecha de Nacimiento	**Día**	26	**Mes**	2	**Año**	1993
Profesión	Abogado					
Tipo de Trabajo	**Dependiente**	X		**Independiente**		
Centro Laboral	IOT Ent Comp					
Dirección	Av. Virgen de Chapi 186					
Teléfonos		**Email**	demo@iotientcomp.com			
Cargo Actual	Analista Desarrollador					
Tiempo de Servicio	5 Meses					
Trabajo Anterior						
Ingreso Neto	**Fijo (S/.)**	3,500.00		**Variable (S/.)**	1,500.00	
Otros Ingresos	**Profesional Independiente (S/.)**	2,500.00		**Otros (S/.)**	0.00	
T. Ingresos Familiares / mes	S/. 0.00					
T. Gastos Familiares / mes	S/. 0.00					

I. Datos del Cónyuge

Nombres	
Apellido Paterno	
Apellido Materno	
DNI / CI / CE	
Centro Laboral	
Cargo	
Teléfonos	

Fuente: Elaboración propia

Programa de Administración de Crédito de Estudios - Escuela de Post Grado
Universidad

Tiempo de Servicio	
Ingreso Fijo Neto (S/.)	0.00
Otros Ingresos (S/.)	0.00

II. Información Patrimonial

Activos (Propiedades)			
Vehículos	0	**Valor Referencial(S/.)**	S/. 0.00
Inmuebles	1	**Valor Referencial(S/.)**	S/. 450,000.00
Otros	0	**Valor Referencial(S/.)**	S/. 0.00
Pasivos (Deudas)			
Importe en S/.	S/. 3,500.00		
Corto Plazo	S/. 2,000.00	**Línea de Tarjetas**	S/. 1,500.00
Hipotecario	S/. 0.00	**Alquileres**	S/. 0.00
Otros	0.00		

III. Datos Adicionales

Forma de Pago			
Cuota Inicial S/.	S/. 15,000.00	**Nro de meses**	21
Modalidad	Simple		

DECLARO QUE LA INFORMACIÓN VERTIDA EN ESTA SOLICITUD ES CIERTA, AUTORIZANDO A LA UNIVERSIDAD PERUANA DE CIENCIAS APLICADAS SAC, VERIFICAR LOS DATOS POR QUE ESTIME CONVENIENTE.

_____ DE_____ Del 201____

_____
Firma del alumno

Fuente: Elaboración propia

ANEXO 4: CRIPT DE AUTOMATIZACIÓN DE DESPLIEGUE

Microsoft Azure ofrece scripts de automatización de recursos, que pueden ser creados a nivel de los grupos de recursos o a nivel de cada servicio, según se necesite. Para este proyecto, se ha creado el script de creación del grupo de recursos integral, el cual incluye 29 parámetros y 28 recursos, los cuales se cargan en formato JSON al destino.

Debido al tamaño del código fuente de este script, se mostrará una parte del mismo, de cuatro páginas, incluyendo los puntos más importantes del mismo.

```
    "$schema": "https://schema.management.azure.com/schemas/2015-01-
01/deploymentTemplate.json#",

    "contentVersion": "1.0.0.0",
    "parameters": {

        "servers_siecsrv_name": {
            "defaultValue": "siecsrv",
            "type": "String"

        },

        "serverfarms_siecSP_name": {
            "defaultValue": "siecSP",
            "type": "String"

        },

        "sites_siec_dev_service_name": {
            "defaultValue": "siec-dev-service",
            "type": "String"

        },

        "components_siec_dev_service_name": {
            "defaultValue": "siec-dev-service", "type":
            "String"

        },

        "databases_siecdb_name": {
            "defaultValue": "siecdb",
            "type": "String"

        },

        "config_web_name": {
            "defaultValue": "web",
            "type": "String"

        },

        "keys_ServiceManaged_name": {
            "defaultValue": "ServiceManaged",
```

Código Fuente A.1: Fragmento de código fuente de script de automatización Fuente: Documento de exportación de infraestructura de Microsoft Azure

MODELO DE ENCUESTA EMPLEADO EN LAS PRUEBAS DE USABILIDAD

Figura 84. Modelo de encuesta de usabilidad (1 de 2)

ENCUESTA DE USABILIDAD

NOMBRE DE LA ORGANIZACIÓN:		
NOMBRE DE LA APLICACIÓN: SIEC (Sistema Integral de Evaluación Crediticia)		
ÁREA: Créditos Educativos	**REALIZADA POR:** Adolfo Ibarra Landeo	**FECHA:** 28-06-2018

La presente encuesta tiene por objetivo recabar información para identificar el nivel de aceptación y la experiencia del usuario, correspondientes a las pruebas de usabilidad del sistema desarrollado.

INSTRUCCIONES: Marcar con una equis (X) en la escala del 1 al 5, aproximándose 1 a "En desacuerdo" y 5 a "Totalmente de acuerdo", según considere que la respuesta se adecúe mejor a su experiencia de uso con el sistema.

1. La aplicación carga rápidamente.

1	**2**	**3**	**4**	**5**

2. El sistema presenta la información de forma clara y comprensiva.

1	**2**	**3**	**4**	**5**

3. El tipo y tamaño de letra empleados en el sitio permiten su manejo sin esfuerzo visual.

1	**2**	**3**	**4**	**5**

4. La apariencia de la aplicación web no se distorsiona en la pantalla, de escritorio o portátil, que utilizo.

1	**2**	**3**	**4**	**5**

5. La navegación con el botón TAB es posible, facilitando la navegación.

1	**2**	**3**	**4**	**5**

6. Las páginas existentes en el sistema muestran contenido bien escrito, sin errores ortográficos.

1	**2**	**3**	**4**	**5**

7. Existe claridad sobre la función de los botones, las listas y los controles proporcionados por el sistema.

1	**2**	**3**	**4**	**5**

Fuente: Elaboración propia

Figura 85. Modelo de encuesta de usabilidad (2 de 2)

8. Existe coherencia en la distribución de los elementos presentes en todas las páginas de la aplicación.

1	2	3	4	5

9. El sistema me permite realizar todas las funciones que esperaba.

1	2	3	4	5

10. El funcionamiento del sistema es sencillo de entender y manejar.

1	2	3	4	5

11. Los documentos que me otorgaron, a modo de manuales de ayuda, son claros.

1	2	3	4	5

12. Pude replicar el funcionamiento detallado en los manuales de ayuda sin dificultad.

1	2	3	4	5

13. El sistema realiza la validación de la información ingresada para evitar errores (Por ejemplo: no ingresar texto en campos destinados a soportar números).

1	2	3	4	5

14. El sistema solicita confirmación antes de realizar acciones que serán registradas.

1	2	3	4	5

ENCUESTADO: ENCUESTADOR:

Fuente: Elaboración propia